

Learning and leveraging shared domain semantics to counteract visual domain shifts

Présentée le 8 mai 2020

à la Faculté informatique et communications
Laboratoire de vision par ordinateur
Programme doctoral en informatique et communications

pour l'obtention du grade de Docteur ès Sciences

par

Róger BERMÚDEZ CHACÓN

Acceptée sur proposition du jury

Prof. A. Argyraki, présidente du jury
Prof. P. Fua, directeur de thèse
Dr G. Csurka, rapporteuse
Prof. V. Lepetit, rapporteur
Prof. S. Sússtrunk, rapporteuse

... where one's been...
 to ride the spiral to the end and may just go where one's been...
 we'll be together and following us...
 will be together and following us...
 will be together and following us...
 will be together and following us...

lateralus.

To Wile E. Coyote

Acknowledgements

Doing a PhD is a personal journey, but it is far from an individual merit only. I wish to express my gratitude to all who contributed in one way or another during my doctoral studies.

I want to thank Professor Pascal Fua for his supervision, scholarly feedback, and for putting together such a great lab, where I have had the privilege to learn and grow. I would also like to express my deepest gratitude to Dr. Mathieu Salzmänn, who over the years provided me with the scientific guidance, invaluable feedback, and a much needed level head in times of academic despair.

I presented my PhD dissertation to Prof. Aikaterina Argiraki, Dr. Gabriela Csurka, Prof. Vincent Lepetit, and Prof. Sabine Süssstrunk. Our discussion and their assessments undoubtedly improved the quality of this work, for which I want to express my appreciation.

I have had the pleasure to learn from and interact with brilliant and passionate researchers at EPFL, both at the professional and personal level. To all my former colleagues, thank you for making this such an enriching experience.

To Carlos, Agata, Ανδρέας, Δώρα, Išinsu, राधाकृष्णन, Premysław, Kenneth, Pablo, Anne, Mateusz, Sena, विदित, Mario, and Heinitor: you supported me, fed me, entertained me, and kept me sane, and for that I hereby acknowledge you as alright people. Feel free to quote me on that.

Despite the distance, I always felt the warmth and support of my family throughout my PhD as if they were next to me. To my parents, Marvin and Flor, and to my siblings, Marvin, Gabriel, Daniela, Rafael, Paula, Catalina, Andrés, and Valeria, I wish I were not limited by human anatomy to hug you all at once.

Lastly, I would like to thank Dr. Eirini Arvaniti for her extensive academic, linguistic, philosophical, emotional, recreational, intercultural, vacation, and nutritional support. I have been truly fortunate to co-author countless stories together with such an amazing collaborator in life—going through this PhD not the least intense—and I wish for a future at least as prolific henceforth.

Lausanne, January 4th, 2020

R. B. C

Résumé

L'une des principales limites de l'intelligence artificielle est son incapacité à s'adapter aux situations inconnues. L'apprentissage automatique (ML, de l'anglais "Machine learning"), qui apprend à partir des données, est notablement sensible à cela. Le ML s'appuie sur des observations pour apprendre des règles implicites sur les entrées et les sorties de l'algorithme ainsi que les relations entre elles dans le but de résoudre une tâche. Par conséquent, les algorithmes de ML apprennent en observant une version incomplète et nécessairement biaisée du monde. À cause de cela, les méthodes de ML rencontrent des difficultés pour décider comment utiliser correctement leurs connaissances acquises lorsque le monde change. L'adaptation de domaine, le paradigme suivi dans cette thèse, est une branche de la recherche en ML qui cherche à résoudre le problème susmentionné. Sous l'adaptation de domaine, un problème de ML doit être adapté lorsque des *changements de domaine* — ou des changements dans la nature des données — se produisent. Cette thèse aborde le problème de l'adaptation de domaine dans le contexte particulier des applications visuelles, dans le but ultime de réduire autant que possible le besoin d'intervention humaine durant l'entraînement des méthodes de ML. Nous étudions le problème de l'adaptation de domaine en suivant deux angles différents.

Une première idée consiste à exploiter la structure existante des images. Malgré les différences visuelles, les informations structurelles relatives au contenu sémantique de l'image sont souvent préservées dans des images d'origines différentes. Nous présentons une méthode permettant d'exploiter ces *correspondances visuelles* — même lorsque la correspondance est imparfaite — basée sur l'apprentissage à instances multiples, afin d'ajuster les paramètres d'un modèle de ML à de nouveaux domaines. Nous introduisons également une méthode de ML auto-supervisée qui agrège les éléments visuels conformément à une *carte de fréquentation consensuelle*. Comme ces cartes sont de bons remplacements non supervisés pour les annotations réelles, nous les utilisons comme signal de supervision. De plus, nous proposons une architecture *U-Net à deux flux* qui traite simultanément différents domaines. Il combine régularisation des paramètres, adaptation de la distribution et signal auto-supervisé issu des cartes consensuelles, afin de combler l'écart de performance entre les modèles fonctionnant avec des domaines d'image différents.

La deuxième idée va au-delà des informations brutes sur l'image et transforme plutôt les images en représentations compactes latentes préservant la sémantique de l'image. Pour cela, nous introduisons des *réseaux multiflux* : un paradigme de recherche d'architecture de réseau neuronal qui attribue différentes capacités de réseau à différents domaines d'image afin d'extraire des représentations latentes robustes non liées à un domaine. Dans le paradigme de *réseaux multiflux*, des portes entraînaibles, qui sont spécifiques aux différents domaines, modulent la contribution de différentes opérations à l'encodage. Les résultats finaux sont des représentations latentes de différents domaines qui ne souffrent pas du changement de domaine.

Nos résultats dans la segmentation d'images biomédicales, la classification d'objets et la détection d'objets, valident le large éventail d'applicabilité des méthodes présentées dans cette thèse.

Mots-clés Adaptation de domaine, apprentissage par transfert, apprentissage à instances multiples, apprentissage auto-supervisé, recherche d'architecture de réseau neuronal, imagerie biomédicale.

Abstract

One of the main limitations of artificial intelligence today is its inability to adapt to unforeseen circumstances. Machine learning (ML), due to its data-driven nature, is particularly susceptible to this. ML relies on observations in order to learn implicit rules about the inputs, outcomes, and relationships among them, so as to solve a task. An unfortunate consequence of learning based on observations, however, is that ML algorithms learn by observing a partial, inevitably skewed version of the world. As a result, ML methods experience difficulties deciding how to properly use their learned experience when the world as they know it changes.

Domain adaptation, the paradigm followed in this thesis, is an area of ML research that tackles the above problem. In the domain adaptation setup, a ML problem must be adapted when *domain shifts*—or changes in the nature of the data—occur. This thesis tackles the domain adaptation problem in the particular context of visual applications, with the ultimate goal of reducing as much as possible the need for human intervention in training ML methods for visual tasks. We study the domain adaptation problem from two different fronts.

A first idea is to harness the existing structure of the images. Despite visual differences, structural information related to the semantic content of the image is often preserved in images from different origins. We present a method to leverage those *visual correspondences*—even when the match is imperfect—based on Multiple Instance Learning, so as to adjust parameters of a ML model to new domains. We also introduce a self-supervised ML method that aggregates visual correspondences into a *consensus heatmap*. As such heatmaps are good unsupervised proxies for real annotations, we use them as supervisory signal. In addition, we propose a *two-stream U-Net* architecture that processes different domains simultaneously. The two-stream U-Net combines parameter regularization, distribution matching, and the self-supervised signal from the consensus heatmaps, to bridge the performance gap between models operating on different image domains.

The second line of reasoning looks beyond the raw image information and instead maps images to compact latent representations that preserve image semantics. For this, we introduce *multiflow networks*: a neural Network Architecture Search paradigm that assigns different network capacity to different image domains to extract domain-agnostic latent representations. In the multiflow formalism, domain-specific learnable gates modulate the contribution of different operations to the encoding. The end results are latent encodings from different domains that do not suffer from the domain shift.

Our results in biomedical image segmentation, object classification, and object detection, validate the wide range of applicability of the methods introduced in this thesis.

Keywords Domain adaptation, transfer learning, multiple-instance learning, self-supervised learning, neural architecture search, biomedical imaging.

Contents

Acknowledgements	v
Abstract	vii
List of figures	ix
List of tables	x
1 Introduction	1
1.1 Machine learning	2
1.1.1 Annotating examples	2
1.1.2 The same distribution assumption	3
1.1.3 Domain shift	4
1.2 Reducing annotation	5
1.3 Domain adaptation	6
1.3.1 Applications	7
1.4 Thesis contributions	10
1.5 Thesis outline	11
2 Related work	13
2.1 Working with limited annotations	13
2.2 Domain adaptation	16
2.2.1 Reweighting feature distributions	16
2.2.2 Corresponding examples	16
2.2.3 Manifold alignment	17
2.2.4 Representation learning	17
2.2.5 Adversarial training	18
3 Visual correspondences and Multiple Instance Learning	19
3.1 Visual correspondences	19
3.1.1 Sampling representative locations	22
3.2 Visual correspondences and domain adaptation	24
3.2.1 Multiple Instance Learning formulation	24
3.3 Application: Boosted decision trees for Electron Microscopy	25
3.3.1 Threshold learning regularization	27
3.3.2 Datasets	27
3.3.3 Evaluation metric	27
3.3.4 Experimental setup	28
3.3.5 Baselines	29
3.3.6 Results	30
3.4 Conclusion	31
4 Consensus heatmaps and the two-stream U-Net	33
4.1 Consensus heatmaps	33
4.1.1 From visual correspondences to heatmaps	35

4.1.2	Refining correspondences with heatmaps	38
4.1.3	Heatmaps as soft annotations	39
4.2	The two-stream U-Net	39
4.2.1	General structure	40
4.2.2	Parameter regularization and sharing	41
4.2.3	Feature regularization	42
4.2.4	Segmentation scoring	43
4.2.5	Loss function	44
4.3	Evaluation	44
4.3.1	Datasets	45
4.3.2	Experimental setup	45
4.3.3	Unsupervised domain adaptation	46
4.3.4	Semi-supervised domain adaptation	47
4.3.5	Experiments under large domain shift	48
4.4	Conclusion	50
5	Domain-adaptive multiflow networks	51
5.1	Motivation	51
5.2	Multiflow networks	52
5.2.1	Single domain	52
5.2.2	Domain-adaptive multiflow networks	55
5.2.3	Relationship to neural architecture search	58
5.3	Evaluation	59
5.3.1	Baselines	59
5.3.2	Implementation details	59
5.3.3	Digit recognition	60
5.3.4	Object recognition	62
5.3.5	Object detection	68
5.4	Conclusion	69
6	Concluding remarks	71
6.1	Leveraging semantics from structure	71
6.1.1	Visual correspondences	71
6.1.2	Consensus heatmaps	71
6.2	Learning semantic representations	72
6.2.1	Two-stream U-Net	72
6.2.2	Multiflow networks	72
6.3	Some general insights	72
6.3.1	Model adaptation bias	72
6.3.2	Negative transfer	74
6.4	Future research directions	75
	Bibliography	90
	Curriculum Vitæ	91

List of Figures

1.1	Examples of data annotations	3
1.2	Examples of visual domain shifts	5
1.3	Domain adaptation for biomedical imaging	7
1.4	Domain adaptation for autonomous driving	8
1.5	Domain adaptation for sports	9
1.6	Domain adaptation for forgery detection	9
3.1	Visual correspondences overview	21
3.2	Sampling of relevant positive locations	23
3.3	Example visual correspondences for Electron Microscopy datasets	29
3.4	Jaccard indices for different number of visual correspondences	30
3.5	Visual correspondences segmentation results	31
4.1	Aggregating visual correspondences	34
4.2	Consensus heatmap construction overview	36
4.3	Heatmap construction from mixture of Gaussians	38
4.4	Heatmap construction by overlaying label masks	39
4.5	Refined correspondences via consensus heatmap	40
4.6	The U-Net architecture	41
4.7	The two-stream U-Net architecture	42
4.8	Segmentation results for unsupervised adaptation	47
4.9	Segmentation results for semi-supervised adaptation	48
4.10	Datasets exhibiting large domain shift	49
5.1	Domain-adaptive multiflow networks overview	53
5.2	Multiflow computational unit	53
5.3	Domain specific computations as a combination of multiple flows	56
5.4	Domain adaptation for autonomous driving	58
5.5	Examples from digit recognition datasets.	60
5.6	Multiflow LeNet architecture	61
5.7	Multiflow SVHNet architecture	61
5.8	Multiflow ResNet-50 architecture	64
5.9	Evolution of the gate activations for multiple domains	67
5.10	Multiflow architecture for drone detection	69
6.1	Different paradigms for parameter evolution	73

List of Tables

3.1	Orientations defined by the vertices of platonic solids	21
3.2	Electron Microscopy datasets	28
3.3	Unsupervised domain adaptation results for organelle segmentation on Electron Microscopy images via visual correspondences and Multiple Instance Learning	31
4.1	Unsupervised domain adaptation results for organelle segmentation on Electron Microscopy images with two-stream U-Net and consensus heatmaps	47
4.2	Semi-supervised domain adaptation results for organelle segmentation on Electron Microscopy images with two-stream U-Net and consensus heatmaps . . .	48
4.3	Unsupervised domain adaptation results for organelle segmentation on Electron Microscopy images under large domain shift	49
5.1	Unsupervised domain adaptation results for object classification and detection with domain-adaptive multifold networks	66

Introduction

CHAPTER 1

The field of computer vision seeks to enable computers to understand images.

For humans, understanding images comes naturally. Since we are born, we are exposed to vast amounts of visual information in an almost continuous stream, and this information is presented to us in a particular context and combined with other stimuli. This, in turn, allows us to create complex models of the world and to successfully use them to predict and make decisions based on new visual information. More extraordinarily, we easily *adapt* our understanding of the world to new situations from limited or indirect visual input.

Computers, on the other hand, typically require explicit instructions and are, so far, incapable of creating general and adaptive models of the world. Even programs based on artificial intelligence—arguably designed with this in mind—cannot cope with sudden changes in their input in a way that even remotely matches ours.

Despite this, computers succeed where humans fall short: massive, parallel, rapid, sleepless processing. As more and better algorithms, devices, and techniques are continuously developed, and as more and larger corpora of images are generated by the day, computer-automated image analysis is not only a sensible course of action, for most modern applications it is the only practical one.

In the last decades, computer vision has moved from being mostly an academic subject to becoming an integral part of a large number of disciplines and activities, including art, entertainment, sports, security, manufacturing, medicine, space exploration, agriculture, meteorology, and robotics. Beyond that, surprising areas such as urban planning, ecology, and banking, have also recently found fresh, unforeseen applications for computer vision.

There are two main reasons for this computer vision surge. The first has to do with infrastructure: more powerful hardware, larger throughput, larger storage, omnipresence and low cost of consumer cameras, and the internet. The second reason is software; in particular, the development of large-scale processing methods based on *machine learning*.

Machine learning algorithms, as powerful as they are, have difficulties readjusting to unfamiliar scenarios too. Granting computer vision methods based on machine learning the power to adapt to new situations is the motivation for this dissertation.

1.1 Machine learning

Machine learning (ML) techniques, generally speaking, identify patterns to relate inputs to outputs; those patterns are subsequently used for either describing particular qualities of the input or generating predictions on different inputs. ML is a powerful paradigm with applications in virtually all disciplines that drive the world today. ML is also a thriving area of research which intersects with other scientific branches such as natural language processing, computational biology, computer vision, and speech recognition. This thesis will focus exclusively on ML for computer vision.

ML differs from other previous techniques used in computer vision, in that the decisions of what patterns to identify from images and how to combine and use such patterns is not explicitly stated, but *learned* from examples. In the most simple scenario, known as *supervised* ML, a program is fed with *annotated* examples—images and their corresponding expected response—and a training process finds the right configuration for the program to compute an output as similar as possible to its expected response. By repeating this multiple times for different images, the program generally learns implicit rules to generate the expected response from the input image. With these rules internalized, the program is then able to compute coherent responses when presented with new images.

1.1.1 Annotating examples

Training a supervised ML algorithm requires providing a so-called *supervisory signal*, most commonly in the form of annotated examples, as mentioned above. These annotated examples reflect the understanding of a human about the task, and are therefore, in most of the cases, obtained manually, with an annotator deciding the accepted or expected output given an image. Figure 1.1 shows examples of annotations for different problems.

In a simple scenario such as object classification, the annotations—or *labels*—are simply which object is in a picture, from a predefined pool of possible objects. Other tasks require much more careful annotations. Problems in biomedical imaging, such as characterizing tissues from magnetic resonance imaging (MRI) acquisitions, require providing the algorithm with examples of specific region boundaries where different parts of the tissue are found. The former type of annotations can be obtained quickly, easily, and with little to no ambiguity. The latter, however, requires extremely well trained professionals, and long hours of labor in front of a computer. Manually annotating images can also be prone to subjective criteria that might cause annotations from different people to disagree, and make it difficult to establish a *ground truth* about the expected output.

To make matters worse, most ML algorithms are complex and require annotated data in large amounts to successfully generalize beyond the training images.

As annotating training examples is difficult, expensive, time-consuming, tedious, and error-

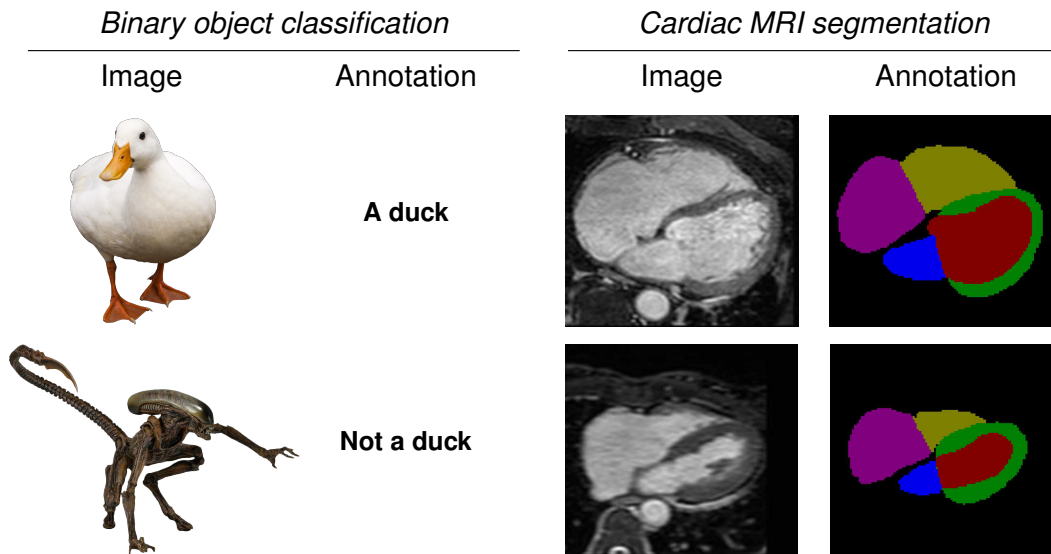


Figure 1.1 Different tasks require simpler or more complex annotations. For a simple binary classification task (left), all that is required is to flag an image as being the object of interest or not. For more complex tasks, such as semantic segmentation of the heart (right) (Zhang et al., 2018b), the annotator must accurately identify particular regions within the image that belong to different parts of the object to segment, shown here with different colors.

prone, there is substantial interest in reducing the need for annotating data, or even eliminating it altogether. Proposing solutions to this problem is the central topic of this dissertation. Chapter 2 offers an overview of current techniques in this direction.

1.1.2 The same distribution assumption

While it is true that learning-based computer vision methods effectively remove the need for manually-crafted image descriptions and models, at the same time they introduce a strict dependency on the training examples. It is now a requirement of the training process to provide images and annotations that fairly represent the relevant aspects of the images according to the task to perform. Those training examples must also be abundant enough to successfully drive the learning process.

We now state more formally the training of a supervised ML method. We define the *training data* as a set $D_{\text{train}} = \{(X_i, y_i); 1 \leq i \leq N\}$ of N *examples*, each composed of an image X_i and a label y_i associated to it. In particular, we assume that the examples are sampled from a joint probability distribution, $(X_i, y_i) \sim P(X, Y)$, defined for a particular joint space $\mathcal{X} \times \mathcal{Y}$ of images and labels.

A *model*, in this context, is a function f with some parameters Θ that is meant to reproduce the relationship between the example images and labels, that is,

$$f(X_i; \Theta) \approx y_i \quad \forall (X_i, y_i) \in D_{\text{train}}, \quad (1.1)$$

and *training* the model means finding the right Θ —the configuration under which the model best approximates the desired output—by inspecting all X_i and y_i from the training set. The ultimate goal of the model is to learn the underlying rules to relate the input with the output in the general case, that is, given a different set of examples $D_{\text{test}} = \{(X_j, y_j) \sim P(X, Y)\}$ —known as a *test set*—drawn from the *same distribution* $P(X, Y)$ as the training set, the model should be able to recover all the y_j by using only the image information X_j and the learned parameters Θ .

1.1.3 Domain shift

Under the same distribution assumption, a supervised ML model is expected to work appropriately when D_{train} and D_{test} are sampled from a common distribution. As a consequence, models trained on images from a given origin cannot be expected to perform well on images from a different origin. The processing defined by the parameters Θ , learned by the model to map images to their outputs, is particular for the training image domain. Processing images from a different origin will therefore demand new annotations and retraining. This violation of the same distribution assumption is known as *domain shift*. Figure 1.2 depicts this issue.

The main source of domain shift occurs when the semantics of the image domains are shared, but the representation is so different that it is safe to consider them separate domains altogether. Consider, for example, a model that classifies animals in photographs. Suppose we train a model on *photographs* of animals and their names. If, after training, we present the model with *drawings* of animals from preschoolers, the model will not guess the species of the drawn animal correctly, as the drawings will most likely not resemble any of the images analyzed during training.

As a secondary source of domain shift, systematic shifts in the sampling process can make the training data under- or overrepresent the statistics of the image domain¹. Back to our example, suppose that, during training, the only images of dogs that are presented to the model come from the two most popular dog breeds—Labrador Retriever and German Shepherd. Once the model is trained, if it is presented with a picture of a Chihuahua, it is very likely that the model will not predict having seen a dog, but rather a rat².

In practice, more subtle domain shifts also affect prediction. One example that we will study in detail is Electron Microscopy imaging of subcellular structures. In such image acquisitions, slight differences in the configuration of the microscopes, or the sample treatment, although not extremely apparent visually, can cause enormous differences in the predictive capabilities of a model.

¹Some experts do not consider sampling bias as domain shift, but rather as a different problem altogether. For the purposes of this thesis, we still consider sampling bias as a source of domain shift, since, in practice, both shifts in representation and sampling disrupt ML models in a similar way.

²—and justifiably so.

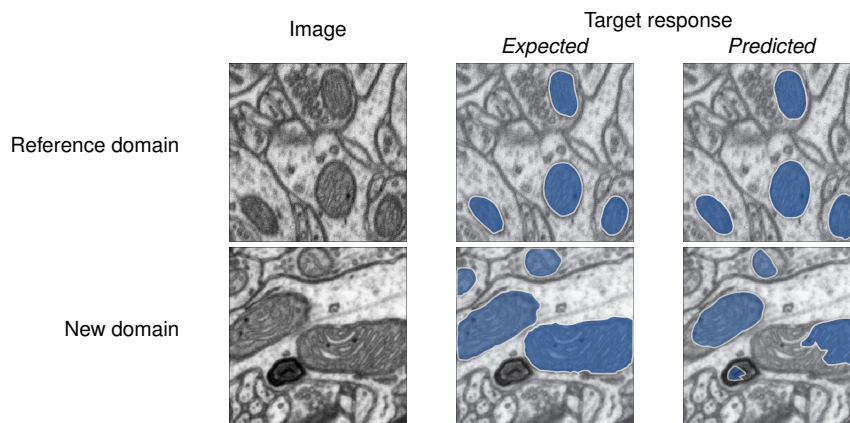


Figure 1.2 Visual domain shift. On an image coming from the same domain as the training data (top), a model successfully predicts the expected output. The same model cannot predict as expected on an image from a new, different domain (bottom), because of the domain shift.

1.2 Reducing annotation

As annotating and training are major bottlenecks for effective ML, there is increasing interest and ample research on how to reuse information encoded in trained models, how to train better and with fewer annotations, how to annotate in a smart way, and how to combine related ML models, in order to avoid or reduce the annotation workload.

These are some popular related paradigms:

- *Semi-supervised learning* techniques leverage small amounts of annotated data in conjunction with other statistics found in an abundant pool of unlabeled data.
- *Multitask learning* methods simultaneously consider annotated examples from separate image domains and tasks for training, constraining the private operations per domain to share common characteristics, arguably requiring less task-specific training data.
- *Meta-learning* ideas revolve around creating mechanisms to extract general information about tasks, that can be reutilized for novel tasks.
- *Active learning* methods minimize the labeling requirements by choosing which examples are the most beneficial to annotate and include in the training set.
- *Transfer learning* methods aim to find shared semantics between a reference image domain and a new image domain and leverage those to modify—or *transfer*—some aspect of the model or the data, to perform well on the new image domain.
- *Domain adaptation* is a specific kind of transfer learning in which the task to solve is the same on both domains³. As this is the main approach that we follow in this dissertation, we provide a more comprehensive definition next.

³There seems to be disagreement in the community about the terms *transfer learning* and *domain adaptation*, and their strict meanings. In practice they are often used interchangeably.

1.3 Domain adaptation

While most of the above ideas lessen the effects of domain shift in an indirect, implicit manner, domain adaptation tackles the domain shift directly.

The main aspect that differentiates domain adaptation from other techniques, in particular from general transfer learning, is the constraint to operate on the same *task*. In this context, the task of a model is the purpose for which the model is conceived, or the problem it aims to solve. In other words, we want to learn the same aspect of the problem, but from two or more different image domains.

Under this constraint, we assume that the two problems share a common label space \mathcal{Y} , and that semantically we want to obtain the same information from the model output from two image domains which are different from one another. This is arguably a more tractable scenario, because the differences between the two problems are ascribed to the image domains only.

The term domain *adaptation* suggests that some aspect of the model changes from one image domain to the other. In the most common setup, we are presented with annotated training examples from a reference domain—the **source domain**, $D^s = P(X^s, Y)$ —where it is possible to perform standard supervised training as explained in Section 1.1.2. The challenge is then to use the source domain D^s and trained model f^s to obtain a different, *adapted* model f^t , which should output suitable values when exposed to data coming from another domain—the **target domain**, $D^t = P(X^t, Y)$ —exhibiting domain shift, that is, $P(X^s) \neq P(X^t)$.

Formally, we define two models operating on different domains,

$$f^s(X_i^s; \Theta^s) \approx y_i^s \quad (X_i^s, y_i^s) \sim D^s, \quad (1.2)$$

$$f^t(X_i^t; \Theta^t) \approx y_i^t \quad (X_i^t, y_i^t) \sim D^t, \quad (1.3)$$

with corresponding terms from either domain, and a shared label space. We can tune f^s with supervision from an annotated training set $D_{\text{train}}^s = \{X_i^s, y_i^s\}$, but we do not have access to the equivalent set of examples from the target domain. At best, we can access unlabeled samples from the target domain $\{X_i^t\} \sim X^t$.

Some common ways to address this include:

- Modifying the target domain images, or some representation of them, so as to resemble those of the source domain.
- Finding target domain images for which it is reasonable to “guess” their labels.
- Crafting source and target domain models so that their corresponding parameters can influence one another.

We offer a more thorough description of these strategies in Chapter 2.

1.3.1 Applications

Domain adaptation mostly favors ML applications that both are data-intensive and for which annotated training examples are difficult to come by. We provide some notable examples of real-world applications below.

Biomedical imaging

Image analysis is instrumental for healthcare and biomedical research. It also extends to revolutionary fields like human-computer interfaces (Hinterberger et al., 2004). Biomedical image modalities include X-ray, ultrasound, tomography, magnetic resonance, and electron microscopy. Each modality exhibits particular idiosyncracies, which in turn require specific, non-trivially transferable expertise to interpret. Unsurprisingly, this implies that automating their analysis is extremely challenging.

A concrete ML problem, depicted in Figure 1.3, is semantic segmentation of the heart (Suri, 2000). The goal is to identify regions that belong to different cardiac structures automatically. Accurate segmentations enable practitioners to evaluate cardiac performance, model heart mechanics, measure cardiac stress, detect deformities, and assist in catheterization. Segmenting heart imaging, however, is as difficult as it is useful. Qualities of the image change with the patient, the condition, the orientation, the modality employed, the settings of the capturing device, and the skill of the operator. This translates into an inconsistent procedure with enor-

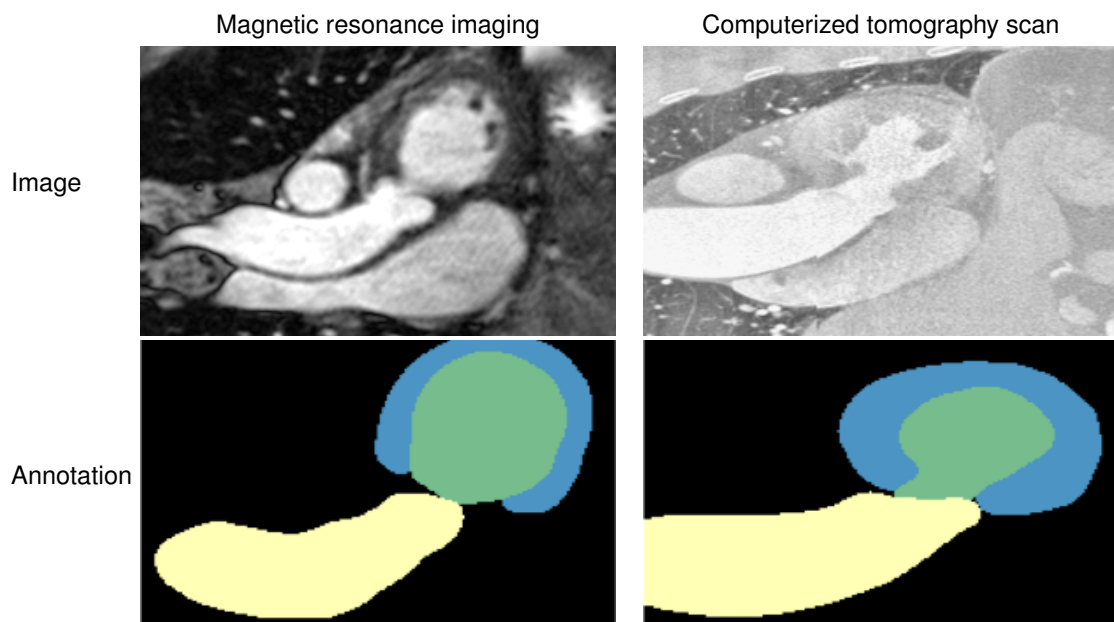


Figure 1.3 *Cardiac semantic segmentation for different image modalities.* Cardiac structures shown in the example (Dou et al., 2018), such as the ascending aorta (yellow), left ventricle blood cavity (green), and left ventricle myocardium (blue), follow similar shapes in different acquisitions, as seen from the segmentations (bottom), but the visual contrast between modalities is large (top).

Chapter 1. Introduction

mous qualitative variance. Moreover, the task itself is not perfectly well defined; it is ambiguous, even for an expert, where exactly the structure boundaries are located in the image.

Despite their current limitations, semantic segmentation, as well as other ML algorithms, is extensively used in medicine and research. Domain-adaptive models make it possible to better utilize preexisting annotated data for new scenarios, hence alleviating the need for manual annotations and extending the reach of automatic image analysis with ML even further. In Chapters 3 and 4, we propose two such models, for adaptation between Electron Microscopy domains, applied to the segmentation of subcellular structures.

Autonomous vehicles

Autonomous vehicles use imagery captured from the surroundings to estimate the positions, sizes, and relative locations of specific objects (detection), understanding the layout of the scene and its precise outlines (segmentation), and predicting moving object trajectories (tracking). As it is unrealistic to provide examples for all possible situations that a vehicle can encounter, domain adaptation is paramount for the general deployment of autonomous vehicles.

A common way to provide source domains for domain adaptation comes from simulations, video games, and other techniques that generate scenes automatically, as shown in Figure 1.4. Here, the objects and actions on a scene are synthesized and hence known beforehand. It is then easy and safe to simulate all sorts of example situations for training—especially dangerous ones—which would otherwise be infeasible or irresponsible to collect in real life. The

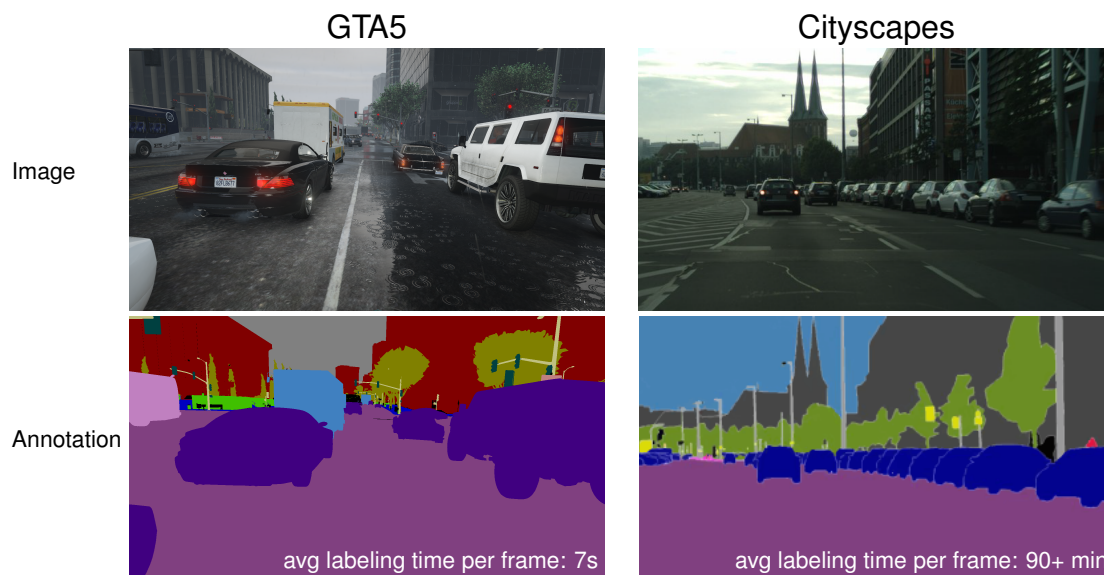


Figure 1.4 Autonomous vehicles can benefit from domain adaptation by leveraging annotated synthetic domains. One example is adaptation from the video game *Grand Theft Auto V*, where annotations are easy to collect (Richter et al., 2016), to real-world scenarios like the ones found in the *Cityscapes* dataset (Cordts et al., 2016), where annotating takes orders of magnitude longer.

idea of using artificial training data is showcased in Chapter 5. We apply domain adaptation to the detection task in drones—crucial for avoiding mid-air collisions—where we leverage synthetically-generated images of drones for weakly-supervised training.

Other applications

Sports. Action recognition—or identifying actions of a subject from successive estimations of their poses in video frames over short time periods—can be used to automate strategy analysis in sports. Figure 1.5 provides an example of this. Domain adaptation can translate this analysis to different sports and activities (FarajiDavar et al., 2011; Jamal et al., 2018). Other related problems, such as crafting *smart personal trainers* to assist players or enthusiasts to perfect their movements and technique (Huber et al., 2017; VAY Sports, 2019), can also benefit from action information encoded in different domains.

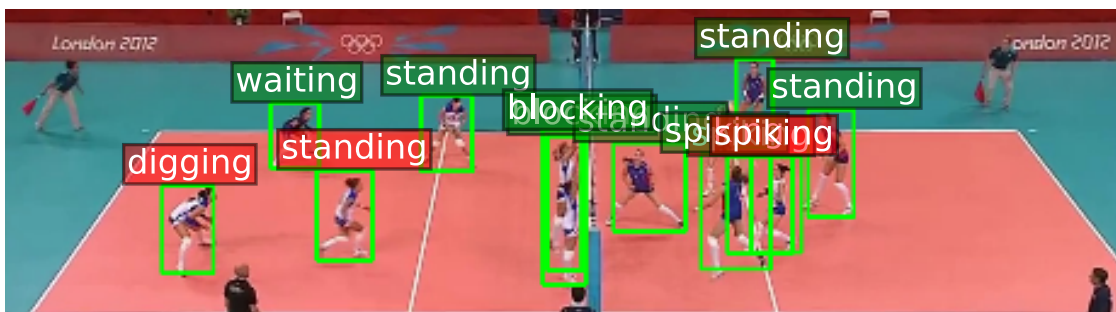


Figure 1.5 Action recognition via ML (Bagautdinov et al., 2017) in an olympic volleybal match.

Forgery detection. Image manipulation software is widely used; manipulated images are shared easily over the internet; generative deep network algorithms are becoming better and better at producing convincingly real images and videos. For all those reasons, there is an increasing need for assessing the authenticity of visual content, as shown in Figure 1.6.

Domain adaptation has been applied to the image edition detection scenario (Cozzolino et al., 2018; Rössler et al., 2019), where models trained on synthetic image editions can inject their knowledge into target domain models to solve the problem on a real image-doctoring domain.

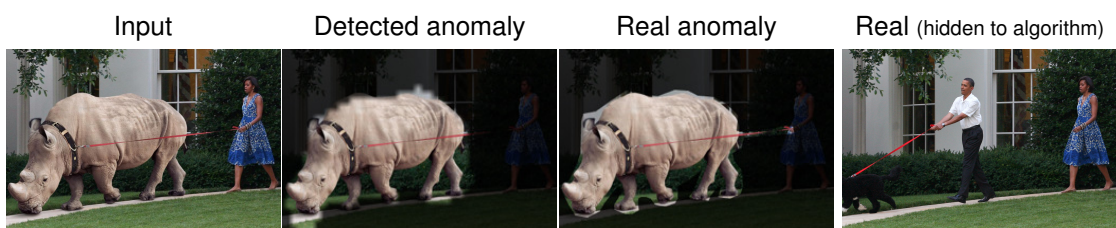


Figure 1.6 Given an altered input image, forgery detection algorithms (Huh et al., 2018) can identify regions that have been edited in (Detected anomaly), which largely coincide with the actual edition (Real anomaly).

1.4 Thesis contributions

A central question at the heart of most domain adaptation methods is what can we find in common between domains to guide the adaptation. This thesis explores two possible answers to this question, stated below, which in turn unfold into a number of domain adaptation methods. We validate these methods in different tasks in biomedical and natural images.

— **Leveraging shared image semantics.** As a first vehicle for domain adaptation, we consider the special case of structured annotations, such as the ones provided for segmentation tasks. In this case the labels exhibit local features that describe and relate to the semantics of the segmented object—which we presume are transferable to new domains. Two complementary observations govern our analysis. First, that by comparing context and visual hints, humans are robust to the domain shift; we attempt to emulate this trait. Second, that local structures present in the annotations are similar between domains.

We propose in Chapter 3 an algorithm that exploits these remarks under a Multiple Instance Learning formulation. We achieve solid performance when adapting simple models for segmentation of Electron Microscopy images using this method. We revisit and further develop these ideas for deep neural networks based on a special architecture, the *two-stream U-Net*, in Chapter 4. By combining such architecture with the consensus heatmap—a spatial encoding of the correspondences between image domains—we achieve an additional boost in segmentation performance.

— **Learning shared image semantics.** In a different basis for domain adaptation, instead of exploiting known shared semantics, we aim to learn them. The core idea is based on the popular strategy of projecting images from different domains onto a common latent representation. The two-stream U-Net of Chapter 4 implements such a criterion to assist adaptation. In addition, we propose a general-purpose framework, predicated on the observation that different image domains exhibit different complexities, and may require being processed by models with different capacity to arrive to the shared latent space.

The materialization of these observations—the *domain-adaptive multiflow network* (DAMNet), described in Chapter 5—corresponds to a dynamic architecture learning strategy. The specific operations that images from different domains undergo in a DAMNet is decided during training. DAMNets are backed by a large performance gain compared to other techniques, as extensively demonstrated in image classification and object detection tasks.

Broad-view insights. The reference data and model parameters obtained or obtainable from the source domain influence and bias domain adaptation. This thesis gives two core insights about the effects of this bias. First, that domain adaptations that strongly preserve and enforce the bias towards the source domain benefit from exploiting known domain semantics, such as shared structure in the annotations. Second, that modulating this bias automatically—for example by allowing different domains to undergo different but related operations—results in more flexible models that acknowledge and embrace the domain shift, and are more effective for extracting shared semantics from different domains.

1.5 Thesis outline

- In **Chapter 1. Introduction** we have established the conceptual framework and summarized the contributions of this thesis.
- **Chapter 2. Related work** is a short review of methods related and relevant to this work.
- **Chapter 3. Visual correspondences and Multiple Instance Learning** presents an algorithm based on finding correspondences between different images and combining them in an unsupervised way. We focus on the segmentation problem in Electron Microscopy images of cerebral tissue. This work appears in Bermúdez-Chacón et al. (2016, 2019).
- In **Chapter 4. Consensus heatmaps and the two-stream U-Net**, we build upon the previous formalism and propose a domain-adapting deep network architecture and a mechanism to infer soft labels from interdomain correspondences, where they are used for unsupervised segmentation of Electron Microscopy images. This chapter presents published work from Bermúdez-Chacón et al. (2018, 2019).
- **Chapter 5. Domain-adaptive multiframe networks** proposes a powerful, general technique for learning multi-domain deep network architectures. We evaluate the method over different domain adaptation tasks in unsupervised settings on natural images. The contents of this chapter are based on the findings appearing in Bermúdez-Chacón et al. (2020).
- Lastly, in **Chapter 6. Concluding remarks**, we offer a discussion of the thesis in general, closing remarks and possible avenues of research inspired by this work.

As motivated in the previous chapter, this thesis proposes several strategies for reducing annotation efforts via domain adaptation methods.

We present here some related techniques proposed in the literature to tackle the general problem of reducing training and annotating effort. Then, we present the state-of-the-art in domain adaptation and contrast it with the ideas proposed in this thesis.

2.1 Working with limited annotations

Fine-tuning. Fine-tuning is perhaps the simplest and most widely-used technique to jump-start training. The idea is to use parameters from the source domain model as initial conditions for those of the target domain, and, *under supervision*, resume training on the target data. When the source domain is abundant and rich, and the source model is expressive enough, fine-tuning is known to be much more effective than training from scratch with limited training examples (Donahue et al., 2014; Hoffman et al., 2013; Lin et al., 2017).

Some insights about the effects of fine-tuning have been both empirically and formally established. Cireřan et al. (2012) employs a fine-tuning schedule that progressively unfreezes and retrains deep layers of neural networks, which accelerates training. Yosinski et al. (2014) provides a method to quantify the effects of fine-tuning layers at different levels in neural networks.

Fine-tuning has been extensively applied in natural (Girshick, 2015; Yanai and Kawano, 2015; Campos et al., 2017; Kornblith et al., 2019) and biomedical imaging (Tajbakhsh et al., 2016; Zhou et al., 2017), natural language processing (Devlin et al., 2018; Eisenschlos et al., 2019), and robotics (Nagabandi et al., 2018).

Semi-supervised learning. Semi-supervised learning exploits relationships between unlabeled examples, along with a small amount of annotated examples, to guide the training.

Popular techniques are based on diffusing labels along edges of a graph connecting neighboring examples in feature space (Zhu and Ghahramani, 2002; Zhou et al., 2004). Other methods exploit properties of the density of the underlying feature distribution, such as low- (Chapelle

et al., 2006) or high-density regions (Ratsaby and Venkatesh, 1995). Another idea—so called Learning by Association—is to learn a latent encoding from labeled and unlabeled data where class assignments and similarities between examples across domains are consistent (Häusser et al., 2017b).

Gradient-based unsupervised methods are often adapted to the semi-supervised setting by adding terms to the optimization criterion that consider the additional supervised signal (Rozantsev et al., 2019; Tran, 2019). The method that we present in Chapter 4 uses this tactic.

Weakly-supervised learning. For some problems, obtaining approximate manual annotations can sometimes be done easily, or even carried out without user intervention. Leveraging the signal that those approximate annotations carry is known as weakly-supervised learning (Zhou, 2017).

For example, recognizing multiple faces from news pictures given their caption, where the labeling is ambiguous, is addressed by Zeng et al. (2013); their method encodes relationships between pictures and face identities into an optimizable low-rank matrix. Koziński et al. (2018) is another example of this; it assists the 3D segmentation with 2D annotations by exploiting the visual hull imposed by the maximum intensity projections of the annotations. Similarly, Tekin et al. (2017) proposes learning neural networks that fuse operations working on different modalities—images and 2D pose annotations for 3D pose estimation in their case—at a given, learnable depth. Rad et al. (2018) also leverages multimodality information, by exploiting a depth channel generated synthetically to assist 3D annotations for pose estimation. Yet another way to use approximate annotations is to infer example prototypes present on the data, so as to denoise labels (Jiangfan et al., 2019).

One field that has seen huge benefits from weak supervision is biomedical image analysis, where obtaining fine-grained annotations is particularly problematic. Anirudh et al. (2016) uses single pixel locations for lung nodule detections in 3D CT scans. Rajchl et al. (2017) infers segmentations from bounding boxes alone. Hu et al. (2018) exploits approximate segmentation labels to guide 3D deformable registration in medical procedures. Mahapatra et al. (2013); Feng et al. (2017) solve the segmentation problem by relating image-level annotations to specific regions given by superpixels. Wang et al. (2018) uses image-level labels and coarse spatial annotations for lung cancer image classification.

One particularly common weakly-supervised strategy consists in assigning labels to sets of examples describing a feature of the set as a whole. This fuzzy-labeling technique, known as Multiple Instance Learning (MIL), circumvents the need for labeling each example individually. Example applications are found in Xu et al. (2012a,b, 2014), which uses image-level labels to infer patch-level segmentations for histopathology image segmentation, and Zhu et al. (2017), which applies a similar strategy to mammogram classification. The method introduced in Chapter 3 establishes visual correspondences across domains following a MIL formulation, applied to segmentation of Electron Microscopy images.

Active learning. The premise of *active learning* is to analyze unlabeled input and determine a minimum, often increasing set of examples that, if annotated, would benefit the training process the most (Settles, 2010). Active learning has been successfully applied to a variety of tasks in biomedical imaging (Gala et al., 2014; Mosińska et al., 2017; Yang et al., 2017; Zhou et al., 2017) and natural image processing (Vijayanarasimhan and Grauman, 2011; Top et al., 2011; Vijayanarasimhan and Grauman, 2014).

Multi-task learning. Concurrently training a model for multiple tasks on a dataset is known as multi-task learning (Zhang and Yang, 2017). Under multi-task learning, parts of the model observing different aspects of the training data and extracting different outputs collaborate among themselves. By doing so, the model learns to extract more general shared semantics that improve performance for all tasks—inclusive those for which training data is scarce. Multi-task learning often uses abundant annotated data for the different tasks at training.

A prime example of this is the multi-headed architecture of Doersch and Zisserman (2017), which uses a common feature extractor and then forks into specific network streams for classification, detection, and depth prediction tasks.

Meta-learning. Meta-learning, or *learning to learn*, aims at creating models that encode general characteristics from different tasks that are conceptually related to each other; its goal is to easily adapt to new tasks from few examples. Meta-learning is a wide area and the approaches are diverse.

In Hoffman et al. (2014), classifiers are trained on subsets of the labels, for both classification and bounded object detection, to learn general encodings applicable to unseen classes. Ha et al. (2017); Rebuffi et al. (2017) train a network to predict the parameters of another depending on the domain.

The methods of Vinyals et al. (2016); Santoro et al. (2016) combine neural networks with memory modules where new tasks can be compared with tasks seen before. Snell et al. (2017) extracts prototypes related to clusters that can be reused in new tasks that partially overlap with the training tasks.

Other techniques follow different approaches. Andrychowicz et al. (2016) offers a strategy to optimize the design of optimization algorithms itself; their method yields update steps suitable for families of related problems. Finn et al. (2017) presents a technique to compute good initial parameters for a new task by interpolating parameters learned on other tasks.

2.2 Domain adaptation

Domain adaptation assumes the existence of a reference domain where a supervised model can be obtained. Domain adaptation methods thus seek to homologate some shared aspect of the task on both source and target domains, so that they are compatible (Csurka, 2017; Wang and Deng, 2018). We detail the main approaches next.

2.2.1 Reweighting feature distributions

A popular idea for linking source and target domain models regards the features of the source and target examples as probability distributions that can be related to one another. The observation is that, under some circumstances, it is possible to approximate one probability distribution with another via a reweighting scheme. The following equation captures this intuition:

$$P_{D^t}(x) = \frac{P_{D^t}(x)}{P_{D^s}(x)} P_{D^s}(x) = w(x) P_{D^s}(x), \quad (2.1)$$

that is, the target domain density can be expressed as that of the source domain, reweighted by the likelihood ratio $w(x) = \frac{P_{D^t}(x)}{P_{D^s}(x)}$.

The TrAdaBoost algorithm (Dai et al., 2007) exploits this idea to adapt AdaBoost (Freund and Schapire, 1995), which reweights misclassifications, to the domain shift scenario. TrAdaBoost leverages annotations from the source domain alongside a small set of target domain examples with labels. An improvement over this idea for unsupervised adaptation, presented in Habrard et al. (2013), learns a model, common for both domains, that predicts correctly on the annotated source domain, while also maximizing decision boundary margins on the target domain.

Matching distributions by reweighting in a reproducing kernel Hilbert space was also studied in Pan et al. (2008); Gretton et al. (2009). Garcke and Vanck (2014) finds the weights that minimize the empirical Kullback-Leibler divergence between $P_{D^t}(x)$ and $w(x)P_{D^s}(x)$.

Heimann et al. (2013) approximates the likelihood ratios from the output probabilities of an auxiliary classifier that is trained to predict the domain from which examples are taken. A similar strategy is used in Zhang et al. (2018a) for adversarial training in a deep network.

2.2.2 Corresponding examples

When some target domain supervision exists, an intuitive idea is to find examples between domains with known shared semantics. For object recognition, for example, Saenko et al. (2010) applies a kernel-based transformation of the target domain features such that source and target representations are similar for similar objects, and dissimilar for objects from different categories. Other methods based on correspondences find coinciding low-dimension projections for both domains (Zhai et al., 2010) for pose estimation, or constrain the alignment of manifolds learned with deep neural networks, for classification of medical images (Guerrero et al., 2014), and object detection in videos (Donahue et al., 2013).

2.2.3 Manifold alignment

Features extracted from images are assumed to lie on a low-dimensional manifold. One popular domain adaptation technique, when there is limited access to target domain annotations, is to align the projections of source and target domain features onto those manifolds. The intuition is that, if the source and target manifolds coincide, any operation performed on the source domain, possibly under supervision, should also apply to the target domain.

Using a small annotated set of target domain examples to infer the manifold alignment was proposed in Wang and Mahadevan (2011). The methods of Fernando et al. (2013, 2015) work in the unsupervised setting; they use principal components as criteria for the alignment. Other methods exploit different aspects of the manifold configuration to guide the alignment, such as feature correlation (Kuss and Graepel, 2003; Sun et al., 2016; Sun and Saenko, 2016) and mutual information (Memisevic et al., 2011).

2.2.4 Representation learning

Similar to the above strategy, representation learning attempts to homologate the source and target domain representations. The idea in this case is to provide mechanisms to guide the extraction of the domain representations.

A common strategy for this is to favor representations that are quantifiably close from one another. Maximum-Mean Discrepancy (MMD) (Pan et al., 2008) is a popular metric to quantify divergence between representations (Sejdinovic et al., 2013). MMD measures the disagreement between distribution moments in a reproducing kernel Hilbert space. It has been successfully used for domain adaptation (Pan et al., 2010), and extended to neural networks (Tzeng et al., 2014; Long et al., 2015; Csurka et al., 2017a; Rozantsev et al., 2018). Our work in Chapter 4 also uses MMD to assist learning common representations.

Other similar strategies use different terms to constraint the learned representation, such as metrics of smoothness (Donahue et al., 2013), class-wise structure preservation, and affinity (Yao et al., 2015).

Decompositions of the representations into private and shared subspaces is a general technique that, by construction, takes the domain shift into account. Damianou et al. (2012) finds a factorization of the latent space into private and shared subspaces per domain, where the domains are different image modalities—color and depth corresponding images, in their paper. Bousmalis et al. (2016) proposes an encoding-decoding network architecture, with private and shared streams, that learns specific and common latent representations per domain.

Constraining the search process itself is another strategy followed in the literature. This is done, for example, by enforcing the learned representations to follow a geodesic path (Gopalan et al., 2011; Gong et al., 2012; Zheng et al., 2012; Baktashmotlagh et al., 2013).

Other representation learning strategies include reconstruction methods based on autoencoders (Chen et al., 2012; Kan et al., 2015; Clinchant et al., 2016; Csurka et al., 2017b), methods

that enforce bijective associations between source and target examples of the same class (Häusser et al., 2017a,b), and strategies to bias the learning towards representations that produce structurally similar responses between source and target tasks (Tsai et al., 2018). The methods introduced in Chapter 4 follow the latter approach.

2.2.5 Adversarial training

All strategies above attempt to obtain domain-agnostic encodings of source and target domains. Their mechanisms explicitly favor predefined criteria for a specific task and set of conditions. Recent works have instead attempted to infer the appropriate encodings implicitly, by making use of adversarial training.

The most common idea to incorporate adversarial training into the encoding extraction is to make use of an auxiliary *domain classifier*—a network that identifies, given an encoding, to which domain the original example belongs. Adversarial training will attempt to deceive the domain classifier into believing that target domain encodings come from the source domain. As a result, the feature encoding process will produce representations for both domains that are, to the domain classifier, indistinguishable from one another.

A referent method of this type is that of Ganin and Lempitsky (2015). Their formulation adds to the above idea a particular operation—the Gradient Reversal Layer—that enables adversarial training to be incorporated into gradient based optimization techniques. Many modern domain adaptation methods use this idea as part of their pipelines (Tzeng et al., 2017; Rozantsev et al., 2018, 2019). The multifold method that we introduce in Chapter 5 also leverages this idea to obtain domain-agnostic encodings.

A conceptually similar formulation, based also on confusing a domain classifier, is presented in Hoffman et al. (2018). Their method, named CyCADA, converts source images to the visual style of the target domain, applies the adversarial idea to confuse a classifier observing the encodings, and attempts to fool a second classifier that directly observes the transformed images. In a similar fashion, Tsai et al. (2018) uses a discriminator network to encourage the output segmentations, rather than the encodings, to be similar.

Many other recent methods include an adversarial term as part of their adaptation. DIRT-T (Shu et al., 2018), in addition to the adversarial step, implements strategies to leverage the cluster assumption for classification—that decision boundaries should cross low-density regions of the marginal input distribution. Saito et al. (2017, 2018), in addition to the regular domain classifier criterion, incorporate other adversarial criteria that discourage target features to lie on regions of the source domain distribution with low support.

Having laid out the conceptual framework for this dissertation in Chapter 1, and the main related ideas and implementations from the literature in this chapter, we now delve into the details of our researched strategies and findings.

Visual correspondences and Multiple Instance Learning

CHAPTER 3

This chapter introduces an unsupervised domain adaptation strategy that exploits the presence of image regions that are structurally similar across different domains. The initial input of this method is an existing model, pretrained to perform segmentation on a reference source domain, and whose parameters are then tuned for data coming from a new target domain. In order to do this parameter tuning, we aggregate *visual correspondences*—motifs that are visually similar between different image domains—to infer the appropriate adaptation of the model parameters, via a Multiple Instance Learning formulation. In particular, we examine the annotations of an existing acquisition to determine pivot locations that characterize the reference source domain segmentation, and use a patch matching algorithm to find candidate visual correspondences coming from the new image domain. Domain adaptation happens when the model learns to assign the same class identities to source domain locations and their respective target domain visual correspondences. We validate the effectivity of this method in the problem of neuron organelle segmentation from Electron Microscopy images, where this strategy allows us to obtain high-quality segmentations on new images that are qualitatively consistent with results obtained under full supervision, but with no need for new annotations.

3.1 Visual correspondences

The method presented in this chapter performs domain adaptation by leveraging similitudes across domains found directly in the image channels. It is motivated by the fact that, despite the domain shift, key structural regions belonging to the object to segment are often readily identifiable by visual inspection, regardless of the domain from which the images come. To emulate this behavior we make use of a simple pattern matching algorithm based on Normalized Cross-Correlation. The patch matching algorithm detects regions on a target image that most resemble a selected location from the source image. We name those regions the *visual correspondences* of the source domain location.

Given a signal I —which can be either a 2D image or a 3D volume—and an image patch, or template, T , the *Zero-Normalized Cross-Correlation (NCC)* is a spatial response of the same

dimensions as the image, computed at each location \mathbf{x} , as

$$NCC(I, T)_{\mathbf{x}} = \frac{1}{n} \sum_{\mathbf{x}', \mathbf{u}'} \frac{(I_{\mathbf{x}'} - \mu_{\mathbf{x}})(T_{\mathbf{u}'} - \mu_T)}{\sigma_{I_{\mathbf{x}}} \cdot \sigma_T}, \quad (3.1)$$

where the indices \mathbf{x}' cover a neighborhood around \mathbf{x} of the same size as the template, their corresponding template locations are given by \mathbf{u}' , $\mu_{\mathbf{x}}$ is the mean intensity of the image region around \mathbf{x} of the same size as the template, μ_T is the mean intensity of the template, and $\sigma_{I_{\mathbf{x}}}$ and σ_T are the corresponding standard deviations; finally, n represents the number of locations, either pixels or voxels, contained in the template.

The NCC value, which is within the $[-1, 1]$ range, quantifies how similar every location from the image I is to the template T , by aggregating pixel correlations at every corresponding location on a sliding-window manner. As the operation is performed on patches whose intensities are locally normalized, the resulting value is relatively robust to changes in contrast and illumination between images. Image regions with large corresponding NCC values are visually similar to the template, and so finding peaks in the NCC response is a general patch matching method used in diverse computer vision applications (Lewis, 1995; Briechle and Hanebeck, 2001; Tsai and Lin, 2003; Zhao et al., 2006).

In our formulation, we expect to relate a specific location in a source domain image I^s to a number of locations from a target domain image I^t that resemble their source location. Our patch matching strategy consists in defining a neighborhood around the desired source domain location \mathbf{x} , which we denote as $I^s[[\mathbf{x}]]$, and extract such neighborhood to be used as the template patch T of Equation 3.1. Then NCC is applied between the entire target domain image and this template. The size of such neighborhood depends on the image resolution and size of the texture cues found in the images, and is therefore problem-specific.

The NCC response between a source domain location and a particular target domain location measures the quality of the match, which we expect to correlate with the agreement between class identities: the higher the NCC , the more likely it is that the target domain location and the reference source domain location share the same class.

Since possible matches could potentially appear at arbitrary orientations in the target domain image, we apply NCC on rotated versions T_{ϕ} of the source domain patch, and preserve the NCC value of the best-scoring orientation at each location.

We therefore define the visual correspondence score NCC^* as

$$NCC^*(I, T)_{\mathbf{x}} = \max_{\phi} NCC(I, T_{\phi})_{\mathbf{x}}, \quad (3.2)$$

and conversely, the best orientation ϕ^* as

$$\phi^*(I, T)_{\mathbf{x}} = \arg \max_{\phi} NCC(I, T_{\phi})_{\mathbf{x}}. \quad (3.3)$$

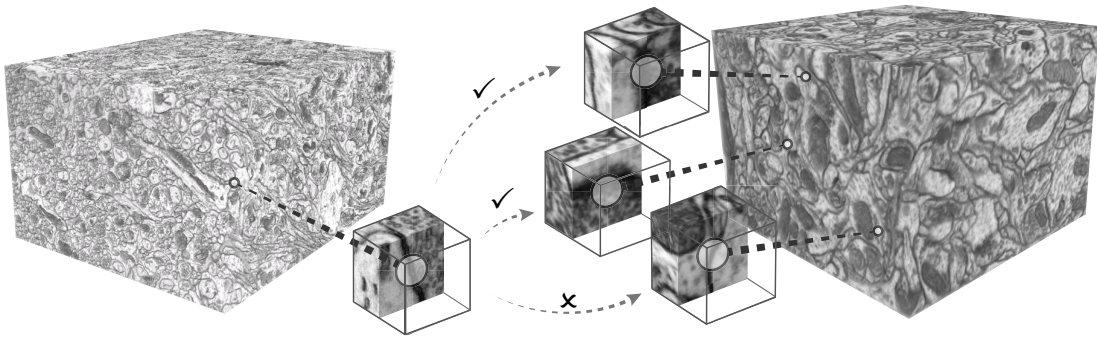


Figure 3.1 *Visual correspondences between Electron Microscopy acquisitions.* Patch matching from a location belonging to the object to segment (left)—a synapse in this example—finds similar regions in a different image (right). Despite general differences in contrast, brightness, noise, and other artifacts, most of those regions—denoted by a check mark—do correspond to synapses in the second image.

For 2D patches, we take those orientations to be n uniformly spaced rotations, $\phi_i = 2\pi i/n$, with n a hyperparameter of the method. For the 3D scenario, in order to define an evenly distributed set of orientations, we only consider orientations that are described by rays pointing to the vertices of platonic solids (Gilbert, 1828) centered at the origin, as described in Table 3.1, and as such we limit ourselves to a number of either 1, 4, 6, 8, 12, or 20 orientations.

An additional geometric consideration is that, in order to guarantee that we can extract the required neighborhood around a location under any orientation, we need to avoid locations close to the border of the image. In particular, for a D -dimensional neighborhood of side w centered at a location \mathbf{x} , we require a slack of at least $w\sqrt{D}$ units—voxels or pixels—to achieve this. This defines a margin around the image of $\lceil \frac{w}{2}\sqrt{D} \rceil$ units that we choose to exclude from the possible source domain locations to use as reference for patch matching.

By applying non-maximum suppression on the visual correspondence score NCC^* , we obtain sparse local maxima locations—specific coordinates of pixels or voxels on the target domain that score better than its neighbors. Out of these locations, those with visual correspondence score below a defined threshold are removed. The remaining high-scoring and locally isolated locations define the positions of the target domain visual correspondences for the given source domain location.

Table 3.1 *Orientations defined by the vertices of platonic solids.*

Sides	Vertices	Name	Vertex coordinates*
4	4	Tetrahedron	$(1, 1, 1), (1, -1, -1), (-1, 1, -1), (-1, -1, 1)$
6	8	Hexahedron	$(\pm 1, 0, 0), (0, \pm 1, 0), (0, 0, \pm 1)$
8	6	Octahedron	$(\pm 1, \pm 1, \pm 1)$
12	20	Dodecahedron	$(\pm 1, \pm 1, \pm 1), (0, \pm \frac{1}{\varphi}, \pm \varphi), (\pm \frac{1}{\varphi}, \pm \varphi, 0), (\pm \varphi, 0, \pm \frac{1}{\varphi})$
20	12	Icosahedron	$(0, \pm 1, \pm \varphi), (\pm 1, \pm \varphi, 0), (\pm \varphi, 0, \pm 1)$

* φ is the golden ratio $\frac{1+\sqrt{5}}{2} \approx 1.618$.

3.1.1 Sampling representative locations

Finding visual correspondences to each and every source domain location, however, would be prohibitively expensive, especially in volumetric data. Furthermore, exhaustively examining all source domain locations would result in redundancies, since locations close to one another will have almost the same correspondences. It would also result in spurious computations, since regions with scarce texture will not be semantically informative and are best left unvisited. Hence, we opt for a more principled way to define source domain locations that are useful for our task.

We devise a relevance sampling scheme that enables us to infer informative image locations, while at the same time assigning different sampling weights to different regions of the source images. Instead of randomly sampling from all possible locations from the source image, we use these weights to bias the sampling towards promising locations. These locations are defined differently for the foreground and background, or in other words, whether or not they belong to the object to segment, as discussed below. Since we are interested here in obtaining locations that characterize the object to segment, we operate mostly on the image annotations, which are what defines the segmentation masks that encode the known semantics of the source domain image.

Sampling from the foreground

Our main criterion for foreground location relevance is the context it provides. We are therefore interested in texture-rich regions near the border of the structures to segment, where we can observe from the surrounding context both the patterns that belong to the structure and the interface between the structure and the background. We also want to favor locations that are at a minimum distance from one another, so that, once we extract patches around them, such patches tend not to overlap.

To select locations that fulfill these requirements, we discretize the segmentation map by defining a grid of a fixed coarseness given by the size of the patches that we want to use for *NCC* template matching, as shown in Figure 3.2. The foreground structures are then divided into pieces that can occupy a part or the whole of a cell imposed by the grid. Since the boundaries of the annotations are often ambiguous, we define a small exclusion margin around the border of the annotated structure, which in practice is done by applying an erosion operation on the label mask. Our candidate sampling locations are the centroids of each division of the structure, after boundary removal. We assign them sampling weights that are inversely proportional to the distance between the selected locations and the nearest structure border, after accounting for the exclusion margin.

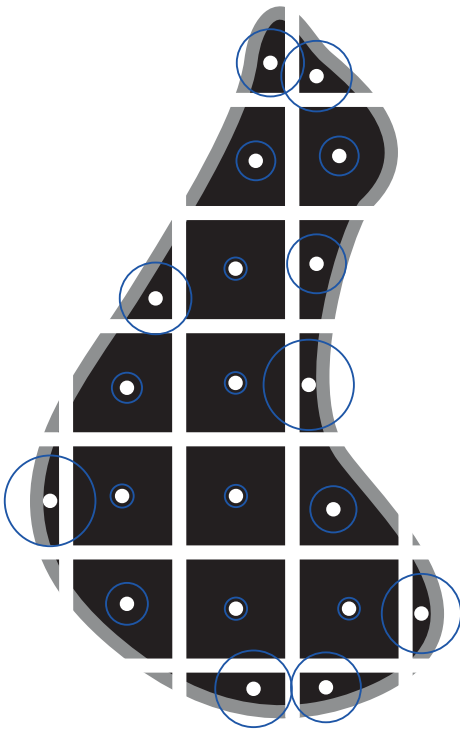


Figure 3.2 *Relevant positive locations.*

From the annotations of an object of interest, after excluding a small ignore region (outlined in gray), we select the centroids of regions imposed by a regular grid as relevant locations. Their relevance, or sampling weight, represented by the area of the blue circles, is inversely proportional to the distance to the boundaries of the original annotation. An equivalent formulation in 3D is used for volumetric annotations.

Sampling from the background

We combine two different criteria for background location relevance. First, we want to sample locations that are close to the structures to segment, so that we obtain enough evidence to tell foreground and background apart, in the same spirit as for the foreground. Second, we also want locations from areas that may be confusing or ambiguous for patch matching, that is, that may be visually similar to the foreground and therefore likely to match a structure of interest instead of the background.

Including those failure modes of patch matching in the set of selected background locations will provide the subsequent domain adaptation method with evidence to distinguish difficult cases. This is particularly important to avoid being overly confident in visual matching.

Implementing the first requirement is analogous to the foreground sampling explained above. To implement the second, we use the same patch matching strategy—namely finding candidate matches under different orientations via *NCC*—but this time between image locations labeled as foreground and background *within the same source domain image*. The goal of this process is to identify background locations that look like foreground to *NCC*. Locations known to belong to the background that score high are the potentially confusing ones and should be included in the set of relevant background locations.

3.2 Visual correspondences and domain adaptation

Applying the sampling procedure outlined in Section 3.1.1 to a source domain image yields a set of source domain locations that we regard as relevant to describe the objects to segment and the background. We aggregate both foreground and background selected source domain locations into a set of relevant locations $\mathbf{R}^s = \{\mathbf{x}_1^s, \dots, \mathbf{x}_N^s\}$ with known labels $\{y_1^s, \dots, y_N^s\}$, $y_i^s \in \{0, 1\}$. For each location \mathbf{x}_i^s , the set of K_i target domain visual correspondences $\mathbf{C}_i^t = \{\mathbf{x}_{i,1}^t, \dots, \mathbf{x}_{i,K_i}^t\}$ is determined as described in Section 3.1. In practice, however, we hold a fixed number of K best-scoring correspondences, and hence drop the subindex in the remainder of the chapter.

In an ideal scenario, where the visual correspondences—which come from the target domain image—indeed share the same class identities as their respective source domain locations, we could adjust the parameters of a model $f(\mathbf{x}; \Theta^s)$, with parameters Θ^s learned on the source domain, to a new set of parameters $\hat{\Theta}^t$ that make $f(\mathbf{x}; \hat{\Theta}^t)$ predict the assumed class identities for locations on the target domain, as

$$\hat{\Theta}^t = \operatorname{argmin}_{\Theta^t} \frac{1}{|\mathbf{R}^s|} \sum_{\mathbf{x}_i^s \in \mathbf{R}^s} \frac{1}{|\mathbf{C}_i^t|} \sum_{\mathbf{x}^t \in \mathbf{C}_i^t} \|f(\mathbf{x}^t; \Theta^t) - y_i^s\|. \quad (3.4)$$

In other words, we would assume that we have a labeled set in the target domain, and would perform supervised training.

3.2.1 Multiple Instance Learning formulation

Evidently, the situation described above is unrealistic, as it requires that the patch matching algorithm always yield target domain locations that correspond in class identity to their source domain locations. One way to mitigate this problem is to select only a reduced number of best correspondences, according to their matching score. This still, however, does not guarantee that the correspondences be correct. We overcome this uncertainty in the class identity of the matches by using a *Multiple Instance Learning* (MIL) approach.

The key idea behind MIL is to optimize one single value coming from a set of examples that are semantically related among themselves. MIL is a natural formulation for problems where different versions of objects exist, individual labeling of each version is costly, and a single response per object is sufficient (Carbonneau et al., 2018).

A classical example, coming from computational chemistry, is the problem of deciding if a particular molecule is a good drug candidate, given several potential molecule configurations and their corresponding measured effects (Dietterich et al., 1997). In this scenario, if any one of the configurations produces desirable effects, the whole set of configurations—known as a *bag* in the MIL formalism—is labeled as positive, meaning that the molecule becomes a candidate for further research. If none of the configurations does, the bag is labeled as negative and the molecule is dropped from the study. In other words, the learning process acts at the bag level instead of on individual instances and only one response per bag is needed.

3.3. Application: Boosted decision trees for Electron Microscopy

We adapt this idea to our visual correspondence problem, by collecting all correspondences to different source domain locations into bags. Correspondences to a location labeled as foreground are placed in a bag labeled as positive, as we expect some of the correspondences to belong to the object in the target domain image. By contrast, correspondences to locations labeled as background are placed in a negative bag that we expect not to contain the object of interest.

In this way, we adapt Equation 3.4 for MIL, as

$$\hat{\Theta}^t = \operatorname{argmin}_{\Theta^t} \frac{1}{|\mathbf{R}^s|} \sum_{\mathbf{x}_i^s \in \mathbf{R}^s} \operatorname{softmin}[\ell_{i,1}, \dots, \ell_{i,K}], \quad (3.5)$$

where

$$\ell_{i,j} = L\left(f(\mathbf{x}_{i,j}^t; \Theta^t), y_i^s\right) \quad (3.6)$$

is a classification loss function. The softmin function of Equation 3.5 is where MIL aggregates the responses coming from the different elements in the bags into a single value. Our softmin is an adaptation of the log-mean-exponential function (Pinheiro and Collobert, 2015), of the form

$$\operatorname{softmin}[x_1, \dots, x_N] = -\frac{1}{r} \ln \frac{1}{N} \sum_{i=1}^N \exp(-r x_i), \quad (3.7)$$

where r is a parameter, adjusted via grid search, that controls the behavior of our softmin: increasing values of r will bring the softmin function closer to a real minimum function.

3.3 Application: Boosted decision trees for Electron Microscopy

The above sections are formulated for the general case of an arbitrary source domain model. As a particular instantiation of our method, we enable *boosted decision trees* (BDT) for unsupervised domain adaptation, in the context of semantic segmentation of organelles on Electron Microscopy (EM) 3D images.

BDTs are ensemble models that use decision trees as its *weak learners*, each of which is learned via gradient boosting (Dai et al., 2007). In its general form, the ensemble is given by

$$f(\mathbf{x}) = \sum_{d=1}^{N_d} \alpha_d f_d(\mathbf{x}). \quad (3.8)$$

In other words, a number of N_d weak learners are trained for inference one at a time, and both the relative weights α_d and the parameters of the weak learner f_d are learned at the same time via the gradient boosting algorithm (Dai et al., 2007).

BDTs have proven well suited for segmentation of EM images (McDonald and Sheehan, 2004;

Chapter 3. Visual correspondences and Multiple Instance Learning

Becker et al., 2012; Becker, 2016; Oguz et al., 2017). One of the most successful implementations for EM segmentation is that of Becker et al. (2012), which uses decision stumps—decision trees of depth 1 each—as weak learners, in conjunction with *context cues*. Context cues are features extracted at each location of the image, that describe such location in terms of different measurements obtained from its surrounding context. In particular, the context cue algorithm learns to identify “boxes” of specific sizes, and at a specific offset with respect to the location to describe—or more precisely, with respect to its computed normal plane—and integrates the values of spatial filters, such as intensity, gradient magnitude, and laplacians, inside those boxes. Context cues attempt to compensate for common challenges in EM segmentation, such as lack of quality, low image resolution, presence of artifacts, and arbitrary tissue orientation, by considering not only the pixels or voxels within the object to segment, but also other image characteristics that might occur frequently in the object’s surroundings.

Following the formulation of Becker et al. (2012), given a set of context cues $\{\hat{x}_1, \dots, \hat{x}_C\}$, describing each image location \mathbf{x} , the BDT model based on decision stumps, for a single image domain, is given by

$$f(\mathbf{x}; \Theta) = \sum_{d=1}^{N_d} \alpha_d \text{sign}(\hat{x}_{c_d} - \tau_d), \quad (3.9)$$

that is to say, the output value of each weak learner at a given location \mathbf{x} corresponds to whether or not some single context cue \hat{x}_{c_d} , chosen by boosting, is above some learned value τ_d .

In order to enable the BDT model for domain adaptation with our MIL approach, we update its model parameters for the target domain. The relative weights of the weak learners α_d measure the importance of each decision to the whole model, and are directly related to how informative a particular context cue is to the problem at hand. As we expect and require that the same spatial description of a location be preserved across domains, we preserve those model parameters across image domains. The remaining parameters, namely the thresholds τ_d^s to apply to each context cue, are the ones to undergo adaptation via MIL and visual correspondences.

Our goal then is to learn a new set of thresholds $\hat{\Theta}^t = \{\tau_1^t, \dots, \tau_{N_d}^t\}$ for the target domain. Following the MIL idea of Section 3.2.1, we express the learning problem as

$$\hat{\Theta}^t = \underset{\tau_1^t, \dots, \tau_{N_d}^t}{\text{argmin}} \sum_{\mathbf{x}_i^s \in \mathbb{R}^3} \text{softmin}[\ell_{i,1}, \dots, \ell_{i,K}], \quad (3.10)$$

where $\ell_{i,j} = L(f(\mathbf{x}_{i,j}^t; \Theta^t), y_i^s)$, the model f is that of Equation 3.9, and the loss function L is the Huber loss,

$$L_\delta(y, \hat{y}) = \begin{cases} \frac{1}{2}(y - \hat{y})^2 & \text{if } |y - \hat{y}| \leq \delta \\ \delta|y - \hat{y}| - \frac{1}{2}\delta^2 & \text{otherwise} \end{cases}. \quad (3.11)$$

3.3.1 Threshold learning regularization

We learn the new thresholds one at a time via *gradient boosting* (Friedman, 2002). In addition, as the correspondences provide only a limited number of examples for training, we regularize the training by using the original source domain thresholds τ_d^s as priors for the adjusted target domain thresholds τ_d^t , so as to prevent overfitting.

In order to do so, we estimate probability distributions for both the source and target domain thresholds, by assuming that they follow normal distributions, $\tau_d^{(\cdot)} \sim N(\mu_{\tau_d^{(\cdot)}}, (\sigma_{\tau_d^{(\cdot)}})^2)$, and by estimating their distribution parameters by bootstrap resampling (Efron, 1982). For the source domain, we learn multiple values for each τ_d^s from random subsamples of the training data, and then estimate the mean and variance from these values. Similarly, for the target domain, we randomly sample subsets of the source-target matches, and minimize Equation 3.10 for each subset. From these multiple estimates of τ_d^t we can compute the required means and variances.

Finally, we take as the learned target domain parameters $\tau_d^t = \arg \max_{\tau} p(\tau_d^s = \tau) p(\tau_d^t = \tau)$. In this way, $p(\tau_d^s)$ acts as a prior over the target domain thresholds: if the target domain correspondences produce high variance estimates, the distribution learned in the source domain acts as a regularizer.

3.3.2 Datasets

We evaluate our method on the problem of mitochondria and synapse segmentation on Electron Microscopy volumetric images. In particular, the volumes we use are Focussed Ion Beam Scanning Electron Microscope (FIB-SEM) acquisitions, which were each obtained from brain tissue coming from different mouse specimens.


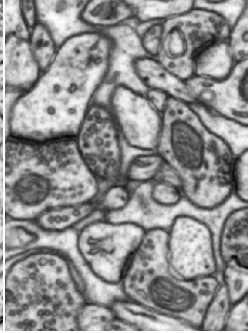
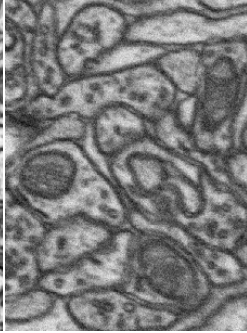
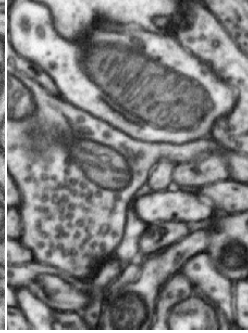
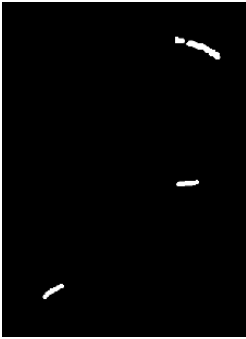
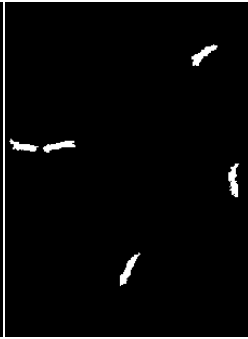
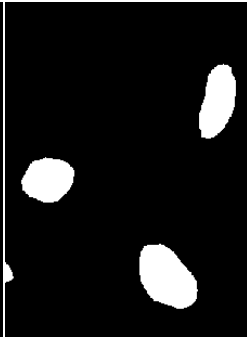

All image volumes consist of a series of parallel cross-sections (slices) imaged at an isotropic resolution. Details on the images and example patches extracted from them are shown in Table 3.2.

As training annotations, manually-produced segmentation maps are provided. Those segmentation maps are handcrafted in a slice-by-slice fashion, and provide a binary voxel-wise labeling at every location as either the voxel belonging to the object to segment or not.

3.3.3 Evaluation metric

The adapted model computes a prediction of the class identities at every target domain location \mathbf{x}^t , as $\hat{y}^t = f^t(\mathbf{x}^t)$. In order to evaluate the performance of our method, we measure the aggregated agreement between corresponding expected and predicted annotations, y^t and \hat{y}^t ,

Table 3.2 Electron Microscopy datasets description.

Dataset	Synapses		Mitochondria	
	Hippocampus	Striatum	Cerebellum	Somatosensory cortex
Resolution	5 nm (isotropic)	5 nm (isotropic)	6.2 nm (isotropic)	6.2 nm (isotropic)
Dimensions	1027×987×219	750×564×750	1024×653×165	853×506×496
Image (Example region of size 300×400)				
Annotations				

in terms of their *Jaccard index*, also known as Intersection over Union, expressed as

$$J(Y^t, \hat{Y}^t) = \frac{\sum_{y^t \in Y^t} \mathbb{1}[y^t = 1 \wedge \hat{y}^t = 1]}{\sum_{y^t \in Y^t} \mathbb{1}[y^t = 1 \vee \hat{y}^t = 1]}. \quad (3.12)$$

To avoid overpenalizing at the locations that are close to the boundary of the objects to segment, for which the annotations are subject to high uncertainty, we consider an ignore zone around the object boundaries. We obtained the appropriate value of 4 voxels for this parameter via grid search. The locations within this region are not considered in the evaluation.

3.3.4 Experimental setup

We extract visual correspondences for patches of $35 \times 35 \times 35$ voxels, centered around each reference source domain location, as explained in Section 3.1. Such patch dimensions, which are problem and resolution dependent, should capture enough contextual information for the visual correspondences process to yield suitable correspondence candidates, as shown in Figure 3.3. For both the synapse and mitochondria segmentation problems, we use the relevance sampling scheme of Section 3.1.1 to obtain a set of 300 relevant locations extracted from the structure of interest, and 600 extracted from the background. We threshold the *NCC* scores

3.3. Application: Boosted decision trees for Electron Microscopy

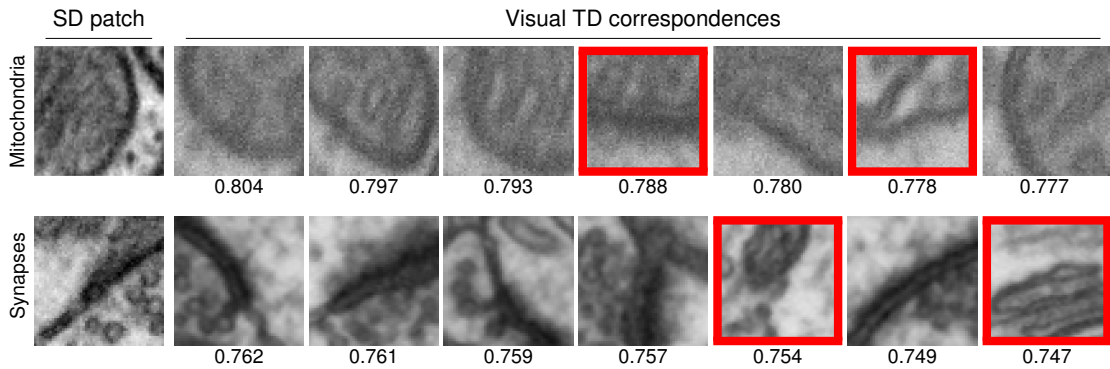


Figure 3.3 *Example visual correspondences for Electron Microscopy datasets. Central slices of visual correspondences across two different domains, for the mitochondria (top) and synapse (bottom) EM datasets, and their corresponding NCC scores. High-scoring matches that do not conclusively correspond to the source domain object are highlighted in red.*

at 0.2 to eliminate low-quality candidate visual correspondences. Such value is determined by manually removing spurious correspondences and inspecting at what threshold they stop occurring.

3.3.5 Baselines

We evaluate our method against the following related strategies:

No adaptation. We use the model trained on the source domain directly for prediction on the target domain, to show the need for domain adaptation.

Histogram matching. We change the gray levels in the target stack prior to feature extraction to match the distribution of intensity values in the source domain. We apply the classifier trained on the source domain on the modified target stack, to rule out that a simple transformation of the images would suffice.

Best TD match. For each source example, we assume that the best match found by *NCC* is a true correspondence, which we annotate with the same label. A classifier is trained on these labeled target examples.

Subspace alignment (SA). We test the method of Fernando et al. (2013)—one of the very few state-of-the-art domain adaptation approaches directly applicable to our problem. It first aligns the source and target PCA subspaces and then trains a linear SVM classifier. We also tested a variant that uses an AdaBoost classifier on the transformed source data to check if introducing non-linearity helps.

3.3.6 Results

As quantitative evaluation, we report the obtained Jaccard indices (Equation 3.12) for all the methods in Table 3.3.

Effect of the number of correspondences

In order to understand how sensitive adaptation is to the number of potential correspondences, we apply our method under different values of K , and present a summary of the results in Figure 3.4. It shows that our method is robust to a wide range of K , with the best performance obtained for K between 3 and 15. This confirms the importance of using MIL over simply choosing the highest ranked correspondence. However, too large a K is detrimental, since the ratio of right to wrong candidates then becomes lower. From these results we decide on a fixed value of $K = 8$ for all our experiments.

Comparison against related methods

Table 3.3 offers a quantitative comparison of our approach to the baselines mentioned in Section 3.3.5. Note that we significantly outperform them for both datasets studied. We conjecture that the inferior performance of SA (Fernando et al., 2013) is because our features are highly correlated, making PCA a suboptimal representation to align both domains. The training time for the baselines was around 30 minutes each. Our method takes around 35 minutes for training. Finding correspondences for 10000 locations takes around 24 hours when parallelized over 10 cores, which corresponds to around 81 seconds per source domain patch. While our approach takes longer overall, it yields significant performance improvement with no need for user supervision. All the experiments were carried out on a 20-core Intel Xeon 2.8GHz.

In Figure 3.5 we provide qualitative results by overlaying on a single target domain slice segmentation results with and without our domain adaptation. For mitochondria segmentation, our method predicts objects that are structurally similar to the expected ones. For synapse segmentation, our approach both successfully removes spurious detections and correctly predicts the existence of small structures that would have been missed otherwise.

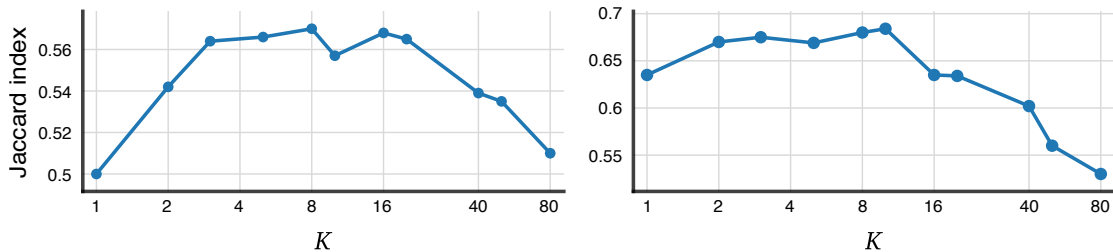


Figure 3.4 Jaccard indices for different numbers of candidate correspondences K , for the synapse (left) and mitochondria (right) datasets. Using either a very small or a very large number of candidate correspondences causes a decrease in predictive power, due to overconfidence of the matching algorithm, or confusion of the Multiple Instance Learning optimization.

Table 3.3 Jaccard indices for our method and the baselines of Section 3.3.5.

	No adaptation	Histogram matching	TD only	SA (Fernando et al., 2013) + Linear SVM + AdaBoost		Ours
Synapses	0.22	0.32	0.39	0.13	0.39	0.57
Mitochondria	0.50	0.39	0.57	0.24	0.59	0.62

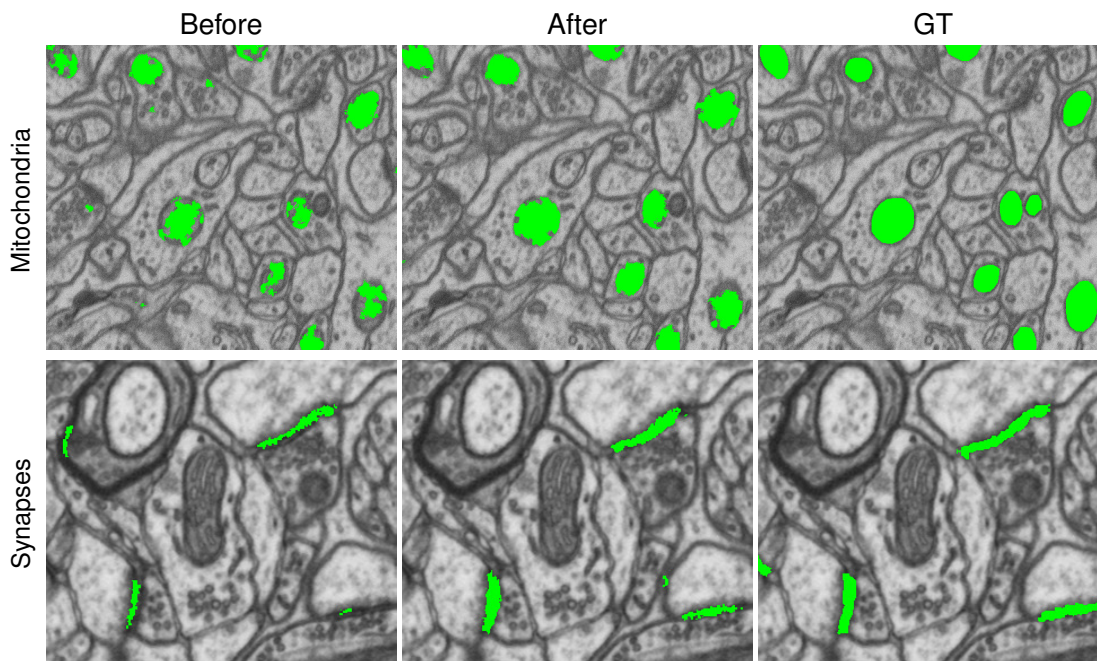


Figure 3.5 Example segmentations obtained with our method for the synapses and mitochondria Electron Microscopy datasets, overlaid on one slice of the target domain stacks. In both cases, we display from left to right the results obtained without domain adaptation, with domain adaptation, and the ground truth.

3.4 Conclusion

This chapter introduced an unsupervised domain adaptation method based on the automated discovery and exploitation of interdomain visual correspondences. One-to-many correspondences between domains are handled by a Multiple Instance Learning formulation that adjusts model parameters for a target dataset, circumventing the need for explicit manual annotations. Results provided on unsupervised segmentation of neural organelles demonstrate the effectiveness of this method.

One limitation of this approach is that the computation of visual correspondences, performed independently for each of the selected source domain locations, finds only a small, *sparse* set of locations on the target domain as their correspondences, which is used to guide the parameter

Chapter 3. Visual correspondences and Multiple Instance Learning

adaptation. In the next chapter, we will explore how to incorporate structural information in a way that is more consistent with the structure of the source domain annotations, in order to obtain a larger set of training examples.

In addition, this method is based on adapting the parameters of simple boosted decision trees models. Adapting larger, more expressive models, such as deep neural networks, is likely to result in overfitting, due to the increased number of parameters and few examples to tune them. More aggressive regularization strategies to constrain the adaptation are proposed in the next chapter in order to alleviate this.

Consensus heatmaps and the two-stream U-Net

CHAPTER 4

This chapter improves on the idea of visual correspondences, presented in Chapter 3, to build a better-informed set of interdomain correspondences, and introduces a deep-learning framework that enables cooperation between models acting on different domains. In particular, the method presented here aggregates candidate correspondences by a voting scheme and uses them to construct a *consensus heatmap*—a map of how frequently locations on a new image are matched to relevant locations from the original one. High-scoring regions of the heatmap are highly correlated to regions where the object to segment appears, and are thus used as soft labels to be incorporated in deep domain adaptation pipelines. We introduce one such pipeline—the *two-stream U-Net*—and provide empirical evidence that orchestrating both ideas results in a high-quality unsupervised segmentation strategy.

4.1 Consensus heatmaps

One of the main assumptions when using visual correspondences, as introduced in Chapter 3, is that pattern matching via *NCC* is a sensible way to establish correspondences. In particular, we assume that visual correspondences found by *NCC* should induce semantic correspondences, and hence class identities between specific source and target domain locations could be also expected to correspond, at least in general. The Multiple Instance Learning formulation detailed in Section 3.2.1 is an attempt to relax this dependence. A second assumption is that the information encoded in those correspondences can be used to adapt model parameters so that they are effective for segmentation on the target domain. However, modern machine learning techniques, for example those based on deep learning, involve a staggering number of parameters, and information coming from visual correspondences—which is scarce and sparse—does not provide sufficient constraints for the parameter adaptation problem to be solved in an effective way in such scenarios.

A better approach consists in collecting evidence coming from all the visual correspondences at once. This can be done by keeping track of, for each target domain location, how many times and with what certainty it has been found as a visual correspondence for different source

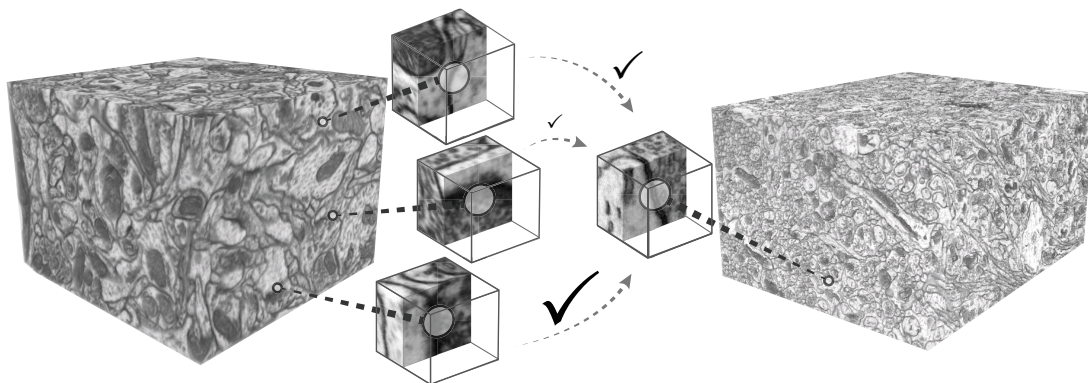


Figure 4.1 Aggregating visual correspondences. Different source domain locations (left) choosing the same target domain location (right) as their candidate visual correspondence provides a strong evidence that such target domain location belong to the object to segment. The quality of the patch matching score, hinted here by the size of the checkmarks, allows for a weighted vote scheme that considers how confident the patch matching was.

domain locations. The resulting tally, known as the *consensus heatmap*, highlights popular regions commonly matched to locations belonging to the structure of interest in the source domain.

In other words, as illustrated in Figure 4.1, if the original visual correspondences formulation establishes a one-to-many relationship between source and target domain locations—one location from the source domain points to many target domain correspondences—the consensus heatmap establishes a *many-to-one* relationship instead. This is more desirable because, since the class identities of source domain locations are known, having many source domain locations agreeing to one particular target domain location is a stronger evidence that its class is shared with that of the source locations.

Consensus heatmaps are motivated by the following two considerations:

1. The visual correspondences of different source domain locations labeled as foreground tend to be high-scoring locations that are often found close to one another in the target domain. The equivalent applies to locations labeled as background. Therefore, if we consider all visual correspondences for all sampled source domain locations at once, we can expect to observe clusters of high-scoring target domain locations. In other words, different source domain locations will often be assigned the same or neighboring regions as their visual correspondences. Therefore, if a particular region of the target domain image is selected by many source domain samples of the same class—either foreground or background—it is likely to correspond to the same class as well.
2. The patch matching used to establish visual correspondences considers source domain patches under different orientations and chooses the best. When a source domain region matches a target domain region under a specific orientation, it is reasonable to expect that their segmentations should also match, under the same orientation. Therefore,

the local segmentations from the source domain, at the given location and under such orientation, should give us an idea of the expected local segmentations on the target domain at the location of the visual correspondence.

These two observations taken together suggest that we can aggregate the information provided by all visual correspondences into one single response—the consensus heatmap—that unifies them, considers structure found in the annotations, and improves reliability. We now describe the heatmap generation algorithm in more detail and then how to use the heatmaps to train models.

4.1.1 From visual correspondences to heatmaps

A consensus heatmap is essentially a numerical score at each target domain location, that quantifies our belief that such location belong to either the structure of interest or the background, given the aggregated evidence coming from visual correspondences. Each visual correspondence between domains contributes to the scoring at the particular location and its neighborhood, according to the quality of its *NCC* match. This can be understood as a weighted voting scheme—aggregating the evidence makes regions consistently voted as being of a particular class stand out. Conversely, it tends to discount locations where the evidence is inconclusive, either because of their lack of support due to few or low-quality matches, or because of contradictory votes coming from both foreground and background regions.

Building the consensus heatmap

Building the consensus heatmap entails the following steps:

1. Initialize the heatmap as a response of the same dimensions as the target domain image, with zeroes everywhere.
2. Sample source domain locations as outlined in Section 3.1.1.
3. Find all visual correspondences to those locations as explained in Section 3.2.
4. For all visual correspondences, increase or decrease the consensus heatmap at and around the location of the match, according to the class at the source domain location.
5. Discard values at locations that have not being visited often enough, or with small absolute value.

A detailed pseudocode of the algorithm is presented in Algorithm 1. Visualizations of the process at each step are shown in Figure 4.2.

We have investigated two strategies to assign consensus score values to target domain locations. Both rely on placing values at and around locations that produce visual correspondences, according to the local source domain annotation, guided by the local labels under the orientation

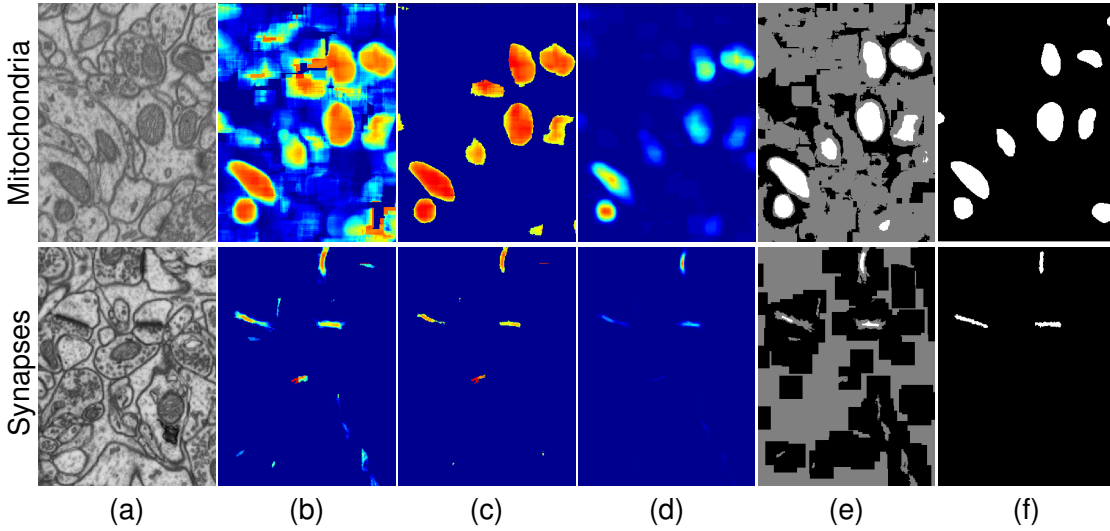


Figure 4.2 The responses aggregated from all correspondences for a target domain image (a) are used to build a heatmap (b). A threshold is then applied to discard the regions that have low aggregated NCC (c). Different regions of the heatmap can aggregate responses from a different number of source domain locations (d). We assign labels (e) only for the regions that are supported by a minimum number of votes. This labeling largely overlaps with the actual labels (f).

that produces the best match. We now introduce them and discuss how we combine them to generate the heatmaps.

Mixture of Gaussians. We assign values indicating that a given target domain location \mathbf{x} belong to the structure of interest as a mixture of Gaussian responses, as

$$\mathbf{H}_G(X^t; \mathbf{R}^s)_\mathbf{x} = \sum_{\mathbf{x}_i^s \in \mathbf{R}^s} \sum_{\mathbf{x}_j^t \in C_i^t} \mathbf{S}_{\mathbf{x}_j^t} \cdot \mathcal{N}(\mathbf{x} - \mathbf{x}_j^t, \Sigma_{i,j}), \quad (4.1)$$

i.e., we place Gaussian density centered around the target domain locations \mathbf{x}_j^t that were identified as visual correspondences to the source domain reference voxels \mathbf{x}_i^s , with a weight $\mathbf{S}_{\mathbf{x}_j^t}$ given by the quality of the match, that is, $\mathbf{S} = \text{NCC}^*(X^t, X^s[[\mathbf{x}_i^s]])$. The parameters of the covariances $\Sigma_{i,j}$ are estimated so as to approximate the shape of the pixels or voxels labeled as positive in the neighborhood of the source domain location and under the orientation that produced the match.

Label overlay. Analogously, we explore using a mixture model that exploits the label structure in a more direct way; instead of using the structure of the annotations only to estimate parameters, we use it as a mask to determine scoring locations.

Here, the value of the heatmap obtained from the set of visual correspondences for each target

Algorithm 1: Consensus heatmap construction

input : A source domain image X^s
 Source domain annotations Y^s
 A target domain image X^t of dimensions $w \times h \times d$
 A set of relevant source domain locations \mathbf{R}^s
 A threshold controlling the minimum background support τ_{low}
 A threshold controlling the minimum foreground support τ_{high}
 A minimum match count $\tau_{\mathbf{C}}$

Output: The consensus heatmap \mathbf{H}

```

1  $\mathbf{H} \leftarrow \text{zeros}(w \times h \times d)$  // The consensus heatmap
2  $\mathbf{C} \leftarrow \text{zeros}(w \times h \times d)$  // Match count at each location
3 for  $\mathbf{x}_i^s \in \mathbf{R}^s$  do // selected source domain locations
4    $\blacksquare \leftarrow X^s[\mathbf{x}_i^s]$  // extract patch from  $X^s$  around  $\mathbf{x}_i^s$ 
5    $\square \leftarrow Y^s[\mathbf{x}_i^s]$  // extract patch from  $Y^s$  around  $\mathbf{x}_i^s$ 
6    $\mathbf{S} \leftarrow \text{NCC}^*(X^t, \blacksquare)$  // Best NCC score, Equation 3.2
7    $\Phi \leftarrow \phi^*(X^t, \blacksquare)$  // Best orientation, Equation 3.3
8   for  $\mathbf{x}^t \in \mathbf{C}_i^t$  do // locations of visual correspondences
9      $\diamond \leftarrow \text{rotate}(\square, \Phi_{\mathbf{x}^t})$  // reorient local labels to orientation of the match
10     $\mathbf{C} \leftarrow \text{add\_at}(\mathbf{C}, \diamond, \mathbf{x}^t)$  // increase count at and around location of the match
11     $\mathbf{H} \leftarrow \text{add\_at}(\mathbf{H}, \mathbf{S}_{\mathbf{x}^t} \cdot \diamond, \mathbf{x}^t)$  // place rotated labels, scaled by their match score
12  $\mathbf{H}[\mathbf{H} > \tau_{\text{low}} \ \& \ \mathbf{H} < \tau_{\text{high}}] \leftarrow 0$  // discard values at locations with low aggregated NCC support
13  $\mathbf{H}[\mathbf{C} < \tau_{\mathbf{C}}] \leftarrow 0$  // discard values at locations with few votes
14 return  $\mathbf{H}$ 

```

domain location \mathbf{x} is given by

$$\mathbf{H}_L(X^t; \mathbf{R}^s)_{\mathbf{x}} = \sum_{\mathbf{x}_i^s \in \mathbf{R}^s} \sum_{\mathbf{x}_j^t \in \mathbf{C}_i^t} \mathbf{S}_{\mathbf{x}_j^t} \cdot \mathbb{1}[\mathbf{x} \in \mathbf{o}_{i,j}], \quad (4.2)$$

where $\mathbf{o}_{i,j}$ is a binary mask obtained by extracting the source domain labels at and around \mathbf{x}_i^s , rotating them to the orientation of the match, and overlaying them at the location of the match. In other words, we place a uniform response around the matching locations according to $\mathbf{o}_{i,j}$, and scaled to the quality of the match $\mathbf{S}_{\mathbf{x}_j^t}$. Figure 4.4 depicts the construction of this mask. Successively overlaying the local labels of the regions that produced the match with this procedure highlights the popular matching neighborhoods in a way that preserves the original structure of the object of interest. Since the sampling protocol of Section 3.1.1 favors patches near the border of the target object, the corresponding labels will often include details describing the outline of the object, which we expect to be similar across domains.

Combining the models

The above two models serve different purposes. The mixture of Gaussians is suitable for general modeling with limited regards to the structure of the annotations; the Label-driven model is

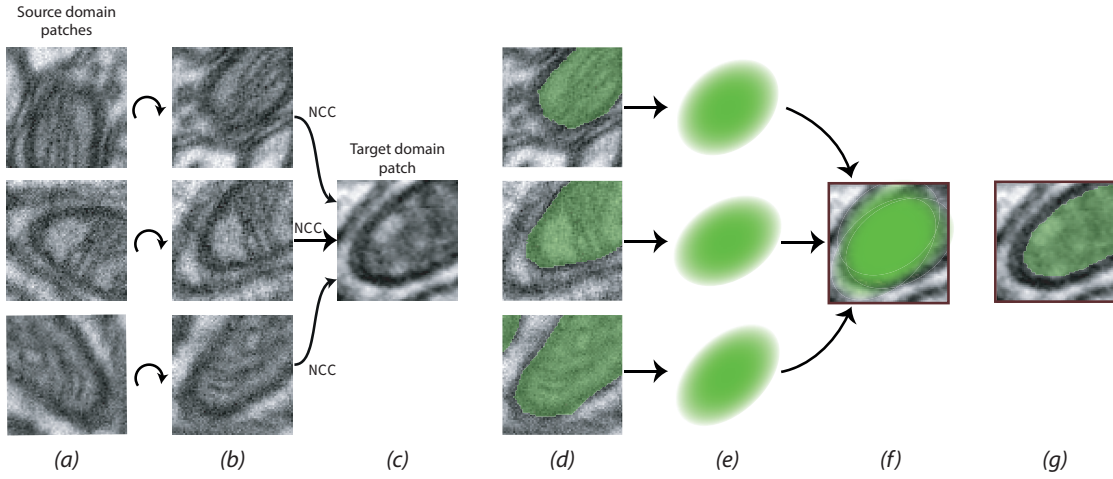


Figure 4.3 Heatmap construction by mixture of Gaussians. Source domain patches (a), after reorientation (b), find a common target domain location (c) as their visual correspondence. Their corresponding annotations, under the same orientation (d), are used to infer the shapes of Gaussian responses (e) that, when aggregated, will approximate the expected target annotations (f).

strongly bound to the structure of the labels, and disregards responses that do not comply with it. As visual correspondences can be obtained for both foreground and background, and since the annotations provide structural information of the foreground, we use the first scoring scheme to assign values to matches with the background, and the second one for matches with the structure of interest. We then combine both formulations into our consensus heatmap,

$$\mathbf{H}(X^t; \mathbf{R}^s) = \mathbf{H}_L(X^t; \mathbf{R}_{(+)}^s) - \mathbf{H}_G(X^t; \mathbf{R}_{(-)}^s), \quad (4.3)$$

where we have separated the selected source domain locations into the sets containing locations labeled as foreground $\mathbf{R}_{(+)}^s$ and background $\mathbf{R}_{(-)}^s$. Large positive or negative values therefore indicate how certain we are that a voxel at a given location belong to the foreground or background, respectively.

4.1.2 Refining correspondences with heatmaps

As they aggregate information from many correspondences, we expect our heatmaps to be more dependable than individual correspondences. We can replace the raw *NCC* scores attached to the correspondences by their heatmap values and use these values to rank them. This is beneficial for removing spurious correspondences that score high in terms of their *NCC* but whose classes do not coincide with their reference source domain locations. A qualitative evaluation of using consensus heatmaps instead of the raw *NCC* values for ranking correspondences is provided in Figure 4.5.

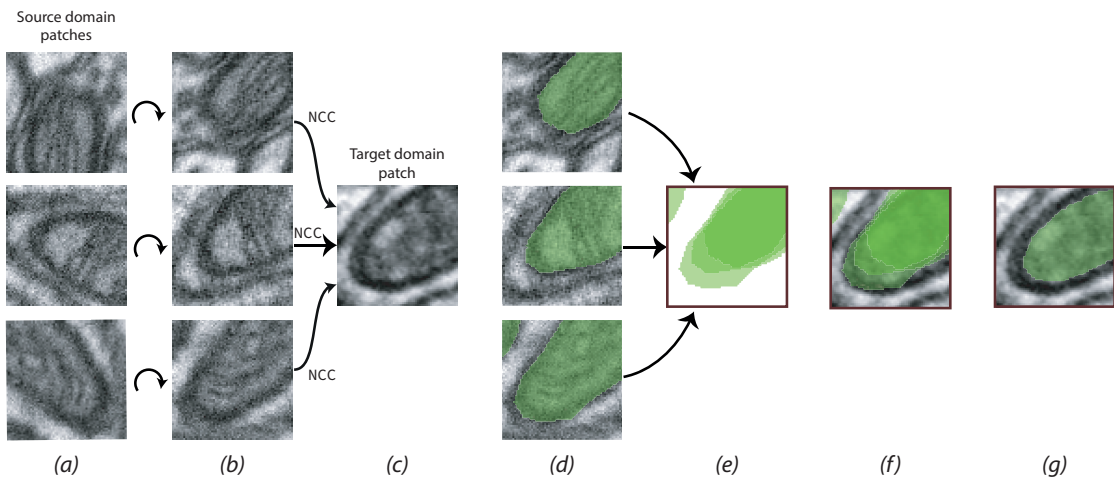


Figure 4.4 Label-driven heatmap construction. Source domain patches (a), after reorientation (b), find a common target domain location (c) as their visual correspondence. Their corresponding annotations, under the same orientation (d), are aggregated into a mask (e) that will approximate the expected target annotations (f).

4.1.3 Heatmaps as soft annotations

As high-confidence regions from the consensus maps largely coincide with the expected labels on the target domain, we select those locations by establishing a lower and an upper threshold on the heatmaps. We label as background the regions that fall below the lower threshold τ_{low} , and as foreground the regions that score above the upper threshold τ_{high} . From those potential regions, we accept as true annotations only the voxels that aggregate responses from at least a minimum number of source domain locations τ_{C} . Locations that have not been voted by at least that number of source domain locations, as well as those whose heatmap values are between the rejecting thresholds, are regarded as unannotated.

The protocol above already gives us a partially annotated set of target domain locations where we could train a predictor. Doing so, however, would most likely lead to model overfitting. Instead we propose to use those partial annotations, which are obtained in an unsupervised fashion, as input to an arbitrary supervised or semisupervised domain adaptation method, to turn it into an unsupervised one. We introduce one such model next.

4.2 The two-stream U-Net

The U-Net, introduced in Ronneberger et al. (2015), is a deep convolutional network architecture originally created for biomedical image segmentation, but which has proven very effective for a wide range of applications beyond its original domain (Mosińska et al., 2018; Wang et al., 2019; He et al., 2019). The architecture, outlined in Figure 4.6, comprises a succession of convolutional and pooling layers that enables it to encode the image at multiple levels of granularity, upscaling layers that apply the inverse spatial operations—so as to recover a segmentation

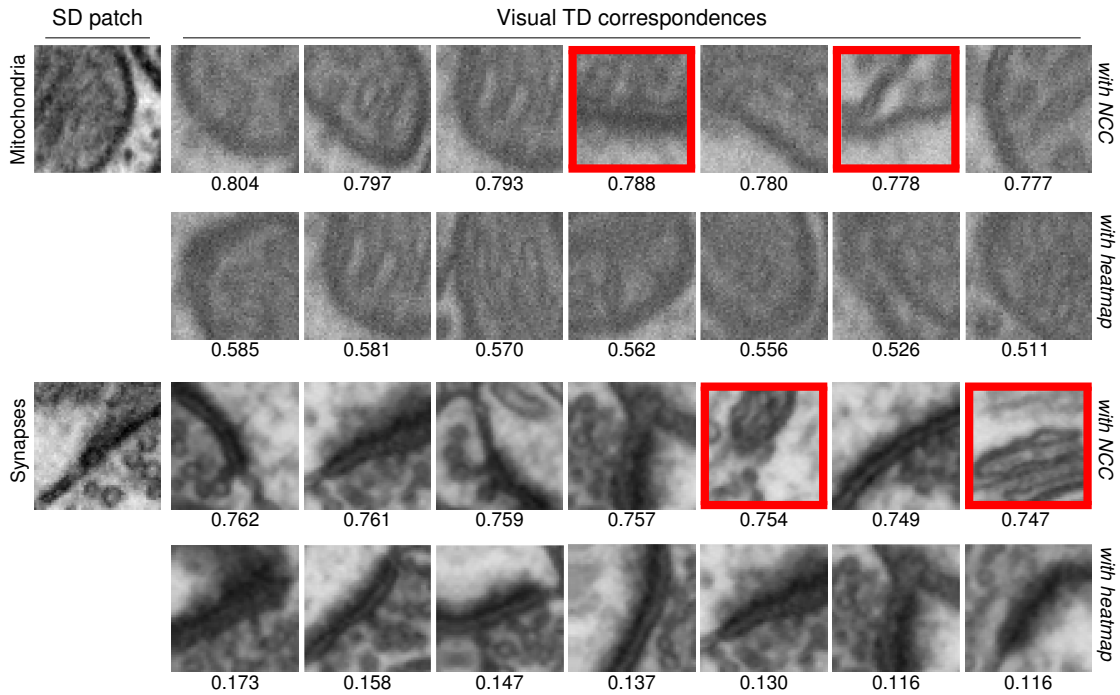


Figure 4.5 Refined correspondences via consensus heatmap.

Central slices of visual correspondences across two different domains, for the mitochondria (top) and synapse (bottom) datasets. For each dataset, the top row contains the best scoring locations, in terms of their NCC score, and the bottom row contains the best scoring locations in terms of their consensus heatmap score. Scoring values are shown under each image. The occurrence of high-scoring matches that do not conclusively correspond to the source domain object, highlighted in red, is largely avoided by using the consensus heatmap instead of NCC as the ranking criterion.

map of dimensions similar to the original—and skip connections that allow for combinations of features across encoding and decoding layers.

Albeit powerful, out-of-the-box the U-Net suffers from the domain shift problem, and cannot successfully segment images with different characteristics as those used for training. We propose a strategy to overcome this limitation, based on co-regularization and weight sharing between two U-Nets acting concurrently on different image domains.

4.2.1 General structure

Our proposed model, depicted in Figure 4.7, consists of two *streams*—two independent U-Nets $f^s(\cdot; \Theta^s)$ and $f^t(\cdot; \Theta^t)$ with their own sets of parameters Θ^s and Θ^t , each specialized for each image domain—with identical structure. As the structure is the same for both streams, all weights from the source domain stream have a corresponding weight on the target domain one. During training, we feed images to each stream coming from its associated domain only. Each stream outputs the predicted segmentation map for its domain, as a spatial response that contains, for each location, the probability that the pixel or voxel at that location belong

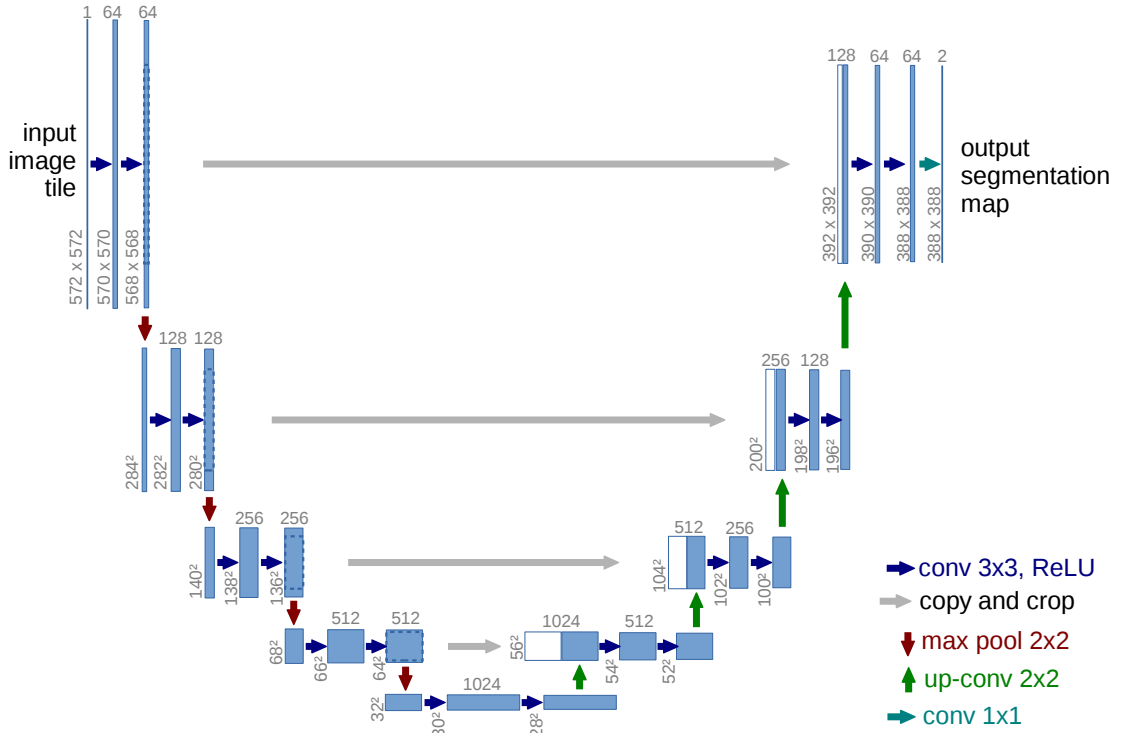


Figure 4.6 Original U-Net (Ronneberger et al., 2015)

to the background or foreground. Those responses may be subsequently binarized to obtain the final segmentation.

4.2.2 Parameter regularization and sharing

Our first strategy to relate both streams is to link weights coming from corresponding layers. For a given pair of corresponding source-target weights, we can decide either to learn a single value that is to be shared between the two streams, to constrain one of the parameters so that it does not deviate arbitrarily from the other, or to not relate them at all and allow them to vary freely during training. The first and the third options are straightforward to implement, as they represent keeping two references pointing to the same vector of weights during training, and keeping two disjoint vectors of parameters with no further consideration, respectively. For the second case, we build on the weight regularization of Rozantsev et al. (2019), where the target domain weights are enforced to follow a linear relationship with those of the source domain.

We define a set of layers Ω from the source domain stream where we want to apply weight regularization. On all such layers f_j^s , $j \in \Omega$, parametrized by weights θ_j^s , and their corresponding target domain layers, parametrized by θ_j^t , we define the weight regularization term r_w as

$$r_w = \frac{1}{|\Omega|} \sum_{j \in \Omega} \|a_j \theta_j^s + b_j - \theta_j^t\|_2^2, \quad (4.4)$$

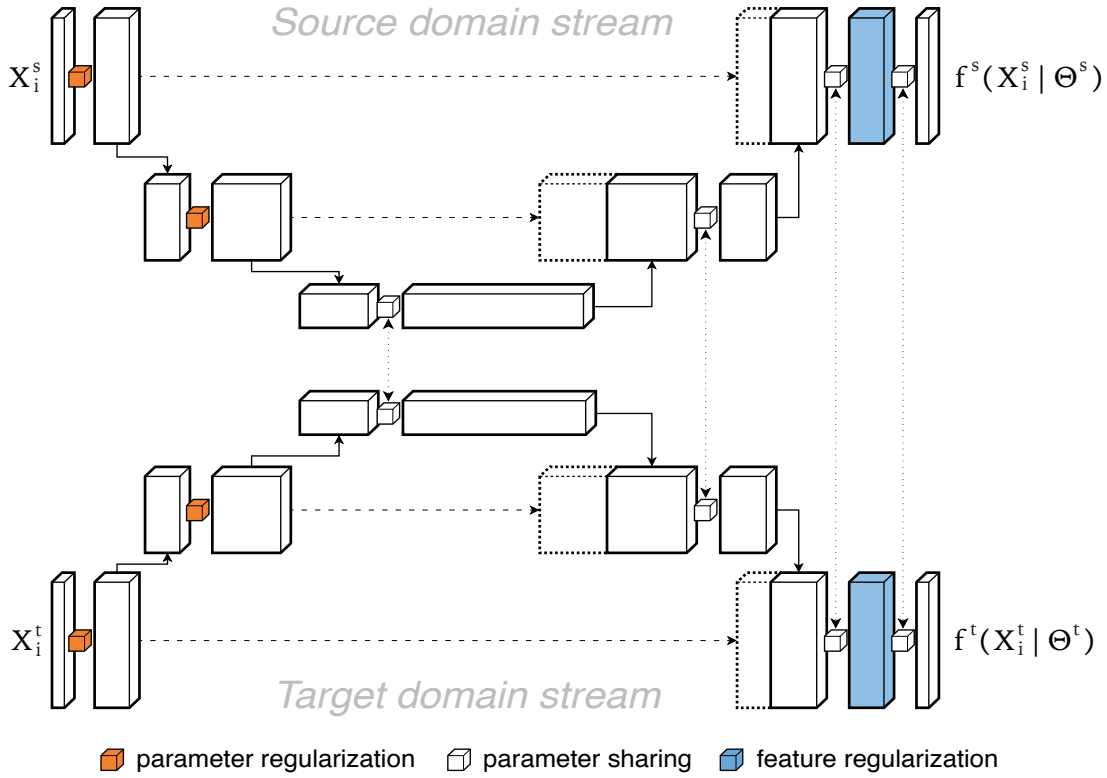


Figure 4.7 Our reference deep learning model, a two-stream U-Net (Bermúdez-Chacón et al., 2018), consists of individual streams that process images in each domain. The parameters of corresponding layers in both streams are co-regularized. Our approach enables this architecture to work in the unsupervised scenario.

with scalar model parameters a_j and b_j to be learned jointly with the network weights during training. This effectively binds the pairs of parameters by encouraging the target domain parameters to restrict their deviation from the source domain ones.

4.2.3 Feature regularization

The target domain stream should predict the segmentations for images coming from the target domain. However, often we do not have enough—or any—annotations on the target domain to guide the training and adjust the target stream parameters in a supervised fashion. One common strategy to compensate for this is to add terms to the loss function that ensure that the predictions keep consistency with what we expect. In the two domain scenario, we achieve this consistency by encoding images from both domains into representations that are similar. As we have annotations in the source domain, we therefore aim to make target domain image encodings similar to the supervised source domain ones. In order to do so, we apply a regularizer to penalize differences between the distributions of the final feature maps f_o^s and f_o^t produced by the two streams, that is, the representation preceding the classifier layer.

We evaluated two such standard regularization terms.

The first one corresponds to the Minimum Mean Discrepancy (MMD, Gretton et al. (2007)) between the source and target features, and can be expressed as

$$r_o^{\text{MMD}} = \left\| \frac{1}{N^s M} \sum_{i=1}^{N^s} \sum_{\mathbf{x} \in \mathbf{M}} \phi(f_o^s(X_i^s; \Theta^s)_{\mathbf{x}}) - \frac{1}{N^t M} \sum_{i=1}^{N^t} \sum_{\mathbf{x} \in \mathbf{M}} \phi(f_o^t(X_i^t; \Theta^t)_{\mathbf{x}}) \right\|_{\mathcal{H}}^2, \quad (4.5)$$

where \mathbf{x} sums over the spatial positions \mathbf{M} on the output feature maps, $M = |\mathbf{M}|$, and $\phi(\cdot)$ is a mapping to a reproducing kernel Hilbert space. The MMD can in fact be expressed in terms of a kernel function, as

$$r_o^{\text{MMD}} = \sum_{\substack{i,j \\ \mathbf{x}, \mathbf{y}}} \frac{k\left(f_o^s(X_i^s)_{\mathbf{x}}, f_o^s(X_j^s)_{\mathbf{y}}\right)}{(N^s M)^2} - 2 \sum_{\substack{i,j \\ \mathbf{x}, \mathbf{y}}} \frac{k\left(f_o^s(X_i^s)_{\mathbf{x}}, f_o^s(X_j^t)_{\mathbf{y}}\right)}{N^s N^t M^2} + \sum_{\substack{i,j \\ \mathbf{x}, \mathbf{y}}} \frac{k\left(f_o^s(X_i^t)_{\mathbf{x}}, f_o^s(X_j^t)_{\mathbf{y}}\right)}{(N^t M)^2} \quad (4.6)$$

and in practice we use the Radial Basis Function as the kernel k .

The second regularizer we evaluated is based on the correlation alignment method of Sun et al. (2016). Let C_o^s be the correlation matrix of the final source feature map $f_o^s(X_i^s, \Theta^s)$, and C_o^t be the equivalent matrix for the target domain. We then write

$$r_o^{\text{corr}}(f_o^s, f_o^t) = \|C_o^s - C_o^t\|_F^2. \quad (4.7)$$

This expression is more permissive than the MMD formulation, because instead of attempting to align the entire distributions in kernel space, it enforces agreement between distribution orientations only, and can be more beneficial in case of larger domain shifts.

4.2.4 Segmentation scoring

We pose the segmentation problem as a binary classification of whether pixels from the image belong to the object of interest or not. For binary classification, the most commonly used loss function is the binary cross-entropy (Shen, 2005). However, the most popular *metric* to assess segmentation quality is the Jaccard index, or Intersection over Union (*IoU*), introduced in the previous chapter in Equation 3.12. An alternative way to express the Jaccard index for $\{0, 1\}$ binary labels is

$$J(\hat{Y}, Y) = \frac{\sum_i y_i \cdot \mathbb{1}[\hat{y}_i = y_i]}{\sum_i y_i + \sum_i (1 - y_i) \cdot \mathbb{1}[\hat{y}_i = 1 - y_i]}. \quad (4.8)$$

The Jaccard Index is not differentiable and hence cannot be directly used when performing

training with backpropagation. We therefore introduce a differentiable, “soft” version of it

$$J_{\text{soft}}(\hat{Y}, Y) = \frac{\sum_i y_i \cdot s(\hat{y}_i, y_i)}{\sum_i y_i + \sum_i (1 - y_i) \cdot s(\hat{y}_i, 1 - y_i)}, \quad (4.9)$$

based on approximating the indicator function. Here, $s(\cdot, \cdot)$ is some similarity metric that outputs values close to 0 if the values differ, and to 1 if they are equal.

In particular, we use the Radial Basis Function $s(a, b) = \exp(-\|a - b\|^2 / \sigma)$ as similarity metric. The parameter σ controls how close to a true indicator function the soft formulation is. Since the differences between expected and predicted labels are bounded to the $[0, 1]$ range, if we want to approximate the indicator function with the RBF to a certain tolerance ϵ , we should set $\sigma = -\frac{1}{\ln \epsilon}$. This ϵ is a hyperparameter of the method and we set it to 0.001. Other similarity metrics can also be adapted and used in an analogous way. In addition, adapting other popular count-based metrics akin to the Jaccard index, such as the F1 score and the Matthews correlation coefficient, following differentiable formulations equivalent to Equation 4.9 is straightforward.

We then write the segmentation loss function as

$$L_y(\mathbf{y}, \hat{\mathbf{y}}) = 1 - J_{\text{soft}}(\mathbf{y}, \hat{\mathbf{y}}), \quad (4.10)$$

which approximates the difference between the Jaccard index of our predictions and the one that would be obtained for perfect segmentations.

4.2.5 Loss function

The segmentation loss functions from the source domain L_y^s , and for the target domain L_y^t —in case there exists annotated data—are combined with the weight regularizers r_w and the output layer regularizer r_o into a joint loss function, as

$$L = \lambda_y^s L_y^s + \lambda_y^t L_y^t + \lambda_w r_w + \lambda_o r_o \quad (4.11)$$

with relative weights λ_y^s , λ_y^t , λ_w , and λ_o for the different components. This loss function is then minimized by backpropagation via gradient-based optimization methods.

4.3 Evaluation

Below, we first discuss the details of our experimental setup and then present our results for both unsupervised and semi-supervised domain adaptation.

4.3.1 Datasets

We evaluate our method for the synapse and mitochondria segmentation datasets introduced in Chapter 3.

4.3.2 Experimental setup

Each stream from our two-stream U-Net architecture follows the design of Ronneberger et al. (2015) (Figure 4.6), which performs 2D image encoding by four levels of downscaling. In order to simplify the selection of the set Ω of corresponding layers between streams to be regularized, we divide them into three groups: layers that perform encoding, layers that operate on the encoded representation, and layers that perform decoding. We found that we obtained the best performance when regularizing the encoding layers and sharing the parameters from all the rest, by evaluating all possible combinations of layer groups and layer regularization strategies of Section 4.2.2 under supervised cross-validation. As output layer regularizer r_o we found the formulation based on correlation alignment to be more beneficial for our method on the evaluated datasets.

As our network uses partial annotations inferred from computed consensus heatmaps, which in turn utilize most of the machinery used to find the visual correspondences of Chapter 3, we apply the same setup as described in the experimental evaluation of Section 3.3.4, so as to make our experiments comparable. In particular, our patch matching works on templates of shape $35 \times 35 \times 35$, and we use the same 300 foreground and 600 background locations from the source domains to compute the heatmaps. We accept or reject soft annotations by using $\tau_{\text{low}} = -0.5$ and $\tau_{\text{high}} = 0.5$ in Algorithm 1. We tune the parameters of our network for 2000 epochs by using the Adam optimizer (Kingma and Ba, 2015) with a learning rate of 10^{-4} .

We evaluate the following experiments:

Consensus heatmap-based MIL. (CH-MIL) This is the approach of Chapter 3 that relies on Multiple Instance Learning to directly use the visual correspondences to refine classifier parameters learned on the source domain so that they work for the target domain, but top correspondences are chosen here by looking at the Consensus Heatmap score rather than the raw *NCC* to eliminate the weaker ones, as explained in Section 4.1.2.

Two-stream U-Net. (2U-Net) The two-stream U-Net described in Section 4.2, used with full source domain supervision and limited target domains supervision, for the semi-supervised domain adaptation scenario.

Heatmap-based two-stream U-Net. (CH-2UNet) The two-stream U-Net of Section 4.2, using the thresholded heatmaps as partial annotations to provide a supervisory signal in the target domain, as described in Section 4.1.3. We take as initial parameter values for the target domain stream those learned in the source domain stream under full supervision.

4.3.3 Unsupervised domain adaptation

We compare the results obtained with our methods against those of the following baselines that can also operate without manual annotations in the target domain:

Boosted decision trees. (SD-BT) An ensemble of decision trees trained by gradient boosting on data from the source domain is applied to the target domain as is.

U-Net. (SD-UNet) A U-Net (Ronneberger et al., 2015) trained on the source domain is applied to the target domain as is.

Subspace Alignment + Boosting. (SSAB) Align the source and target domain PCA subspaces and find a decision boundary by boosting (Fernando et al., 2013).

Correspondence-based MIL (CB-MIL). This is the approach introduced in Chapter 3.

Shared latent space + Boosting. (SLSB) Learn mappings for both source and target domains onto a shared space, where a boosting classifier determines the pixel-wise class assignments (Becker et al., 2015).

Deep correlation alignment. (CorAl) Align the correlations of the data from different layers in a deep network (Sun et al., 2016).

Deep domain confusion. (DDC) Align distribution of target and source representations using Maximum Mean Discrepancy (MMD) metric (Tzeng et al., 2014).

Domain-adversarial network. (DAN) Use adversarial training to learn a deep neural network so that the target domain’s encoding is indistinguishable from the source domain’s one (Javanmardi and Tasdizen, 2018).

For completeness, we also evaluate **full supervision** (*Full TD*), that is, using a large amount of annotated training data to train a U-Net on the target domain, which can be thought of as an upper bound of what an unsupervised method can possibly achieve for the specific network architecture.

Results

We present a quantitative comparison of the methods in Table 4.1. Overall, our CH-2UNet approach yields the best results on both datasets, closely approaching *Full TD*. Our CB-MIL and CH-MIL methods also outperform all the baselines, including the deep learning ones, despite ours using handcrafted features (Becker et al., 2012). This demonstrates the benefits of leveraging appearance-based correspondences to provide some level of weak supervision in the adaptation process. Note that CH-MIL yields higher Jaccard indices than CB-MIL, which shows the importance of re-ordering the correspondences according to our consensus heatmaps.

Table 4.1 Quantitative segmentation results (Jaccard indices) for different unsupervised domain adaptation methods. Methods introduced in this chapter are presented in bold.

	SD-BT	SD-UNet	SA+B	CorAI	DDC	DAN	CB-MIL	CH-MIL	CH-2UNet	<i>Full TD</i>
Synapses	0.222	0.257	0.135	0.310	0.481	0.565	0.571	0.580	0.660	0.751
Mitochondria	0.500	0.505	0.590	0.481	0.543	0.598	0.618	0.625	0.742	0.806

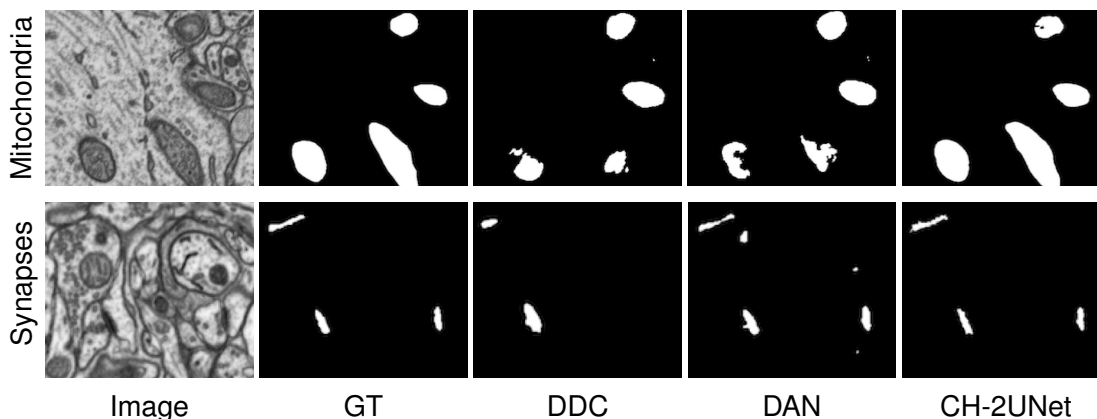


Figure 4.8 Qualitative segmentation results for unsupervised domain adaptation. Example segmentations obtained from different methods.

4.3.4 Semi-supervised domain adaptation

None of the techniques discussed above uses manual annotations in the target domain. However, there are also techniques that can leverage *small* amounts of data in the target domain. They include the following:

Fine-tuning boosting decision trees. (FT-BT) Retrain the parameters of an ensemble of decision trees trained with gradient boosting on the source domain model, using a small quantity of annotated data from the target domain.

Fine-tuning U-Net. (FT-UNet) Retrain the parameters of a U-Net (Ronneberger et al., 2015) trained on the source domain model, using a small quantity of annotated data from the target domain.

Shared latent space + Boosting. (SLSB) Same as in the unsupervised scenario, but leveraging the target annotations when training the boosting classifier (Becker et al., 2015).

Deep domain confusion. (DDC) Same as in the unsupervised scenario, but leveraging the target annotations when training the deep network (Tzeng et al., 2014).

Domain-adversarial network. (DAN) Same as in the unsupervised scenario, but leveraging the target annotations when training the deep network (Javanmardi and Tasdizen, 2018).

Chapter 4. Consensus heatmaps and the two-stream U-Net

To compare against those, we provide annotations for 10% of randomly selected non-contiguous slices from the target domain stacks. For CB-MIL and CH-MIL, we modify our formulation by adding a supervised classification loss term for the target domain that considers those existing annotations. For CH-2UNet, we take the selected target domain annotations and replace the corresponding slices on the thresholded heatmaps.

We report our results in Table 4.2. Note that, as in the unsupervised case, our CH-2UNet approach is the top-performer for both datasets, with the gap between our results and the *Full TD* upper bound being reduced even further. By comparing our approach with the 2UNet baseline, which corresponds to a version of our approach without consensus heatmaps, we note that, while the benefits of heatmaps is imperceptible in the Synapse case, they do improve segmentation accuracy in the Mitochondria one.

Table 4.2 *Quantitative segmentation results (Jaccard indices) for different domain adaptation methods under a small amount of supervision. Methods introduced in this chapter are presented in bold.*

	SD-BT	SD-UNet	FT-BT	FT-UNet	DDC	DAN	SLSB	CB-MIL	CH-MIL	2U-Net	CH-2UNet	<i>Full TD</i>
Synapses	0.222	0.257	0.197	0.441	0.603	0.662	0.670	0.664	0.691	0.723	0.723	0.751
Mitochondria	0.500	0.505	0.574	0.698	0.610	0.661	0.634	0.697	0.708	0.746	0.751	0.806

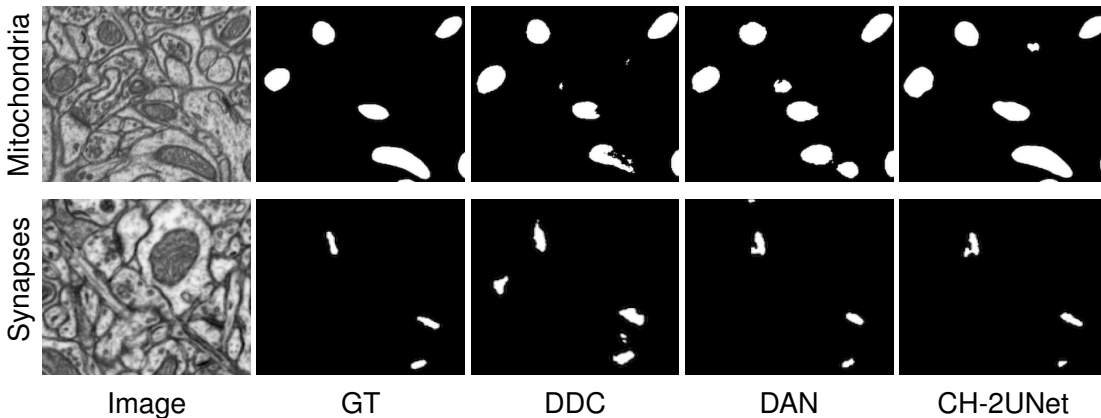


Figure 4.9 *Qualitative segmentation results for semi-supervised domain adaptation. Example segmentations obtained from different methods.*

4.3.5 Experiments under large domain shift

We further validate our method for datasets exhibiting larger visual differences. As shown in Figure 4.10, we take the source domains to be the same mouse striatum and cerebellum volumes for mitochondria and synapse segmentation as before and as explained in Section 4.3.1. As target domains, we now use the publicly available drosophila dataset¹. It comprises 20 fully

¹<https://github.com/unidesigner/groundtruth-drosophila-vnc>

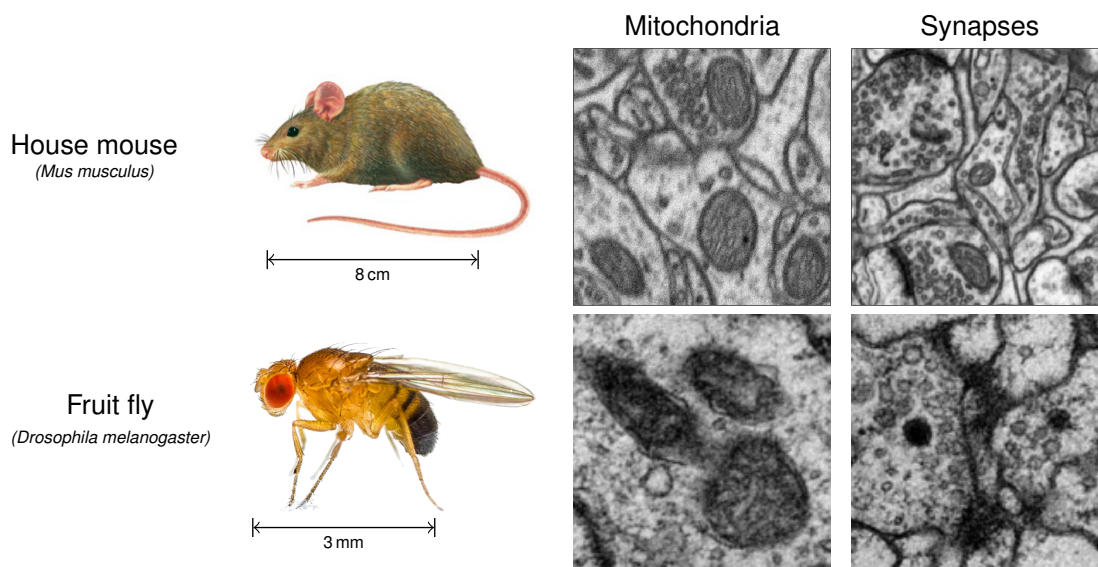


Figure 4.10 Example patches from source (Mouse) and target (Drosophila) domains under a large domain shift.

annotated 2D slices of dimensions 1024×1024 each, acquired from a Serial Section Scanning Electron Microscope, at a resolution of 4.6nm.

The results of a transductive evaluation in this scenario are presented in Table 4.3. As the source and target domain acquisitions come from such unrelated species, they exhibit significant differences, which are both visually apparent and evident from the extremely poor performance of models trained on the source domain. As expected, the increase in domain shift results in a consistent drop in performance for all methods compared to scenarios with smaller domain differences, especially for synapse segmentation, which is significantly more difficult than mitochondria segmentation. However, the behavior of our method is largely consistent with the reported results of Section 4.3.3, and exhibits a clear qualitative improvement when compared with all the other domain adaptation techniques, which demonstrates that our method can handle large domain shifts and is more robust at doing so than the alternatives.

Table 4.3 Quantitative segmentation results (Jaccard indices) for different domain adaptation methods under supervision, for a large domain shift. Methods introduced in this chapter are shown in bold.

	SD-UNet	FT-UNet	DAN	SLSB	2U-Net	CH-2UNet	<i>Full TD</i>
Synapses	0.114	0.380	0.443	0.409	0.521	0.558	<i>0.686</i>
Mitochondria	0.154	0.402	0.452	0.492	0.577	0.608	<i>0.747</i>

4.4 Conclusion

This chapter introduced a technique to aggregate visual correspondences, so as to turn sparse visual matches into a dense *consensus heatmap*, which extracts and encodes important structural information by weighted-voting for promising locations and their surroundings. In addition, we have introduced the *two-stream U-Net*, a general-purpose deep network architecture for domain adaptation, which not only shows good performance in a semi-supervised setting, but also greatly benefits from the partial, soft-annotated data coming from the consensus heatmaps for unsupervised adaptation.

As demonstrated from our results, the weak supervision coming from the consensus heatmaps successfully bridges a large part of the performance gap between unsupervised and fully-supervised training. As consensus heatmaps are learned in an unsupervised fashion, we believe that, as shown for the two-stream U-Net, other supervised and semi-supervised domain adaptation methods can be easily converted into unsupervised ones by using consensus heatmaps as supervisory signal.

In addition, our experimental validation of the two-stream U-Net in the semi-supervised learning scenario, even without the use of consensus heatmaps, suggests that such architecture responds well to scenarios with low supervision. This makes it a good candidate to be incorporated into other semi-supervised pipelines. One possible such scenario is, for example, by alternating labeling and refinement in an Active Learning setting.

The techniques introduced in this chapter, as well as those from the previous one, can be seen as self-supervised methods, in that explicit human annotations are not required, but soft labels are inferred from our understanding of the problem and used as supervisory signal. The resulting domain adaptation methods make limited use of the data from the source domain, either for pretraining, as in Chapter 3, or for parameter regularization, as in this chapter, where each stream of the two-stream U-Net is only exposed to data from a single domain. Furthermore, the processing on both domains uses the same architecture, disregarding potential asymmetries in the complexities of the domains. As the soft target domain signal might still be insufficient for successful adaptation, we explore in the next chapter a more general strategy that allows data coming from both domains to directly influence all model parameters, and also adapts the specific network architecture for each domain.

Another obvious limitation of the approaches presented both in this chapter and in Chapter 3 is that, as they are designed with the segmentation problem in mind, they heavily depend on the structure of the annotations, and as such, will not be directly applicable for problems in image understanding with unstructured labels. The method presented in next chapter introduces a more general network architecture design paradigm that does not suffer from this shortcoming.

Domain-adaptive multiflow networks

CHAPTER 5

Image domains can vary in complexity, and processing them with a machine learning algorithm can be more or less difficult depending on the domain. In this chapter, we present a dynamic network architecture for domain adaptation that acknowledges this fact, and learns to selectively assign either different or shared operations per domain, according to its computational requirements. The core idea is to replace groups of contiguous layers with different sets of parallel operations—or flows—and let the training decide the best combination of those flows in a domain-specific fashion via trainable gates. This *domain-adaptive multiflow network* is significantly more flexible than traditional approaches that either share a single set of operations or perform regularization on corresponding source and target domain parameters. That gained flexibility not only translates into improved performance for diverse visual tasks, including classification, segmentation, and tracking, but also enables seamless integration of arbitrarily many domains when available, and under varying levels of supervision.

5.1 Motivation

The methods introduced in previous chapters exploit Multiple Instance Learning and parameter regularization to leverage a supervisory signal—visual correspondences and consensus heatmaps—in order to perform domain adaptation. While effective for the showcased applications, they are still dependent on the quality and quantity of the supervisory target domain signal and can lead to overfitting when it is scarce or to bias when the soft annotations do not represent the target domain well enough.

An underlying issue with both approaches, and which is prevalent as well in many other popular techniques, is that the transformations that both domains undergo are conceptualized as either exclusive per domain or completely shared between domains. The former occurs in the two-stream U-Net when different, identical network streams process one domain each. In the latter, one hopes to either learn a single sequence of operations that encode the domains into the same representation, or—as in the Boosted Decision Trees method of Chapter 3—to be able to correct the parameters governing those operations so that they work on different data.

In either case, the adaptation depends strongly on the target domain supervision, and the role of the source domain is relegated to pretraining or regularization. More importantly, those strategies assume that both domains require the same computation; in real domain adaptation scenarios, however, it is often the case that some domains are simpler or more complex than others—synthetic images and images taken “in-the-wild”, to name an example—and hence possibly require simpler or more complex processing.

The idea that we propose in this chapter is that, for effective domain adaptation, it is more convenient to allow all available domains to influence the value of the network parameters for all layers at early stages of the training, and as training progresses, to modulate the degree to which different layers specialize to different domains. In other words, we propose a paradigm where different domains are assigned different computation, and where the specific set of operations to which a domain is subjected, that is, the specific architecture per domain, is decided dynamically as part of the training. Our formulation reflects the intuition that source and target domain networks should be similar because they solve closely related problems, but should also perform domain-specific computations to offset the domain shift, and to prevent negative transfer.

Our instantiation of this idea, which we call a *domain-adaptive multiflow network* (DAMNet), is an architecture and training protocol for learning a *common representation* between different domains; that is to say, we dynamically learn suitable network architectures per domain so that their data arrives to the same latent representation from potentially different processing, and once under such representation, regular deep architectures can perform the desired task in a domain-agnostic fashion.

More concretely, a DAMNet is a multiflow architecture that sends the data through multiple network branches in parallel. What gives it the necessary flexibility are trainable gates that are tuned to modulate and combine the outputs of these branches, as shown in Figure 5.1. Assigning to each domain its own set of gates allows the global network to learn what set of computations should be carried out for each one. As an additional benefit, and in contrast to previous strategies for unsharing the source and target streams (Rozantsev et al., 2018, 2019), the suggested formulation naturally extends to the availability of more than two domains, under different levels of supervision.

5.2 Multiflow networks

5.2.1 Single domain

A standard feed-forward neural network can be thought of as a sequence of N_f operations $f^{(i)}(\cdot)$, $1 \leq i \leq N_f$, each transforming the output of the previous one. Given an input image X , this can be expressed as

$$\begin{aligned} X^{(0)} &= X \\ X^{(i)} &= f^{(i)}(X^{(i-1)}). \end{aligned} \tag{5.1}$$

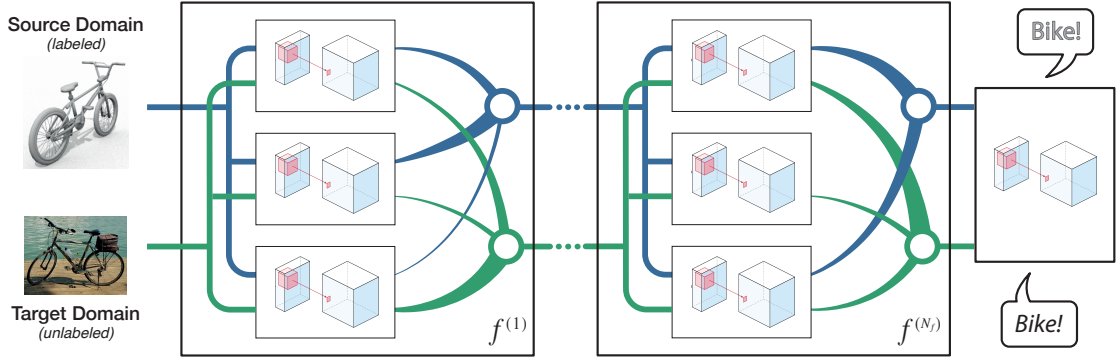


Figure 5.1 A **domain adaptive multiflow network** is a sequence of computational units $f^{(i)}$, each of which processes the data in parallel flows, whose outputs are then aggregated in a weighted manner by a gate to obtain a single response. To allow for domain-adaptive computations, each domain has its own set of gates, one for each computational unit, which combine the flows in different ways. As a result, some computations are shared across domains while others are domain-specific.

As a general convention, each operation $f^{(i)}$ can represent either a single layer or multiple ones. Our formulation extends this definition by replacing each $f^{(i)}$ by multiple parallel computations, as shown in Figure 5.2. More specifically, we substitute each operation $f^{(i)}$ with a *computational unit* $\{f_1^{(i)}, \dots, f_{K_i}^{(i)}\}$ consisting of K_i parallel *flows*. In general, each flow encapsulates an arbitrary sequence of operations, which could potentially differ from flow to flow.

Given this definition, we write the output of each computational unit as

$$X^{(i)} = \hat{\Sigma} \left(f_1^{(i)}(X^{(i-1)}), \dots, f_{K_i}^{(i)}(X^{(i-1)}) \right), \tag{5.2}$$

where $\hat{\Sigma}(\cdot)$ is an aggregation operator that combines the outputs from different flows into a single response. Such aggregation can be defined in many ways. It could be a simple summation or averaging that gives all outputs equal importance or, at the opposite end of the spectrum, a multiplexer that selects a single flow and ignores the rest. The only practical restriction that should be imposed is that the resulting aggregation be of dimensions consistent with what the next computational unit expects.

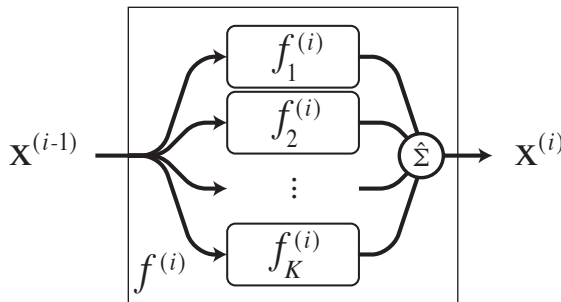


Figure 5.2 A **computational unit** $f^{(i)}$ is an aggregation of the outputs of parallel computations, or flows, $f_j^{(i)}$.

To cover the range between the two alternatives above, we introduce *learnable gates* that enable the network to determine what relative importance the different flows should be given. Our gates perform a weighted combination of the flow outputs. Each gate is controlled by a set of K_i *activation weights* $\{\phi_j^{(i)}\}_{j=1}^{K_i}$, and a computational unit returns

$$X^{(i)} = \sum_{j=1}^{K_i} \phi_j^{(i)} \cdot f_j^{(i)}(X^{(i-1)}). \quad (5.3)$$

If $\forall j, \phi_j^{(i)} = 1$, the gate performs a simple summation. If $\phi_j^{(i)} = 1$ for a single j and 0 for the others, it selects the output of flow j , ignoring the rest, and hence behaves as a multiplexer.

The activation weights $\phi_j^{(i)}$ enable us to modulate the computational graph of network block $f^{(i)}$. To bound them and encourage the network to either select or discard each flow in a computational unit, we write them in terms of sigmoid functions with adaptive steepness, that is,

$$\phi_j^{(i)} = \frac{1}{1 + \exp(-\pi^{(i)} \cdot g_j^{(i)})}, \quad (5.4)$$

where the $g_j^{(i)}$ s are learnable unbounded model parameters, and $\pi^{(i)}$ controls the *plasticity* of the activation—the rate at which $\phi_j^{(i)}$ varies between the binary values 0 and 1 for block i .

During training, we initially set $\pi^{(i)}$ to a small value, which enables the network to explore different gate configurations. We then apply a cooling schedule on our activations, by progressively increasing $\pi^{(i)}$ over time, so as to encourage the gates to reach a firm decision.

Note that this formulation does not require $\sum_{j=1}^{K_i} \phi_j^{(i)} = 1$, that is, we do not require the aggregated output $X^{(i)}$ to be a convex combination of the flow outputs $f_j^{(i)}(X^{(i-1)})$. This is deliberate because allowing the activation weights to be independent from one another provides additional flexibility for the network to learn general additive relationships such as residual mappings.

Finally, a *multiflow network* is the concatenation of multiple computational units, as shown in Figure 5.1.

Some minor considerations must be taken into account when designing multiflow architectures. For the aggregation within each unit $f^{(i)}$ to be possible under our gating strategy, the $f_j^{(i)}$ s' outputs must be of matching shapes. Furthermore, as mentioned above, and as in standard networks, two computational units can be attached only if the output shape of the first one matches the input shape of the second. Lastly, although it would be possible to define computational units at any point in the network architecture, in practice, we take them to correspond to groups of layers that are semantically related. For example, a succession of convolutions, pooling and non-linear operations would be grouped together into a single computational unit under this scheme.

5.2.2 Domain-adaptive multiflow networks

We first discuss how we turn our multiflow networks described above into DAMNets when there are only two domains. We then extend this formulation to the multi-domain case.

Two domains

Our ultimate goal is to perform domain adaptation, that is, to leverage a large amount of labeled images, $\mathbf{X}^s = \{X_1^s, \dots, X_N^s\}$ with corresponding annotations $\mathbf{Y}^s = \{y_1^s, \dots, y_N^s\}$, drawn from a source domain, to train a model for a target domain, whose data distribution is different and for which we only have access to unlabeled images $\mathbf{X}^t = \{X_1^t, \dots, X_M^t\}$.

To this end, we extend the gated networks of Section 5.2.1 by defining two sets of gates, one for the source domain and one for the target one. Let $\{(\phi^s)_j^{(i)}\}_{j=1}^{K_i}$ and $\{(\phi^t)_j^{(i)}\}_{j=1}^{K_i}$ be the corresponding source and target activation weights for computational unit $f^{(i)}$, respectively. Given a sample, X^d coming from a domain $d \in \{s, t\}$, we take the corresponding outputs of the i -th computational unit to be

$$(X^d)^{(i)} = \sum_{j=1}^{K_i} (\phi^d)_j^{(i)} \cdot f_j^{(i)} \left((X^d)^{(i-1)} \right). \quad (5.5)$$

Note that under this formulation, the domain identity d of the sample is required in order to select the appropriate $(\phi^d)_j^{(i)}$. Aside from that, the mechanics of the network remain intact with respect to Section 5.2.1.

The concatenated computational units $f_{\text{DAMN}} = f^{(1)} \circ f^{(2)} \circ \dots \circ f^{(N_f)}$ forming the DAMNet encode sample X^d from domain d into a feature vector $X^z = f_{\text{DAMN}}(X^d, d)$. Since gates for different domains are set independently from one another, the outputs of the flows for each computational unit are combined in a domain-specific manner, dictated by the activation weights $(\phi^d)_j^{(i)}$, even though the flows themselves are shared.

Domain adaptation happens by enforcing the encodings on both domains, $f_{\text{DAMN}}(X^s, s)$ and $f_{\text{DAMN}}(X^t, t)$, to follow a common distribution, the mechanics of which are explained below. Then, a small network f_{out} operates directly on the encodings and returns the desired output $\hat{y} = f_{\text{out}}(X^z)$, thus subjecting the encodings for all samples to the same set of operations. In this way, the samples are encoded to a common space regardless of their original domain, but arrive to it through potentially different computations, which *are* bound to their domains.

Figure 5.3 depicts this process. It makes it unnecessary to predefine independent network architectures for each stream (Zoph and Le, 2017; Liu et al., 2018; Pham et al., 2018) or to have each stream learn its own weights as in Tzeng et al. (2017); Rozantsev et al. (2019). More concretely, the network can learn to share weights for computational unit $f^{(i)}$ by setting $(\phi^s)_j^{(i)} = (\phi^t)_j^{(i)}, \forall j$. It can also learn to fully untie the weights by having $A_i^S \cap A_i^T = \emptyset$, where A_i^S and A_i^T denote the set of non-zero activations in the two domains. Finally, it can learn to use more computation

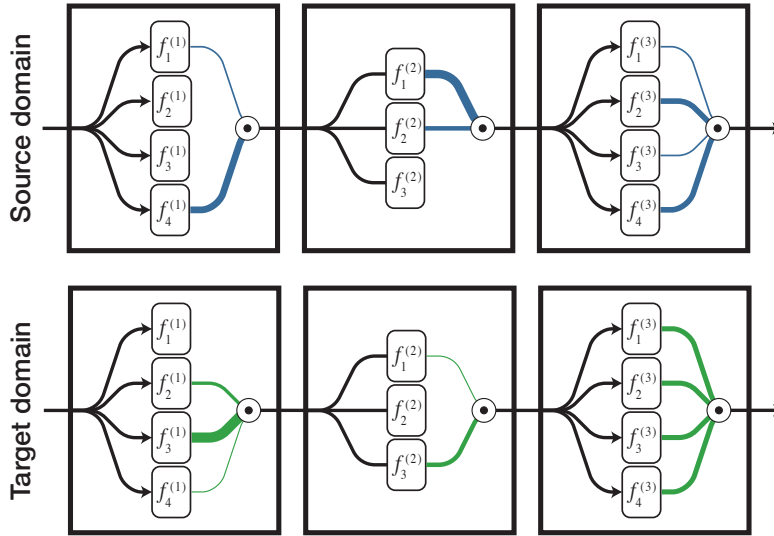


Figure 5.3 Domain-specific computations. The top row shows the computational graph for the source domain, and the bottom row for the target one, for the same network. While both domains share the same computational units, their outputs are obtained by different aggregations of their inner operations. For example, in the first unit, the source domain does not use the middle two operations, whereas the target domain does. By contrast, both domains exploit the fourth operation. In essence, this scheme adapts the amount of computation that each domain is subjected to.

for one domain than for the other by setting $(\phi^s)_j^{(i)} > 0$ for two different flows $f_j^{(i)}$ while having only a single non-zero $(\phi^t)_j^{(i)}$, for a particular computational unit $f^{(i)}$.

Avoiding flow sharing. The above formulation treats all flows for each computational unit as potentially sharable between domains, and exposes all flows to data coming from both domains. However, we can identify cases where it is desirable not to share *at all*. One example of this are *batch-normalization layers*, which accumulate and update statistics of the data over time, even during the forward pass, and are then best exposed to a single domain to learn domain-specific statistics. We solve this by introducing computational units with two flows, whose gates are fixed, yet domain specific, and configured to act as multiplexers to relay input from a single domain to each flow.

Multiple domains

The formulation outlined above extends naturally to more than two domains, by assigning one set of gates per domain. This enables us to exploit annotated data from different source domains, and even to potentially handle multiple target domains simultaneously. In this generalized case, we introduce governing sets of gates with activations $\phi^{d_1}, \dots, \phi^{d_D}$ for D different domains. They act in the same way as in the two-domain case and the overall architecture remains similar.

Training

When training our models, we jointly optimize the gate parameters $(g^d)_j^{(i)}$, from Equation 5.4, along with the other network parameters, using standard back-propagation. To this end, we make use of a composite loss function, designed to encourage correct classification of annotated examples and to align the distributions of all domains, using labeled and unlabeled samples. This loss can be expressed as

$$L_{\text{DAMNet}} = \frac{1}{|\ell|} \sum_{n=1}^{|\ell|} L_y(y_n, \hat{y}_n) + \frac{1}{|\ell \cup u|} \sum_{n=1}^{|\ell \cup u|} L_d(\mathbf{d}_n, \hat{\mathbf{d}}_n), \quad (5.6)$$

where ℓ and u are the sets of labeled and unlabeled samples, respectively, coming from all the domains, and where we assumed, without loss of generality, that the samples are ordered.

The first term in this loss, $L_y(y, \hat{y})$, compares the ground-truth and predictions, and varies with the problem at hand. For object classification, for example, it is the standard cross-entropy, which compares the true class probabilities y_n with the predicted ones $\hat{y}_n = f_{\text{out}}(X^z)$, where, as discussed in Section 5.2.2, $X^z = f_{\text{DAMN}}(X^d, d)$ is the feature encoding of sample X from domain d . Notice that there is no distinction here between source and target domains, which means we can seamlessly incorporate different degrees of supervision from both source and target domains.

For the second term of our loss function, which encodes distribution alignment, we rely on the domain confusion strategy of Ganin and Lempitsky (2015), which is commonly used in existing frameworks. Specifically, for D domains, we make use of an auxiliary domain classifier network f_d that predicts a D -dimensional vector of domain probabilities $\hat{\mathbf{d}}$ given the feature vector X^z . Following the gradient reversal technique of Ganin and Lempitsky (2015), we express the second term in our loss as

$$L_d(\mathbf{d}, \hat{\mathbf{d}}) = - \sum_{i=1}^D \mathbf{d}_i \log(\hat{\mathbf{d}}_i), \quad (5.7)$$

where \mathbf{d} is the D -dimensional one-hot encoding of the ground-truth domain, \mathbf{d}_i indicates the i -th element of \mathbf{d} , and $\hat{\mathbf{d}} = f_d(R(X^z))$, with R the gradient reversal pseudofunction of Ganin and Lempitsky (2015) that enables to incorporate adversarial training directly into back-propagation. That is, with this loss, standard back-propagation trains jointly the domain classifier to discriminate the domains and the feature extractor $f_{\text{DAMN}}(\cdot)$ to produce features that fool this classifier. The outline of the strategy, as proposed in the original paper, is presented in Figure 5.4.

Pruning. When training is complete and the gates have reached a stable state, the flows whose activations are close to zero are deactivated. This prevents the network from performing computations that are irrelevant and allows us to obtain a more compact network to process the target data.

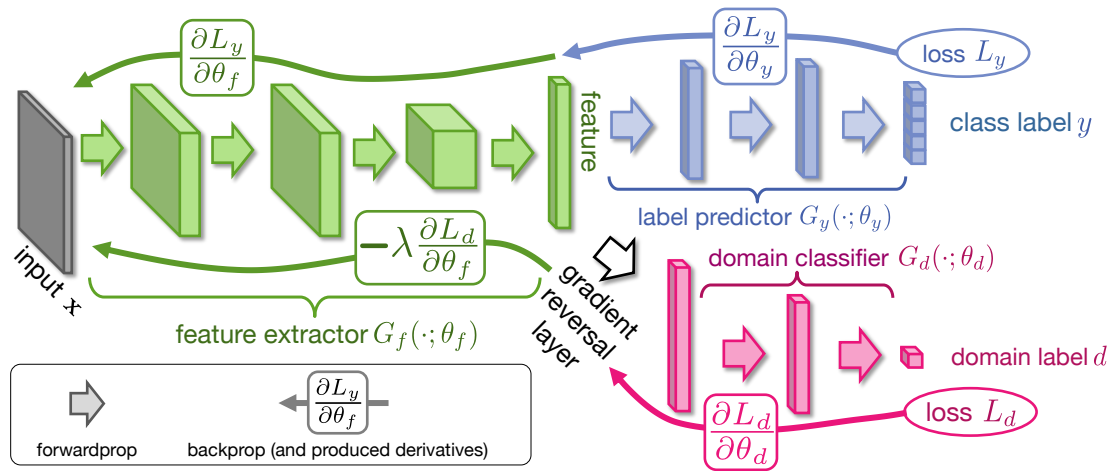


Figure 5.4 Incorporating adversarial domain adaptation directly into backpropagation (Ganin and Lempitsky, 2015).

5.2.3 Relationship to neural architecture search

DAMNets effectively implement dynamic, custom architectures per domain for feature extraction. The field of neural architecture search is ample and an active niche of research (Elsken et al., 2019). Its applications to domain adaptation, however, have not been exhaustively explored. Nevertheless, we provide here a concise review of the current tendencies in neural architecture search, and comment on the similitudes and differences with out DAMNets.

As the performance of a neural network is tightly linked to its structure, there has been a recent push towards automatically determining the best architecture for the problem at hand. While neural architecture search techniques (Zoph and Le, 2017; Liu et al., 2018, 2019; Pham et al., 2018; Zoph et al., 2018; Real et al., 2019; Noy et al., 2019) aim to find one fixed architecture for a given dataset, other works have focused on dynamically adapting the network structure at inference time (Graves, 2016; Ahmed and Torresani, 2017; Shazeer et al., 2017; Veit and Belongie, 2018; Wu et al., 2018). In particular, in Ahmed and Torresani (2017); Shazeer et al. (2017); Veit and Belongie (2018); Bhatia et al. (2019), gates were introduced for this purpose.

DAMNets also regulate computation via gates; their role, however, is substantially different: first, we work with data coming from different domains, whereas these gated methods, with the exception of Bhatia et al. (2019), were all designed to work in the single-domain scenario. Second, and more importantly, these techniques aim to define a different computational path for every test *sample*. By contrast, we seek to determine the right computation for each *domain*. Another consideration is that we freeze our gates for inference while these methods must constantly update theirs. We believe this to be ill-suited to domain adaptation, particularly because learning to adapt the gates for the target domain, for which only unlabeled data is available, is severely under-constrained. This lack of supervision may be manageable when one seeks to define operations for a whole domain, but not when these operations are sample-specific.

5.3 Evaluation

We now evaluate our method on standard benchmark datasets, and compare its performance against different related methods.

5.3.1 Baselines

Since we rely on the domain confusion loss to train our model, we treat the *domain-adversarial neural network* (**DANN**) method of Ganin and Lempitsky (2015), where it was first introduced, as a baseline.

To demonstrate the benefits of our approach over simply untying the source and target stream parameters, we compare our approach against the Residual Parameter Transfer (**RPT**) method of Rozantsev et al. (2019), which constitutes the state of the art in doing so. Note that RPT also relies on the domain confusion loss, which makes our comparison fair. In addition, we report the results of directly applying a model trained on the source domain to the target, without any domain adaptation, which we refer to as “**No DA**”. We also provide the oracle accuracy of a model trained on the fully-labeled target domain, referred to as “**On TD**”.

5.3.2 Implementation details

We adapt different network architectures to the multiframe paradigm for different adaptation problems. For all cases, we initialize our networks’ parameters by training the original versions of those architectures on the source domains, either from scratch, for simple architectures, or by fine-tuning weights learned on ImageNet (Deng et al., 2009), for very deep ones. We then set the parameters of all flows to the values from the corresponding layers. We perform this training on the predefined training splits, when available, or on 75% of the images, otherwise. For simplicity, we use for all computational units as many flows as we have domains, all flows in a computational unit share the same architecture, and the initial values of the gate parameters are defined so as to set the activations to $\frac{1}{K_i}$, for each of the K_i flows at each computational unit. This prevents our networks from initially favoring a particular flow for any domain.

To train our networks, we use Stochastic Gradient Descent with a momentum of 0.9 and a variable learning rate defined by the annealing schedule of Ganin and Lempitsky (2015) as $\mu_p = \frac{\mu_0}{(1+\alpha \cdot p)^\beta}$, where p is the training progress, relative to the total number of training epochs, μ_0 is the initial learning rate, which we take to be 10^{-2} , and $\alpha = 10$ and $\beta = 0.75$ as in Ganin and Lempitsky (2015). We eliminate exploding gradients by ℓ_2 -norm clipping. Furthermore, we modulate the plasticity of the activations at every gate as training progresses by using the relative training progress as the plasticity parameter $\pi^{(i)} = p$. As data preprocessing, we apply mean subtraction, as in Ganin and Lempitsky (2015). We train for 2000 epochs, during which the network is exposed to all the image data from the source and target domains, but only to the annotations from the source domain(s).

Our “*On TD*” oracle is trained on either the preset training splits, when available, or our defined training data and evaluated on the corresponding test data. For the comparison to this oracle to be meaningful, we follow the same strategy for our DAMNets. That is, we use the unlabeled target data from the training splits only and report results on the testing splits. This protocol differs from that of Rozantsev et al. (2019), which relied on a transductive evaluation, where *all* the target images, training and test ones, were seen by the networks during training.

5.3.3 Digit recognition

We use several common digit recognition datasets for our evaluation.

The well-studied **MNIST** (LeCun et al., 1998) dataset consists of black and white images of handwritten digits from 0 to 9. All images are of size 28×28 pixels. We train and evaluate on the standard training and testing splits of 60,000 and 10,000 examples, respectively.

The **MNIST-M** (Ganin and Lempitsky, 2015) dataset is synthetically generated by randomly replacing the foreground and background pixels of random MNIST samples with natural images. Its image size is 32×32 , and we use the standard training and testing splits of 59,001 and 9,001 images.

The Street View House Numbers dataset (**SVHN**) (Netzer et al., 2011) consists of natural scene images of numbers acquired from Google Street View. Its images are also of size 32×32 pixels, and its preset training and testing splits are of 73,257 and 26,032 images, respectively. The SVHN images are centered at the desired digit, but contain clutter, visual artifacts, and distractors from its surroundings. Examples from the three datasets are shown in Figure 5.5.

Figure 5.5 Examples from digit recognition datasets.



Setup

As discussed in Section 5.2, our method is general and can work with any network architecture. To showcase this, we apply it to the LeNet and SVHNet architectures (Ganin and Lempitsky, 2015), which are very simple convolutional networks, well suited for small images. Following Ganin and Lempitsky (2015), we employ LeNet when using the synthetic datasets MNIST

and MNIST-M as source domains, and SVHNet when SVHN acts as source domain. We extend these architectures to multifold ones by defining the computational units as the groups of consecutive convolution, pooling and non-linear operations defined in the original model. For simplicity, we use as many flows within each computational unit as we have domains, and all flows from a computational unit follow the same architecture. The multifold LeNet and SVHNet architectures reflecting the scope of each computational unit are shown in Figure 5.6 and Figure 5.7, respectively, for the two-domain case.

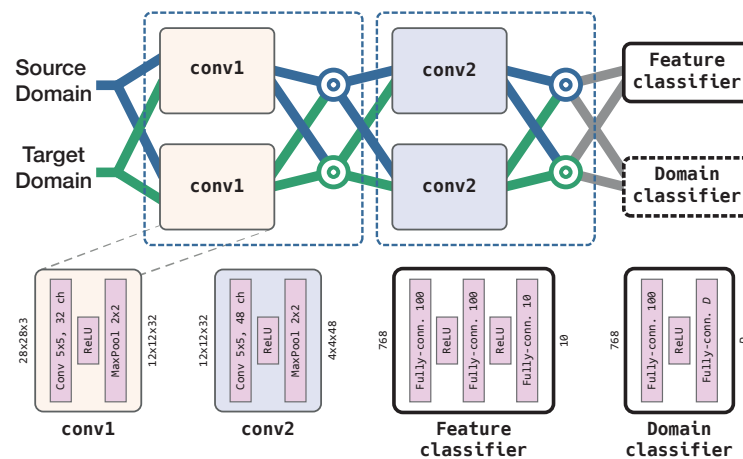


Figure 5.6 *Multifold LeNet.* This architecture is based on the one used by DANN (Ganin and Lempitsky, 2015).

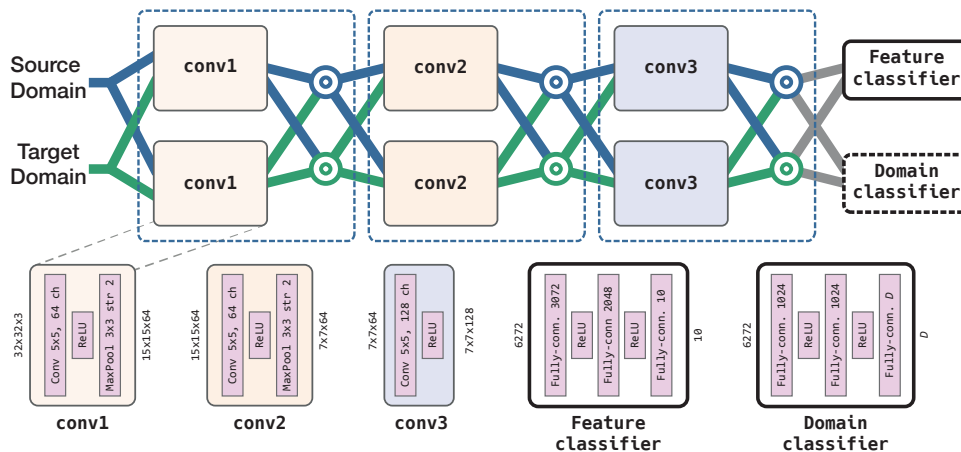


Figure 5.7 *Multifold SVHNet.* This architecture extends the one used by DANN (Ganin and Lempitsky, 2015).

Results

The results of our digit recognition experiments are provided in the top portion of Table 5.1. In addition to the traditional two-domain setup, we also report results when using two fully-annotated source domains simultaneously. Note that Rozantsev et al. (2019) does not apply to this setting, since it was designed to transform a *single* set parameters from source to target.

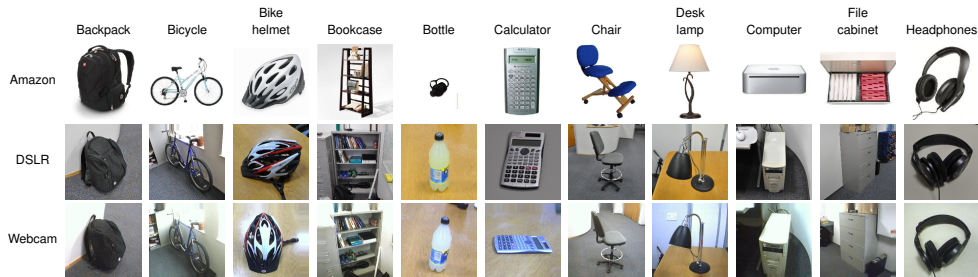
Altogether, our method consistently outperforms the others. Note that the first two rows correspond to the combinations reported in the literature. We believe, however, that the SVHN→MNIST adaptation scenario is quite artificial, since, in practice, one would typically annotate simpler, possibly synthetic images and aim to use real ones at test time. We therefore also report *synthetic*→SVHN cases, which, as can be seen both from the results and from the aspect of the domain examples, are much more challenging.

The multi-source version of our method, in particular, achieves a significant boost over the baselines in this scenario. We ascribe this to the fact that images coming from the MNIST-M dataset show similar characteristics to both MNIST and SVHN, and so the flows have more incentive to learn transformations to encompass a wider range of image appearances, especially those that more resemble the target domain, from the supervised signal. To further demonstrate the potential of our approach in this setting, we replaced its backbone with the much deeper ResNet-50 network, which we detail in the next section, and applied it on upscaled versions of the images. As shown in the row indicated by a \star , this allowed us to achieve an accuracy close to 80%, which is remarkable for such a difficult adaptation task.

5.3.4 Object recognition

We now turn to the task of object recognition, for which we use several domain adaptation benchmark datasets. These datasets are described below.

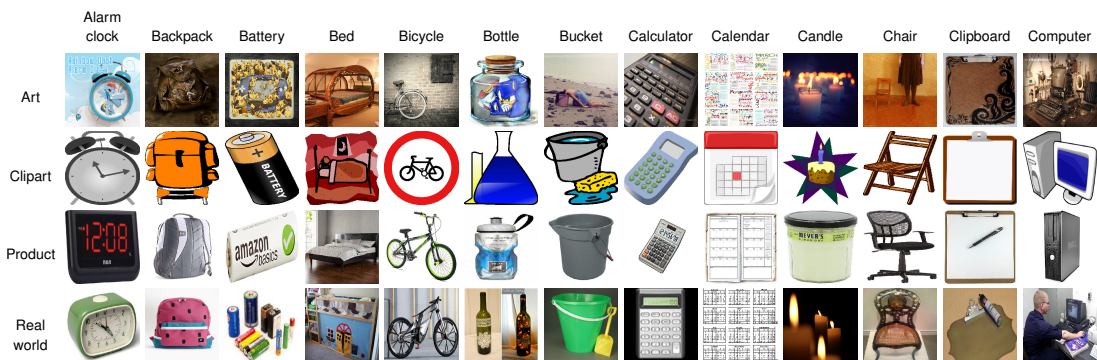
Office (Saenko et al., 2010) is a multiclass object recognition benchmark dataset, aggregating images of 31 categories of objects commonly found in office environments. It contains color images from three different domains: 2,817 images of products scraped from Amazon, 498 images acquired using a DSLR digital camera, and 795 images captured with a webcam. The images are of arbitrary sizes and aspect ratios.



The full list of object classes is:

backpack	bike	bike_helmet	bookcase	bottle	calculator	desk_chair
desk_lamp	desktop_computer	file_cabinet	headphones	keyboard	laptop_computer	letter_tray
mobile_phone	monitor	mouse	mug	paper_notebook	pen	phone
printer	projector	punchers	ring_binder	ruler	scissors	speaker
stapler	tape_dispenser	trash_can				

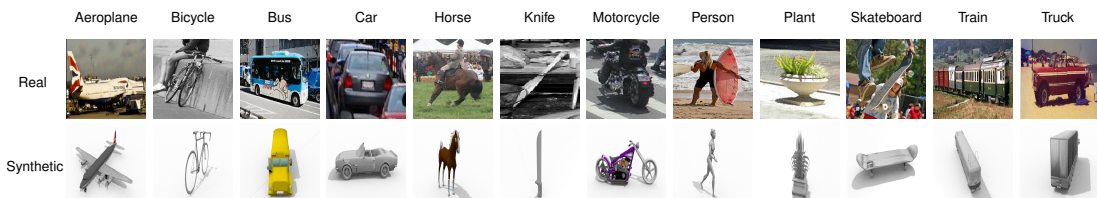
Office-Home (Venkateswara et al., 2017) comprises a larger corpus of color, arbitrarily-sized images from 65 different classes of objects commonly found in office and home environments, coming from four different domains. It contains 2,427 images extracted from paintings (Art), 4,365 clipart images (Clipart), 4,439 photographs of products (Product), and 4,357 pictures captured with a regular consumer camera (Real world).



The full list of object classes is:

Alarm_Clock	Candles	Exit_Sign	Helmet	Mouse	Printer	Sneakers	TV
Backpack	Chair	Fan	Kettle	Mug	Push_Pin	Soda	Webcam
Batteries	Clipboards	File_Cabinet	Keyboard	Notebook	Radio	Speaker	
Bed	Computer	Flipflops	Knives	Oven	Refrigerator	Spoon	
Bike	Couch	Flowers	Lamp_Shade	Pan	Ruler	Table	
Bottle	Curtains	Folder	Laptop	Paper_Clip	Scissors	Telephone	
Bucket	Desk_Lamp	Fork	Marker	Pen	Screwdriver	ToothBrush	
Calculator	Drill	Glasses	Monitor	Pencil	Shelf	Toys	
Calendar	Eraser	Hammer	Mop	Postit_Notes	Sink	Trash_Can	

VisDA 2017 (Peng et al., 2018) includes images of diverse sizes from 12 different categories, coming from two different domains: 55,368 synthetic grayscale renders of 3D models, and 152,397 photographs of the real-world objects. It is larger than the other two datasets, and exhibits a more significant domain shift.



The full list of object classes is

aeroplane bicycle bus car horse knife motorcycle person plant skateboard train truck

Chapter 5. Domain-adaptive multiflow networks

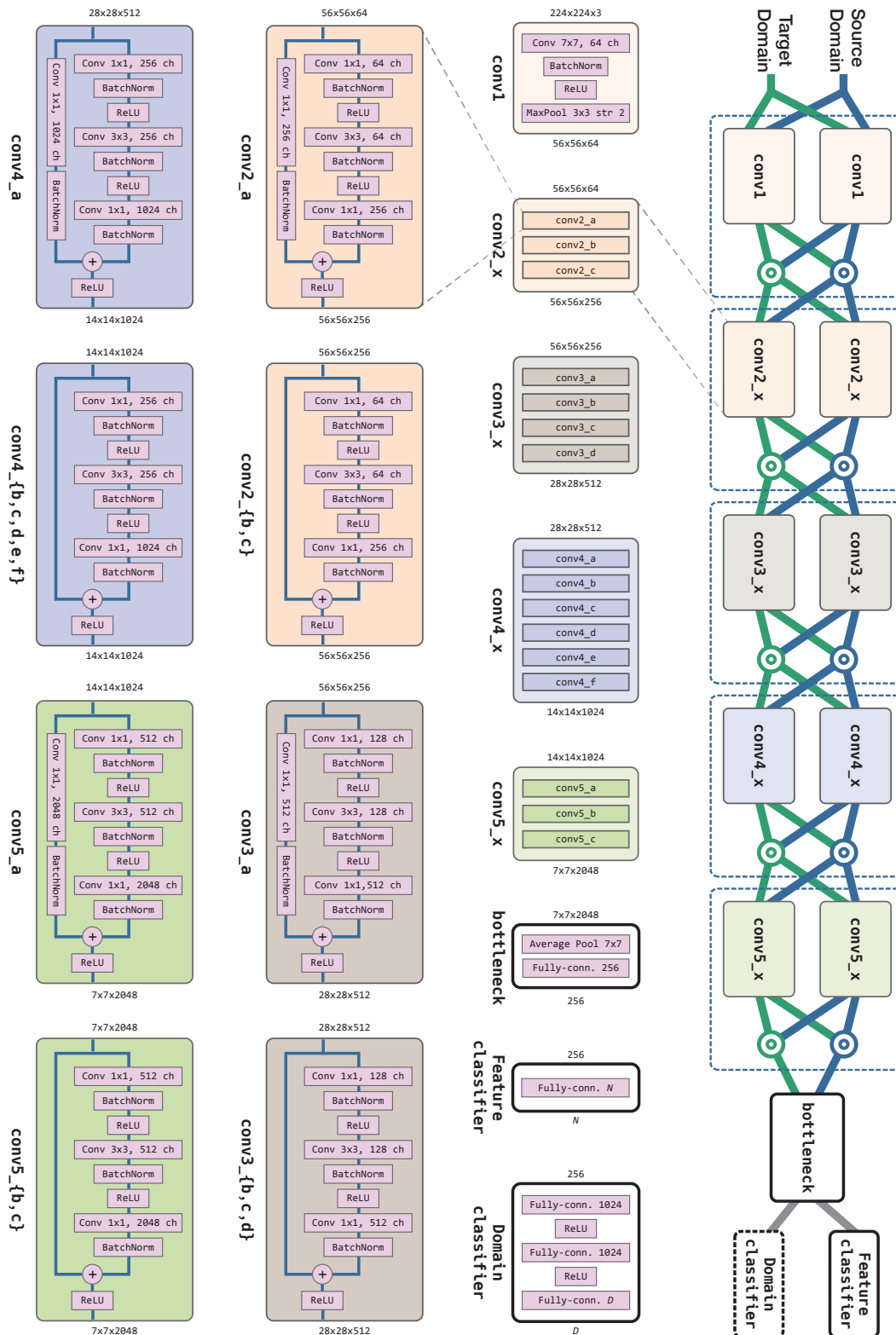


Figure 5.8 Multiflow ResNet-50. This architecture is adapted from the original ResNet-50 (He et al., 2016). We preserve the groupings described in the original paper (He et al., 2016). N denotes the number of classes in the dataset.

Setup

As backbone network to process all of the above datasets, we use a ResNet-50 (He et al., 2016), with the bottleneck layer modification of Rozantsev et al. (2019). While many multiflow configurations can be designed for such a deep network, we choose to make our gated computational units coincide with the layer groupings defined in He et al. (2016), namely conv1, conv2_x, conv3_x, conv4_x, and conv5_x. The resulting multiflow network is depicted by Figure 5.8. We feed our DAMNets images resized to 224×224 pixels, as expected by ResNet-50.

Results

The results for the three object recognition datasets are provided in the 2nd to 4th portions of Table 5.1. Our approach outperforms the baselines in *all* cases.

For the **Office** dataset, DAMNet achieves a noticeable improvement over the baselines in cases where the source domain is Amazon, which would typically correspond to a real application scenario, where one obtains labels from Amazon automatically. In cases where Amazon is the target domain, we still outperform the baselines, albeit by a smaller margin. We believe this to be due to the relatively small amount of labeled data in DSLR and Webcam. This is further evidenced by our larger improvement in the multi-source scenario, where DSLR and Webcam are both used as source domains. Note that the domain shift between those two datasets is small and the adaptation between them is essentially saturated, with all methods achieving equally good performance.

On **Office-Home**, the gap between DAMNet and the baselines is more consistent across the different pairs, thus reflecting the more balanced nature of the data. Note that, here, because of the relatively large number of classes, the overall performance is lower than on Office. Importantly, our results show that, by leveraging all synthetic domains to transfer to the real one, our approach reaches an accuracy virtually equal to that of using full supervision on the target domain.

The **VisDA17** dataset shows the largest visual shift among all datasets, which makes it remarkably difficult for domain adaptation. Our method is again the best performer on both adaptation scenarios, albeit by a smaller margin.

Gate dynamics

To understand the way our networks learn the domain-specific flow assignments, we track the state of the gates for all computational units over all training epochs. In Figure 5.9, we plot the corresponding evolution of the gate activations for the DSLR+Webcam→Amazon task on Office.

Chapter 5. Domain-adaptive multiframe networks

Table 5.1 Domain adaptation results. We compare the accuracy of our DAMNet approach with that of DANN (Ganin and Lempitsky, 2015) and of RPT (Rozantsev et al., 2019), for image classification tasks commonly used to evaluate domain adaptation methods. As illustrated for all datasets in Sections 5.3.3 to 5.3.5, different source and target domain combinations present various degrees of domain shift, and some combinations are clearly more challenging than others. Our DAMNets yield a significant accuracy boost in the presence of large domain shifts, particularly when using more than one source domain.

Datasets	Source(s) → Target	No DA	DANN	RPT	DAMNet	On TD
Digits	MNIST → MNIST-M	52.25	76.66 [†]	82.24	88.80	96.21
	SVHN → MNIST	54.90	73.90 [†]	78.70 [†]	81.30	99.26
	MNIST → SVHN	25.57	31.69	34.72	37.95	89.23
	MNIST-M → SVHN	27.49	37.43	37.90	39.41	89.23
	MNIST + MNIST-M → SVHN	33.52	44.16	<i>n/a</i>	51.83	89.23
	MNIST + MNIST-M → SVHN*	22.88	49.02	<i>n/a</i>	79.45	96.07
Office	Webcam → DSLR	93.60	99.20 [†]	99.40 [†]	99.62	95.20
	Amazon → DSLR	32.80	79.10 [†]	82.70 [†]	84.14	95.20
	DSLR → Webcam	90.45	97.70 [†]	98.00 [†]	98.11	98.49
	Amazon → Webcam	34.67	78.90 [†]	81.50 [†]	85.28	98.49
	Webcam → Amazon	41.42	62.80 [†]	63.60 [†]	65.67	85.11
	DSLR → Amazon	34.47	63.60 [†]	64.70 [†]	64.82	85.11
	DSLR + Webcam → Amazon	45.82	64.86	<i>n/a</i>	68.87	85.11
Office-Home	Art → Product	37.03	58.50	54.51	59.30	87.66
	Clipart → Product	36.67	70.50	63.18	77.50	87.66
	Clipart → Art	29.65	47.93	47.32	51.24	64.42
	Real world → Art	50.91	57.68	51.90	60.74	64.42
	Art → Real world	53.12	56.40	52.15	59.90	77.80
	Clipart → Real world	43.03	57.90	55.05	62.70	77.80
	Product → Real world	46.42	62.30	62.16	65.00	77.80
	Clipart + Product → Real world	53.39	70.53	<i>n/a</i>	72.25	77.80
Art + Clipart + Product → Real world	58.72	72.00	<i>n/a</i>	77.65	77.80	
VisDA 2017	Synthetic → Real	35.46	59.90	61.10	61.40	84.72
	Real → Synthetic	51.12	83.10	82.15	85.20	99.34
UAV-200	Synthetic → Real*	0.377	0.715	0.743	0.792	0.858

[†]Accuracy reported in Ganin and Lempitsky (2015) and Rozantsev et al. (2019)

*Evaluated with a ResNet-50

*Results reported as Average Precision

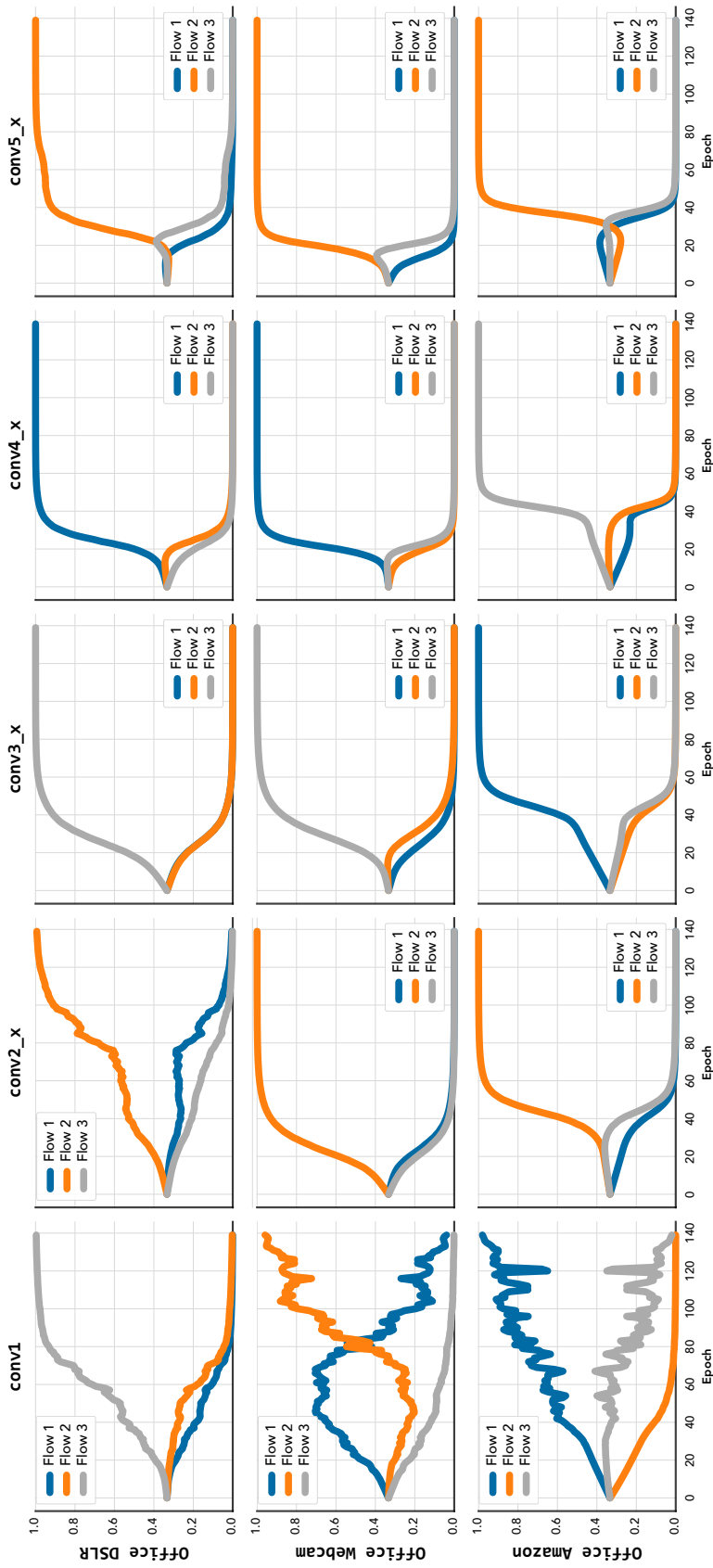


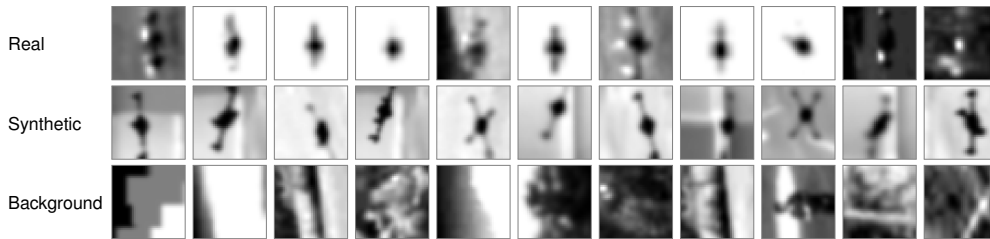
Figure 5.9 Evolution of the gates' activations for each of the computational units in a multiframe ResNet-50 network. These curves correspond to the Office DSLR + Webcam \rightarrow Amazon domain adaptation problem. In the top two rows, we show the gates for the source domains and in the bottom row for the target one. All flows are initialized to parameters obtained from a single ResNet-50 trained on ImageNet. Note how for the first computational unit, conv1, each domain chooses to process the data with different flows. In the remaining units, the two source domains, which have similar appearance, share all the computations. By contrast, the target domain still uses its own flows in conv3_x, and conv4_x to account for its significantly different appearance. When arriving at conv_5x, the data has been converted to a domain-agnostic representation, and hence the same flow can operate on all domains.

Chapter 5. Domain-adaptive multiframe networks

Note that our DAMNet leverages different flows over time for each domain before reaching a firm decision. Interestingly, we can see that, with the exception of the first unit, which performs low-level computations, DSLR and Webcam share all flows. By contrast, Amazon, which has a significantly different appearance, mostly uses its own flows, except in two computational units. This evidences that our network successfully understands when domains are similar and can thus use similar computations.

5.3.5 Object detection

We evaluate our method for the detection of drones from video frames, on the UAV-200 dataset (Rozantsev et al., 2018), which contains examples of drones both generated artificially and captured from real video footage. In particular, it aggregates 200 images of real drones and around 33,000 synthetic ones, as well as around 190,000 patches obtained from the background of the video, which do not contain drones, used as negative examples. All examples are of size 40×40 pixels. We evaluate performance on a validation set comprising 3,000 positive and 135,000 negative patches.



Setup

Our domain adaptation leverages both the synthetic examples of drones, as source domain, and the limited amount of annotated real drones, as target domain, as well as the background negative examples, to predict the class of patches from the validation set of real images. We follow closely the supervised setup and network architecture of Rozantsev et al. (2019), which performs object detection by classifying image patches as either containing a drone or not. As in Rozantsev et al. (2019), we use AdaDelta as optimizer, cross-entropy as loss function, and average precision as evaluation metric. Our multiframe computational units are defined as groupings of successive convolutions, non-linearities, and pooling operations. The details of the architecture are provided in Figure 5.10.

Results

Our method considerably surpasses all the others in terms of average precision, as shown on the last section of Table 5.1, thus validating DAMNets as effective models for leveraging synthetic data for domain adaptation in real-world problems.

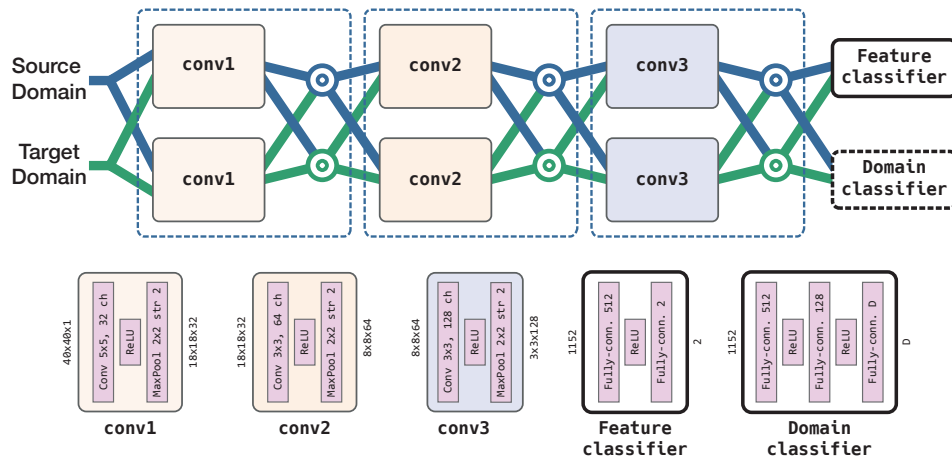


Figure 5.10 Multiflow architecture for drone detection. This architecture is a multiflow extension to the one used by Rozantsev et al. (2019).

5.4 Conclusion

In this chapter, we have introduced a domain adaptation approach that allows for adaptive, separate computations for different domains. Our framework relies on computational units that aggregate the outputs of multiple parallel operations, and on a set of trainable domain-specific gates that adapt the aggregation process to each domain. Our experiments have demonstrated the benefits of this approach over the state-of-the-art weight untying strategy; the greater flexibility of our method translates into a consistently higher accuracy.

We believe that the benefits of DAMNets come from three different qualities of our proposed paradigm. First, aside from the exceptional case of batch normalization, there are no private operations per domain; this drives the parameters to be learned from all available data, and as a result the operations are encouraged to be domain-agnostic if no other forces are at play. Second, the domain-specific learnable gates determine the specific combination of the above-mentioned operations, defining the actual capacity of the network per domain, which can reflect differences in domain complexities. Third, the resulting dynamic architecture avoids restricting all domains to share the same computational graph, which could include operations that are irrelevant or even harmful for the target domain; providing a mechanism that enables the training to conveniently ignore or downweight aspects of the source domain helps avoid negative transfer.

Adapting to new information is one of the major bottlenecks in ML. This thesis has introduced specific methods and general insights, in an effort to grant adaptability to ML models for visual applications. We quickly recapitulate the main ideas introduced in this thesis, and how they are used to tackle domain adaptation.

6.1 Leveraging semantics from structure

6.1.1 Visual correspondences

Finding the same structure between images in source and target domains imposes an annotated target domain set, where one can borrow the class identities from source to target. *Visual correspondences* are a relaxation of this. Structures that look similar across domains are candidates to be true semantic correspondences.

A Multiple-Instance Learning optimization accounts for the fact that such relaxation is imperfect. The optimization finds new, adapted parameters for a model pretrained on the source domain. Under the new parameters, the model adapts appropriately for the target domain. We show this by adapting Boosted Decision Tree classifiers. The results in Electron Microscopy segmentation showcase the value in exploiting structure, and serve as a conceptual base for further work.

6.1.2 Consensus heatmaps

Visual correspondences—which are locations on the target domain—often cluster around particular regions. The consensus heatmap tracks this behavior, by showing which target domain locations are commonly matched to foreground and background. Consensus heatmaps appropriately describe subcellular structures in Electron Microscopy images.

Extreme-scoring regions of the consensus heatmap often coincide with the expected segmentation map. Those regions can be used as a supervisory target domain signal. A first use of consensus heatmaps is to rank visual correspondences, which experimentally improves their reliability. In addition, thresholded heatmaps are used directly as partial annotations for the two-stream U-Net.

6.2 Learning semantic representations

6.2.1 Two-stream U-Net

The two-stream U-Net architecture is a deep network architecture composed of two semi-independent streams, one processing source domain images and one processing target domain images, loosely connected by their corresponding parameters and outputs. It is an adaptation of the popular U-Net for the domain adaptation scenario.

The two-stream U-Net encodes images at different levels of granularity in order to extract a semantic representation suitable for inference. Corresponding operations between streams are encouraged to be related via parameter co-regularization. Predictions are also enforced to preserve similarities across domains. The two-stream U-Net optimizes a custom loss function tailored for segmentation under class imbalance. Semantic segmentations of Electron Microscopy imaging obtained with the two-stream U-Net are comparable with segmentations by humans and do not require user annotations.

6.2.2 Multiflow networks

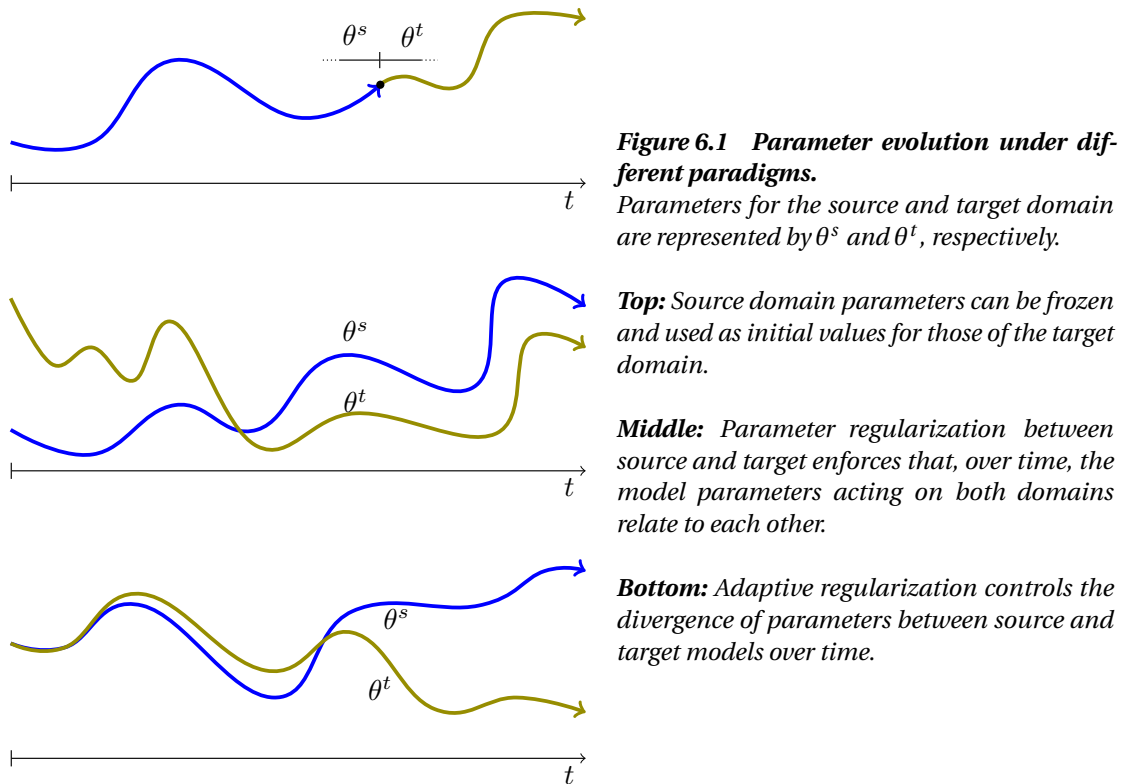
Multiflow networks define alternative computations—or flows—and combine them so as to obtain a latent representation of an image. In the multi-domain scenario, the same flows are used for each domain, but combined in a different way for each domain. The result is a network with operations with different degrees of sharing between different domains. This enables different domains to effectively use different network capacities.

The domain-adaptive aspect of the network comes from the training process. Initially, all flows are exposed to all data, and to all available supervision. Trainable gates, particular for each domain, select the relative contribution of each flow to the encoding. Over time, flows either specialize for one domain, or perform general computation to be shared between domains. Results in object classification and object detection highlight the benefits of domain adaptation with multiflow networks.

6.3 Some general insights

6.3.1 Model adaptation bias

In traditional ML, regularization techniques are commonly used to bias the solution of an ill-posed problem towards those that exhibit some desirable properties. The most common use of regularization is to combat overfitting. As domain adaptation is in general also an underconstrained problem, it benefits from such biases as well. In particular, as the name suggests, in domain adaptation there exists the notion of a reference problem—that on the source domain—which is subsequently *adapted* to the target domain.



The domain adaptation algorithms introduced in this thesis follow different approaches with regards to how the source domain influences adaptation. We distinguish the following three scenarios, depicted in Figure 6.1:

- The source domain model serves as *initial condition* for the target domain model.
- The parameters from the source domain model serve as a *reference* from which the target domain model is prevented from arbitrarily diverge.
- The amount of divergence between source and target domain parameters changes over time as optimization sees fit.

In the first scenario, the adaptation algorithm does not observe any source domain data, and the parameters pre-learned on the source domain are modified for the target domain. The method presented in Chapter 3—which adapts parameters in a Boosted Decision Tree model—operates under this setup. It is effective because the models are simple, and have few parameters; in addition, our formulation includes an additional regularizer, by treating the source domain parameters as priors to estimate the target domain ones.

In the second scenario, the effects of the source domain acting as a reference become more pronounced over time. The two-stream U-Net introduced in Chapter 4 works in this way. As the network learns on both domains, it enforces that the learned parameters on the target

domain follow a constrained transformation with respect to those on the source. In this case, again, we incorporate additional regularizations to help constraining the parameter tuning.

In the last case—which our multiflow networks of Chapter 5 follow—all parameters are influenced by data from both domains, to different extents. The supervised signal from the source domain always has some degree of influence on the learning of all parameters. Operations for the source domain, especially at the beginning of the training, are encouraged not only to perform the source domain task well, but also to be beneficial for the extraction of latent representations for the target domain.

Data availability, computational resources, and the expected degree of domain shift, are examples of criteria to use when deciding what model bias paradigm to use for a given domain adaptation problem. For example, when the supervisory signal from the source domain is no longer available, using the source domain parameters as initial conditions might be the only option. When the domain shift is small, sharing and regularizing weights might be enough. When multiple annotated domains are available, exposing all parameters to the available supervision is more robust.

On the other hand, the methods presented have clear limitations and should be used with caution. Removing the influence of the source domain, in the first case, is not enough signal to tune the sheer number of parameters in modern neural networks. Forcing the same architecture and related weights, as in the second case, may lead to negative transfer under large domains shifts. Providing an overparametrized network with alternative operations from which to choose, as in the third scenario, may be prohibitively expensive; deciding a particular flow assignment over other is, at the moment, not formally justified and might also affect the learning.

6.3.2 Negative transfer

The models introduced in this thesis are structurally different. The *boosted decision trees* model of Chapter 3 proposes a fixed architecture and adapts its parameters. The *two-stream U-Net* of Chapter 4 has independent streams for each domain, but the structure of those streams is identical. The *multiflow networks* of Chapter 5 result into architectures for source and target domains that are different altogether.

We believe that the success of the multiflow paradigm resides precisely in *not* adapting all aspects of the source domain problem to the target domain, but rather in selecting what computations to share and what to keep private. Acknowledging and embracing domain differences in this way helps avoiding negative transfer.

6.4 Future research directions

The findings presented in this thesis serve as starting point for exploration of new ideas related to domain adaptation.

The *domain-adaptive multiflow networks* of Chapter 5 can be seen as an implementation of Neural Architecture Search (NAS). The application of NAS to the domain adaptation problem is a largely unexplored area. As discussed in Section 6.3.2, domain adaptation benefits from using different computations per domain. Following more principled ways to construct dynamic architectures based on NAS, for example by combining them with the reinforcement learning formalism of Baker et al. (2017), is a promising way to promote domain adaptation.

As we saw in Section 6.3.1, the regularization coming from the network structure is determinant for domain adaptation. In multiflow networks, the resulting network architecture is dynamically determined. By enforcing the optimization to favor flows and combinations of flows with desirable properties—low architecture complexity or uncorrelated outputs, for example—the resulting model will perform smart model selection.

Another exciting avenue of research is the learning of hierarchical representations. Image contents are often hierarchically structured into components and subcomponents. We hypothesize that the multiflow formalism can be extended to consider this. In a hierarchical multiflow network, different flows can themselves be computational units, with their own domain-specific gates. Hierarchical multiflow networks could therefore arrive to the latent encoding by combining partial encodings. Relating partial encodings to partial semantics would allow to synthesize the latent representation in a way that considers image parts and their relationships (Lee and Seung, 1999; Felzenszwalb and Schwartz, 2007; Felzenszwalb et al., 2008; Collins et al., 2018; Gonzalez-Garcia et al., 2018). We believe that this decomposition of the image semantics can be beneficial for domain adaptation, as the model would be able to choose what partial semantics to share between domains.

Bibliography

- Ahmed, K. and Torresani, L. (2017). Connectivity Learning in Multi-Branch Networks. In *NIPS Meta-learning Workshop*. (Cited on page 58)
- Andrychowicz, M., Denil, M., Gómez, S., Hoffman, M., Pfau, D., Schaul, T., Shillingford, B., and de Freitas, N. (2016). Learning to learn by gradient descent by gradient descent. In *Advances in Neural Information Processing Systems*, pages 3981–3989. Curran Associates, Inc. (Cited on page 15)
- Anirudh, R., Thiagarajan, J., Bremer, T., and Kim, H. (2016). Lung nodule detection using 3D convolutional neural networks trained on weakly labeled data. In *Medical Imaging 2016: Computer-Aided Diagnosis*, volume 9785, page 978532. International Society for Optics and Photonics. (Cited on page 14)
- Bagautdinov, T., Alahi, A., Fleuret, E., Fua, P., and Savarese, S. (2017). Social Scene Understanding: End-To-End Multi-Person Action Localization and Collective Activity Recognition. In *Conference on Computer Vision and Pattern Recognition*. (Cited on page 9)
- Baker, B., Gupta, O., Naik, N., and Raskar, R. (2017). Designing Neural Network Architectures using Reinforcement Learning. *International Conference on Learning Representations*. (Cited on page 75)
- Baktashmotlagh, M., Harandi, M., Lovell, B., and Salzmann, M. (2013). Unsupervised Domain Adaptation by Domain Invariant Projection. In *International Conference on Computer Vision*, pages 769–776. (Cited on page 17)
- Becker, C., Ali, K., Knott, G., and Fua, P. (2012). Learning Context Cues for Synapse Segmentation in EM Volumes. In *Conference on Medical Image Computing and Computer Assisted Intervention*. (Cited on pages 26 and 46)
- Becker, C., Christoudias, M., and Fua, P. (2015). Domain Adaptation for Microscopy Imaging. *IEEE Transactions on Medical Imaging*, 34(5):1125–1139. (Cited on pages 46 and 47)
- Becker, C. J. (2016). *Towards Scalable Segmentation of 3D Image Stacks*. PhD thesis, EPFL, Lausanne, Switzerland. (Cited on page 26)

Bibliography

- Bermúdez-Chacón, R., Altingövde, O., Becker, C., Salzmann, M., and Fua, P. (2019). Visual Correspondences for Unsupervised Domain Adaptation on Electron Microscopy Images. *IEEE Transactions on Medical Imaging*, pages 1–1. (Cited on page 11)
- Bermúdez-Chacón, R., Becker, C., Salzmann, M., and Fua, P. (2016). Scalable Unsupervised Domain Adaptation for Electron Microscopy. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 326–334. (Cited on page 11)
- Bermúdez-Chacón, R., Márquez-Neila, P., Salzmann, M., and Fua, P. (2018). A Domain-Adaptive Two-Stream U-Net for Electron Microscopy Image Segmentation. In *International Symposium on Biomedical Imaging*, pages 400–404. (Cited on pages 11 and 42)
- Bermúdez-Chacón, R., Salzmann, M., and Fua, P. (2020). Domain-Adaptive Multibranch Networks. In *International Conference on Learning Representations*. (Cited on page 11)
- Bhatia, P., Arumae, K., and Celikkaya, E. B. (2019). Dynamic transfer learning for named entity recognition. In *International Workshop on Health Intelligence*, pages 69–81. Springer. (Cited on page 58)
- Bousmalis, K., Trigeorgis, G., Silberman, N., Krishnan, D., and Erhan, D. (2016). Domain Separation Networks. In *Advances in Neural Information Processing Systems*, pages 343–351. (Cited on page 17)
- Briechle, K. and Hanebeck, U. D. (2001). Template matching using fast normalized cross correlation. In *Optical Pattern Recognition XII*, volume 4387, pages 95–102. International Society for Optics and Photonics. (Cited on page 20)
- Campos, V., Jou, B., and i Nieto, X. G. (2017). From pixels to sentiment: Fine-tuning CNNs for visual sentiment prediction. *Image and Vision Computing*, 65:15–22. (Cited on page 13)
- Carbonneau, M., Cheplygina, V., Granger, E., and Gagnon, G. (2018). Multiple instance learning: A survey of problem characteristics and applications. *Pattern Recognition*, 77:329 – 353. (Cited on page 24)
- Chapelle, O., Scholkopf, B., and Zien, A., editors (2006). *Semi-Supervised Learning*. MIT Press. (Cited on page 13)
- Chen, M., Xu, Z., Weinberger, K., and Sha, F. (2012). Marginalized denoising autoencoders for domain adaptation. *arXiv preprint arXiv:1206.4683*. (Cited on page 17)
- Cireşan, D. C., Meier, U., and Schmidhuber, J. (2012). Transfer learning for Latin and Chinese characters with deep neural networks. In *International Joint Conference on Neural Networks*, pages 1–6. IEEE. (Cited on page 13)
- Clinchant, S., Csurka, G., and Chidlovskii, B. (2016). A Domain Adaptation Regularization for Denoising Autoencoders. In *Annual Meeting of the Association for Computational Linguistics*. (Cited on page 17)

- Collins, E., Achanta, R., and Süsstrunk, S. (2018). Deep Feature Factorization For Concept Discovery. In *European Conference on Computer Vision*. (Cited on page 75)
- Cordts, M., Omran, M., Ramos, S., Rehfeld, T., Enzweiler, M., Benenson, R., Franke, U., Roth, S., and Schiele, B. (2016). The Cityscapes Dataset for Semantic Urban Scene Understanding. In *Conference on Computer Vision and Pattern Recognition*. (Cited on page 8)
- Cozzolino, D., Thies, J., Rössler, A., Riess, C., Nießner, M., and Verdoliva, L. (2018). ForensicTransfer: Weakly-supervised domain adaptation for forgery detection. *arXiv preprint arXiv:1812.02510*. (Cited on page 9)
- Csurka, G. (2017). A comprehensive survey on domain adaptation for visual applications. In *Domain Adaptation in Computer Vision Applications*, pages 1–35. Springer. (Cited on page 16)
- Csurka, G., Baradel, F., Chidlovskii, B., and Clinchant, S. (2017a). Discrepancy-Based Networks for Unsupervised Domain Adaptation: A Comparative Study. In *International Conference on Computer Vision Workshops*. (Cited on page 17)
- Csurka, G., Chidlovski, B., Clinchant, S., and Michel, S. (2017b). An Extended Framework for Marginalized Domain Adaptation. *arXiv preprint arXiv:1702.05993*. (Cited on page 17)
- Dai, W., Yang, Q., Xue, G., and Yu, Y. (2007). Boosting for Transfer Learning. In *Machine Learning*, pages 193–200. (Cited on pages 16 and 25)
- Damianou, A., Ek, C., Titsias, M., and Lawrence, N. (2012). Manifold relevance determination. In *International Conference on Machine Learning*. (Cited on page 17)
- Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. (2009). Imagenet: A Large-Scale Hierarchical Image Database. In *Conference on Computer Vision and Pattern Recognition*. (Cited on page 59)
- Devlin, J., Chang, M., Lee, K., and Toutanova, K. (2018). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *arXiv preprint arXiv:1810.04805*. (Cited on page 13)
- Dietterich, T., Lathrop, R., and Lozano-Pérez, T. (1997). Solving the Multiple Instance Problem with Axis-Parallel Rectangles. *Artificial Intelligence*, 89. (Cited on page 24)
- Doersch, C. and Zisserman, A. (2017). Multi-Task Self-Supervised Visual Learning. In *International Conference on Computer Vision*. (Cited on page 15)
- Donahue, J., Hoffman, J., Rodner, E., Saenko, K., and Darrell, T. (2013). Semi-Supervised Domain Adaptation with Instance Constraints. In *Conference on Computer Vision and Pattern Recognition*, pages 668–675. (Cited on pages 16 and 17)
- Donahue, J., Jia, Y., Vinyals, O., Hoffman, J., Zhang, N., Tzeng, E., and Darrell, T. (2014). DeCAF: A Deep Convolutional Activation Feature for Generic Visual Recognition. In *International Conference on Machine Learning*, pages 647–655. (Cited on page 13)

Bibliography

- Dou, Q., Ouyang, C., Chen, C., Chen, H., Glocker, B., Zhuang, X., and Heng, P. (2018). PnP-AdaNet: Plug-and-play adversarial domain adaptation network with a benchmark at cross-modality cardiac segmentation. *arXiv preprint arXiv:1812.07907*. (Cited on page 7)
- Efron, B. (1982). *The Jackknife, the Bootstrap and other resampling plans*. (Cited on page 27)
- Eisenschlos, J., Ruder, S., Czapla, P., Kardas, M., Gugger, S., and Howard, J. (2019). MultiFiT: Efficient Multi-lingual Language Model Fine-tuning. (Cited on page 13)
- Elsken, T., Metzen, J., and Hutter, F. (2019). Neural Architecture Search: A Survey. *Journal of Machine Learning Research*, 20(55):1–21. (Cited on page 58)
- FarajiDavar, N., Campos, T. D., Kittler, J., and Yan, F. (2011). Transductive transfer learning for action recognition in tennis games. In *2011 IEEE International Conference on Computer Vision Workshops (ICCV Workshops)*, pages 1548–1553. IEEE. (Cited on page 9)
- Felzenszwalb, P., Mcallester, D., and Ramanan, D. (2008). A Discriminatively Trained, Multiscale, Deformable Part Model. In *Conference on Computer Vision and Pattern Recognition*. (Cited on page 75)
- Felzenszwalb, P. and Schwartz, J. (2007). Hierarchical Matching of Deformable Shapes. In *Conference on Computer Vision and Pattern Recognition*. (Cited on page 75)
- Feng, X., Yang, J., Laine, A., and Angelini, E. (2017). Discriminative localization in CNNs for weakly-supervised segmentation of pulmonary nodules. In *Conference on Medical Image Computing and Computer Assisted Intervention*, pages 568–576. Springer. (Cited on page 14)
- Fernando, B., Habrard, A., Sebban, M., and Tuytelaars, T. (2013). Unsupervised Visual Domain Adaptation Using Subspace Alignment. In *International Conference on Computer Vision*, pages 2960–2967. (Cited on pages 17, 29, 30, 31, and 46)
- Fernando, B., Tommasi, T., and Tuytelaars, T. (2015). Joint Cross-Domain Classification and Subspace Learning for Unsupervised Adaptation. *Pattern Recognition Letters*, 65:60–66. (Cited on page 17)
- Finn, C., Abbeel, P., and Levine, S. (2017). Model-Agnostic Meta-Learning for fast adaptation of deep networks. In *International Conference on Machine Learning*, pages 1126–1135. (Cited on page 15)
- Freund, Y. and Schapire, R. (1995). A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting. In *European Conference on Computational Learning Theory*, pages 23–37. (Cited on page 16)
- Friedman, J. (2002). Stochastic Gradient Boosting. *Computational Statistics & Data Analysis*, 38(4):367–378. (Cited on page 27)

- Gala, R., Chapeton, J., Jitesh, J., Bhavsar, C., and Stepanyants, A. (2014). Active Learning of Neuron Morphology for Accurate Automated Tracing of Neurites. *Frontiers in neuroanatomy*, 8:37. (Cited on page 15)
- Ganin, Y. and Lempitsky, V. (2015). Unsupervised Domain Adaptation by Backpropagation. In *International Conference on Machine Learning*, pages 1180–1189. (Cited on pages 18, 57, 58, 59, 60, 61, and 66)
- Garcke, J. and Vanck, T. (2014). Importance weighted inductive transfer learning for regression. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 466–481. Springer. (Cited on page 16)
- Gilbert, D. (1828). XXVI. On the regular or platonic solids. *The Philosophical Magazine*, 3(15):161–165. (Cited on page 21)
- Girshick, R. B. (2015). Fast R-CNN. In *International Conference on Computer Vision*. (Cited on page 13)
- Gong, B., Shi, Y., Sha, F., and Grauman, K. (2012). Geodesic Flow Kernel for Unsupervised Domain Adaptation. In *Conference on Computer Vision and Pattern Recognition*, pages 2066–2073. (Cited on page 17)
- Gonzalez-Garcia, A., Modolo, D., and Ferrari, V. (2018). Do semantic parts emerge in convolutional neural networks? *International Journal of Computer Vision*, 126(5):476–494. (Cited on page 75)
- Gopalan, R., Li, R., and Chellappa, R. (2011). Domain Adaptation for Object Recognition: An Unsupervised Approach. In *International Conference on Computer Vision*, pages 999–1006. (Cited on page 17)
- Graves, A. (2016). Adaptive Computation Time for Recurrent Neural Networks. In *arXiv Preprint*. (Cited on page 58)
- Gretton, A., Borgwardt, K., Rasch, M., Schölkopf, B., and Smola, A. (2007). A Kernel Method for the Two-Sample Problem. In *Advances in Neural Information Processing Systems*, pages 513–520. (Cited on page 43)
- Gretton, A., Smola, A., Huang, J., Schmittfull, M., Borgwardt, K., and Schölkopf, B. (2009). Covariate Shift by Kernel Mean Matching. *Journal of the Royal Statistical Society*, 3(4):5–13. (Cited on page 16)
- Guerrero, R., Ledig, C., and Rueckert, D. (2014). Manifold alignment and transfer learning for classification of alzheimer’s disease. In *International Workshop on Machine Learning in Medical Imaging*, pages 77–84. Springer. (Cited on page 16)
- Ha, D., Dai, A., and Le, Q. (2017). Hypernetworks. In *International Conference on Learning Representations*. (Cited on page 15)

Bibliography

- Habrard, A., Peyrache, J.-P., and Sebban, M. (2013). Boosting for Unsupervised Domain Adaptation. In *Machine Learning and Knowledge Discovery in Databases*, pages 433–448. Springer. (Cited on page 16)
- Häusser, P., Frerix, T., Mordvintsev, A., and Cremers, D. (2017a). Associative Domain Adaptation. In *International Conference on Computer Vision*, pages 2784–2792. (Cited on page 18)
- Häusser, P., Mordvintsev, A., and Cremers, D. (2017b). Learning by association—a versatile semi-supervised training method for neural networks. In *Conference on Computer Vision and Pattern Recognition*, pages 89–98. (Cited on pages 14 and 18)
- He, H., Yang, D., Wang, S., Wang, S., and Li, Y. (2019). Road Extraction by Using Atrous Spatial Pyramid Pooling Integrated Encoder-Decoder Network and Structural Similarity Loss. *Remote Sensing*, 11(9):1015. (Cited on page 39)
- He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep Residual Learning for Image Recognition. In *Conference on Computer Vision and Pattern Recognition*, pages 770–778. (Cited on pages 64 and 65)
- Heimann, T., Mounthey, P., John, M., and Ionasec, R. (2013). Learning Without Labeling: Domain Adaptation for Ultrasound Transducer Localization. *MICCAI*. (Cited on page 16)
- Hinterberger, T., Weiskopf, N., Veit, R., Wilhelm, B., Betta, E., and Birbaumer, N. (2004). An EEG-driven brain-computer interface combined with functional magnetic resonance imaging (fMRI). *IEEE Transactions on Biomedical Engineering*, 51(6):971–974. (Cited on page 7)
- Hoffman, J., Guadarrama, S., Tzeng, E., Hu, R., Donahue, J., Girshick, R., Darrell, T., and Saenko, K. (2014). LSDA: Large scale detection through adaptation. In *Advances in Neural Information Processing Systems*. (Cited on page 15)
- Hoffman, J., Tzeng, E., Donahue, J., Jia, Y., Saenko, K., and Darrell, T. (2013). One-shot adaptation of supervised deep convolutional models. *arXiv preprint arXiv:1312.6204*. (Cited on page 13)
- Hoffman, J., Tzeng, E., Park, T., Zhu, J., Isola, P., Saenko, K., Efros, A., and Darrell, T. (2018). CyCADA: Cycle Consistent Adversarial Domain Adaptation. In *International Conference on Machine Learning*, pages 1989–1998. (Cited on page 18)
- Hu, Y., Modat, M., Gibson, E., Li, W., Ghavami, N., Bonmati, E., Wang, G., Bandula, S., Moore, C., Emberton, M., Ourselin, S., Noble, J., Barratt, D., and Vercauteren, T. (2018). Weakly-Supervised Convolutional Neural Networks for Multimodal Image Registration. *Medical Image Analysis*, 49. (Cited on page 14)
- Huber, S., Priebe, J., Baumann, K., Plidschun, A., Schiessl, C., and Tölle, T. (2017). Treatment of Low Back Pain with a Digital Multidisciplinary Pain Treatment App: Short-Term Results. *Journal of Medical Internet Research*, 4(2):e11. (Cited on page 9)

- Huh, M., Liu, A., Owens, A., and Efros, A. (2018). Fighting fake news: Image splice detection via learned self-consistency. In *European Conference on Computer Vision*, pages 101–117. (Cited on page 9)
- Jamal, A., Namboodiri, V., Deodhare, D., and Venkatesh, K. (2018). Deep domain adaptation in action space. In *British Machine Vision Conference*, page 264. (Cited on page 9)
- Javanmardi, M. and Tasdizen, T. (2018). Domain adaptation for biomedical image segmentation using adversarial training. In *International Symposium on Biomedical Imaging*, pages 554–558. (Cited on pages 46 and 47)
- Jiangfan, H., Ping, L., and Xiaogang, W. (2019). Deep Self-Learning From Noisy Labels. (Cited on page 14)
- Kan, M., Shan, S., and Chen, X. (2015). Bi-Shifting Auto-Encoder for Unsupervised Domain Adaptation. In *International Conference on Computer Vision*, pages 3846–3854. (Cited on page 17)
- Kingma, D. P. and Ba, J. (2015). Adam: A Method for Stochastic Optimisation. In *International Conference on Learning Representations*. (Cited on page 45)
- Kornblith, S., Shlens, J., and Le, Q. V. (2019). Do better ImageNet models transfer better? In *Conference on Computer Vision and Pattern Recognition*. (Cited on page 13)
- Koziński, M., Mosińska, A., Salzmann, M., and Fua, P. (2018). Learning to Segment 3D Linear Structures Using Only 2D Annotations. In *Conference on Medical Image Computing and Computer Assisted Intervention*, pages 283–291. (Cited on page 14)
- Kuss, M. and Graepel, T. (2003). The geometry of kernel canonical correlation analysis. Technical report, Max Planck Institute for Biological Cybernetics. (Cited on page 17)
- LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998). Gradient-Based Learning Applied to Document Recognition. *Proceedings of the IEEE*, pages 2278–2324. (Cited on page 60)
- Lee, D. and Seung, H. (1999). Learning the parts of objects by non-negative matrix factorization. *Nature*, 401(6755):788. (Cited on page 75)
- Lewis, J. (1995). Fast Normalized Cross-Correlation. In *Vision Interface*, pages 120–123. (Cited on page 20)
- Lin, T.-Y., Goyal, P., Girshick, R. B., He, K., and Dollár, P. (2017). Focal Loss for Dense Object Detection. In *International Conference on Computer Vision*. (Cited on page 13)
- Liu, C., Zoph, B., Shlens, J., Hua, W., Li, L., Fei-Fei, L., Yuille, A., Huang, J., and Murphy, K. (2018). Progressive Neural Architecture Search. In *European Conference on Computer Vision*. (Cited on pages 55 and 58)

Bibliography

- Liu, H., Simonyan, K., and Yang, Y. (2019). DARTS Differentiable Architecture Search. In *International Conference on Learning Representations*. (Cited on page 58)
- Long, M., Cao, Y., Wang, J., and Jordan, M. I. (2015). Learning Transferable Features with Deep Adaptation Networks. In *International Conference on Machine Learning*, pages 97–105. (Cited on page 17)
- Mahapatra, D., Vezhnevets, A., Schüffler, P. J., Tielbeek, J. A. W., Vos, F. M., and Buhmann, J. M. (2013). Weakly supervised semantic segmentation of Crohn’s disease tissues from abdominal MRI. In *International Symposium on Biomedical Imaging*, pages 844–847. (Cited on page 14)
- McDonald, J. and Sheehan, F. (2004). Ventriculogram segmentation using boosted decision trees. In *Medical Imaging 2004: Image Processing*, volume 5370, pages 1804 – 1814. International Society for Optics and Photonics, SPIE. (Cited on page 25)
- Memisevic, R., Sigal, L., and Fleet, D. (2011). Shared kernel information embedding for discriminative inference. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(4):778–790. (Cited on page 17)
- Mosińska, A., Marquez-Neila, P., Kozinski, M., and Fua, P. (2018). Beyond the Pixel-Wise Loss for Topology-Aware Delineation. In *Conference on Computer Vision and Pattern Recognition*, pages 3136–3145. (Cited on page 39)
- Mosińska, A., Tarnawski, K., and Fua, P. (2017). Active Learning and Proofreading for Delineation of Curvilinear Structures. In *Conference on Medical Image Computing and Computer Assisted Intervention*, pages 165–173. (Cited on page 15)
- Nagabandi, A., Kahn, G., Fearing, R. S., and Levine, S. (2018). Neural network dynamics for model-based deep reinforcement learning with model-free fine-tuning. In *International Conference on Robotics and Automation*, pages 7559–7566. (Cited on page 13)
- Netzer, Y., Wang, T., Coates, A., Bissacco, A., Wu, B., and Ng, A. (2011). Reading Digits in Natural Images with Unsupervised Feature Learning. In *Advances in Neural Information Processing Systems*. (Cited on page 60)
- Noy, A., Nayman, N., Ridnik, T., Zamir, N., Doveh, S., Friedman, T., Giryes, R., and Zelnik-Manor, L. (2019). ASAP: Architecture search, anneal and prune. *arXiv preprint arXiv:1904.04123*. (Cited on page 58)
- Oguz, B., Shinohara, R., Yushkevich, P., and Oguz, I. (2017). Gradient Boosted Trees for Corrective Learning. In *International Workshop on Machine Learning in Medical Imaging*, pages 203–211. Springer. (Cited on page 26)
- Pan, S., Tsang, I., Kwok, J., and Yang, Q. (2010). Domain adaptation via transfer component analysis. *IEEE Transactions on Neural Networks*, 22(2):199–210. (Cited on page 17)

- Pan, S., Zheng, V., Yang, Q., and Hu, D. (2008). Transfer Learning for Wifi-Based Indoor Localization. In *Association for the Advancement of Artificial Intelligence Workshop*, page 6. (Cited on pages 16 and 17)
- Peng, X., Usman, B., Kaushik, N., Wang, D., Hoffman, J., and Saenko, K. (2018). VisDA: A synthetic-to-real benchmark for visual domain adaptation. *Conference on Computer Vision and Pattern Recognition Workshops*. (Cited on page 63)
- Pham, H., Guan, M., Zoph, B., Le, Q., and Dean, J. (2018). Efficient Neural Architecture Search via Parameter Sharing. In *International Conference on Machine Learning*. (Cited on pages 55 and 58)
- Pinheiro, P. and Collobert, R. (2015). From Image-Level to Pixel-Level Labeling with Convolutional Networks. In *Conference on Computer Vision and Pattern Recognition*, pages 1713–1721. (Cited on page 25)
- Rad, M., Oberweger, M., and Lepetit, V. (2018). Domain transfer for 3D pose estimation from color images without manual annotations. In *Asian Conference on Computer Vision*, pages 69–84. Springer. (Cited on page 14)
- Rajchl, M., Lee, M., Oktay, O., Kamnitsas, K., Passerat-Palmbach, J., Bai, W., Damodaram, M., Rutherford, M., Hajnal, J., Kainz, B., and Rueckert, D. (2017). Deepcut: Object segmentation from bounding box annotations using convolutional neural networks. *IEEE Trans. Med. Imaging*, 36(2):674–683. (Cited on page 14)
- Ratsaby, J. and Venkatesh, S. (1995). Learning from a mixture of labeled and unlabeled examples with parametric side information. In *Proceedings of the eighth annual conference on Computational learning theory*, pages 412–417. ACM. (Cited on page 14)
- Real, E., Aggarwal, A., Huang, Y., and Le, Q. V. (2019). Regularized evolution for image classifier architecture search. In *American Association for Artificial Intelligence Conference*, volume 33, pages 4780–4789. (Cited on page 58)
- Rebuffi, S., Bilen, H., and Vedaldi, A. (2017). Learning multiple visual domains with residual adapters. In *Advances in Neural Information Processing Systems*. (Cited on page 15)
- Richter, S., Vineet, V., Roth, S., and Koltun, V. (2016). Playing for data: Ground truth from computer games. In Leibe, B., Matas, J., Sebe, N., and Welling, M., editors, *European Conference on Computer Vision*, volume 9906 of *LNCS*, pages 102–118. Springer International Publishing. (Cited on page 8)
- Ronneberger, O., Fischer, P., and Brox, T. (2015). U-Net: Convolutional Networks for Biomedical Image Segmentation. In *Conference on Medical Image Computing and Computer Assisted Intervention*, pages 234–241. (Cited on pages 39, 41, 45, 46, and 47)
- Rössler, A., Cozzolino, D., Verdoliva, L., Riess, C., Thies, J., and Nießner, M. (2019). Faceforensics++: Learning to detect manipulated facial images. *arXiv preprint arXiv:1901.08971*. (Cited on page 9)

Bibliography

- Rozantsev, A., Salzman, M., and Fua, P. (2018). Residual Parameter Transfer for Deep Domain Adaptation. In *Conference on Computer Vision and Pattern Recognition*, pages 4339–4348. (Cited on pages 17, 18, 52, and 68)
- Rozantsev, A., Salzman, M., and Fua, P. (2019). Beyond Sharing Weights for Deep Domain Adaptation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41(4):801–814. (Cited on pages 14, 18, 41, 52, 55, 59, 60, 62, 65, 66, 68, and 69)
- Saenko, K., Kulis, B., Fritz, M., and Darrell, T. (2010). Adapting Visual Category Models to New Domains. In *European Conference on Computer Vision*, pages 213–226. (Cited on pages 16 and 62)
- Saito, K., Ushiku, Y., and Harada, T. (2017). Asymmetric Tri-Training for Unsupervised Domain Adaptation. In *International Conference on Machine Learning*. (Cited on page 18)
- Saito, K., Watanabe, K., Ushiku, Y., and Harada, T. (2018). Maximum Classifier Discrepancy for Unsupervised Domain Adaptation. In *Conference on Computer Vision and Pattern Recognition*. (Cited on page 18)
- Santoro, A., Bartunov, S., Botvinick, M., Wierstra, D., and Lillicrap, T. (2016). Meta-Learning with Memory-Augmented Neural Networks. In *Proceedings of The 33rd International Conference on Machine Learning*, pages 1842–1850. (Cited on page 15)
- Sejdinovic, D., Sriperumbudur, B., Gretton, A., and Fukumizu, K. (2013). Equivalence of distance-based and RKHS-based statistics in hypothesis testing. *The Annals of Statistics*, 41(5):2263–2291. (Cited on page 17)
- Settles, B. (2010). Active Learning Literature Survey. Technical report, University of Wisconsin–Madison. (Cited on page 15)
- Shazeer, N., Mirhoseini, A., Maziarz, K., Davis, A., Le, Q., Hinton, G., and Dean, J. (2017). Outrageously Large Neural Networks: The Sparsely-Gated Mixture-Of-Experts Layer. In *International Conference on Learning Representations*. (Cited on page 58)
- Shen, Y. (2005). *Loss Functions for Binary Classification and Class Probability Estimation*. PhD thesis, University of Pennsylvania. (Cited on page 43)
- Shu, R., Bui, H., Narui, H., and Ermon, S. (2018). A DIRT-T approach to unsupervised domain adaptation. In *International Conference on Learning Representations*. (Cited on page 18)
- Snell, J., Swersky, K., and Zemel, R. (2017). Prototypical networks for few-shot learning. In *Advances in Neural Information Processing Systems*, pages 4077–4087. (Cited on page 15)
- Sun, B., Feng, J., and Saenko, K. (2016). Correlation Alignment for Unsupervised Domain Adaptation. *arXiv Preprint*. (Cited on pages 17, 43, and 46)
- Sun, B. and Saenko, K. (2016). Deep CORAL: Correlation Alignment for Deep Domain Adaptation. In *European Conference on Computer Vision*, pages 443–450. (Cited on page 17)

- Suri, J. (2000). Computer vision, pattern recognition and image processing in left ventricle segmentation: The last 50 years. *Pattern Analysis & Applications*, 3(3):209–242. (Cited on page 7)
- Tajbakhsh, N., Shin, J., Gurudu, S., Hurst, R., Kendall, C., Gotway, M., and Liang, J. (2016). Convolutional neural networks for medical image analysis: Full training or fine tuning? *IEEE Transactions on Medical Imaging*, 35(5):1299–1312. (Cited on page 13)
- Tekin, B., Marquez-Neila, P., Salzmann, M., and Fua, P. (2017). Learning to Fuse 2D and 3D Image Cues for Monocular Body Pose Estimation. In *International Conference on Computer Vision*. (Cited on page 14)
- Top, A., Hamarneh, G., and Abugharbieh, R. (2011). Active Learning for Interactive 3D Image Segmentation. *Conference on Medical Image Computing and Computer Assisted Intervention*. (Cited on page 15)
- Tran, P. (2019). Semi-Supervised Learning with Self-Supervised Networks. *arXiv preprint arXiv:1906.10343*. (Cited on page 14)
- Tsai, D.-M. and Lin, C.-T. (2003). Fast normalized cross correlation for defect detection. *Pattern Recognition Letters*, 24(15):2625–2631. (Cited on page 20)
- Tsai, Y., Hung, W., Schuler, S., Sohn, K., Yang, M., and Chandraker, M. (2018). Learning to Adapt Structured Output Space for Semantic Segmentation. *arXiv preprint arXiv:1802.10349*. (Cited on page 18)
- Tzeng, E., Hoffman, J., Saenko, K., and Darrell, T. (2017). Adversarial Discriminative Domain Adaptation. In *Conference on Computer Vision and Pattern Recognition*, pages 7167–7176. (Cited on pages 18 and 55)
- Tzeng, E., Hoffman, J., Zhang, N., Saenko, K., and Darrell, T. (2014). Deep Domain Confusion: Maximizing for Domain Invariance. In *arXiv Preprint*. (Cited on pages 17, 46, and 47)
- VAY Sports (2019). VAY Fitness Coach App (<https://vay-sports.com>). (Cited on page 9)
- Veit, A. and Belongie, S. (2018). Convolutional Networks with Adaptive Inference Graphs. In *European Conference on Computer Vision*, pages 3–18. (Cited on page 58)
- Venkateswara, H., Eusebio, J., Chakraborty, S., and Panchanathan, S. (2017). Deep Hashing Network for Unsupervised Domain Adaptation. *Conference on Computer Vision and Pattern Recognition*, pages 5018–5027. (Cited on page 63)
- Vijayanarasimhan, S. and Grauman, K. (2011). Large-Scale Live Active Learning: Training Object Detectors with Crawled Data and Crowds. In *Conference on Computer Vision and Pattern Recognition*. (Cited on page 15)

Bibliography

- Vijayanarasimhan, S. and Grauman, K. (2014). Large-Scale Live Active Learning: Training Object Detectors with Crawled Data and Crowds. *International Journal of Computer Vision*, 108:97–114. (Cited on page 15)
- Vinyals, O., Blundell, C., Lillicrap, T., Kavukcuoglu, K., and Wierstra, D. (2016). Matching Networks for One Shot Learning. In *Advances in Neural Information Processing Systems*, pages 3637–3645, USA. Curran Associates Inc. (Cited on page 15)
- Wang, C. and Mahadevan, S. (2011). Heterogeneous Domain Adaptation Using Manifold Alignment. In *International Joint Conference on Artificial Intelligence*. (Cited on page 17)
- Wang, M. and Deng, W. (2018). Deep Visual Domain Adaptation: A Survey. *Neurocomputing*, 312:135 – 153. (Cited on page 16)
- Wang, W., Yu, K., Hugonot, J., Fua, P., and Salzmann, M. (2019). Recurrent U-Net for Resource-Constrained Segmentation. *arXiv preprint arXiv:1906.04913*. (Cited on page 39)
- Wang, X., Chen, H., Gan, C., Lin, H., Dou, Q., Cai, Q. H. M., and Heng, P. (2018). Weakly Supervised Learning for Whole Slide Lung Cancer Image Classification. *Medical Imaging with Deep Learning*. (Cited on page 14)
- Wu, Z., Nagarajan, T., Kumar, A., Rennie, S., Davis, L., Grauman, K., and Feris, R. (2018). Block-drop: Dynamic Inference Paths in Residual Networks. In *Conference on Computer Vision and Pattern Recognition*, pages 8817–8826. (Cited on page 58)
- Xu, Y., Zhang, J., Eric, I., Chang, C., Lai, M., and Tu, Z. (2012a). Context-Constrained Multiple Instance Learning for Histopathology Image Segmentation. In *Conference on Medical Image Computing and Computer Assisted Intervention*, pages 623–630. Springer. (Cited on page 14)
- Xu, Y., Zhu, J., Chang, E., Lai, M., and Tu, Z. (2014). Weakly supervised histopathology cancer image segmentation and classification. *Medical Image Analysis*, 18(3):591 – 604. (Cited on page 14)
- Xu, Y., Zhu, J.-Y., Chang, E., and Tu, Z. (2012b). Multiple Clustered Instance Learning for Histopathology Cancer Image Classification, Segmentation and Clustering. In *Conference on Computer Vision and Pattern Recognition*, pages 964–971. (Cited on page 14)
- Yanai, K. and Kawano, Y. (2015). Food image recognition using deep convolutional network with pre-training and fine-tuning. In *2015 IEEE International Conference on Multimedia & Expo Workshops (ICMEW)*, pages 1–6. IEEE. (Cited on page 13)
- Yang, L., Zhang, Y., Chen, J., Zhang, S., and Chen, D. (2017). Suggestive annotation: A deep active learning framework for biomedical image segmentation. In *Conference on Medical Image Computing and Computer Assisted Intervention*, pages 399–407. Springer. (Cited on page 15)

- Yao, T., Pan, Y., Ngo, C., Li, H., and Mei, T. (2015). Semi-supervised Domain Adaptation with Subspace Learning for visual recognition. In *Conference on Computer Vision and Pattern Recognition*. (Cited on page 17)
- Yosinski, J., Clune, J., Bengio, Y., and Lipson, H. (2014). How transferable are features in deep neural networks? In *Advances in Neural Information Processing Systems*, pages 3320–3328. (Cited on page 13)
- Zeng, Z., Xiao, S., Jia, K., Chan, T., Gao, S., Xu, D., and Ma, Y. (2013). Learning by associating ambiguously labeled images. In *Conference on Computer Vision and Pattern Recognition*, pages 708–715. (Cited on page 14)
- Zhai, D., Li, B., Chang, H., Shan, S., Chen, Y., and Gao, W. (2010). Manifold Alignment via Corresponding Projections. In *British Machine Vision Conference*. (Cited on page 16)
- Zhang, J., Ding, Z., Li, W., and Ogunbona, P. (2018a). Importance Weighted Adversarial Nets for Partial Domain Adaptation. In *Conference on Computer Vision and Pattern Recognition*. (Cited on page 16)
- Zhang, Y. and Yang, Q. (2017). A survey on multi-task learning. *arXiv preprint arXiv:1707.08114*. (Cited on page 15)
- Zhang, Z., Yang, L., and Zheng, Y. (2018b). Translating and segmenting multimodal medical volumes with cycle-and shape-consistency generative adversarial network. In *Conference on Computer Vision and Pattern Recognition*, pages 9242–9251. (Cited on page 3)
- Zhao, F., Huang, Q., and Gao, W. (2006). Image matching by normalized cross-correlation. In *2006 IEEE International Conference on Acoustics Speech and Signal Processing Proceedings*, volume 2, pages II–II. IEEE. (Cited on page 20)
- Zheng, J., Liu, M., Chellappa, R., and Phillips, P. (2012). A Grassmann manifold-based domain adaptation approach. In *International Conference on Pattern Recognition*, pages 2095–2099. IEEE. (Cited on page 17)
- Zhou, D., Bousquet, O., Lal, T. N., Weston, J., and Schölkopf, B. (2004). Learning with Local and Global Consistency. In *Advances in Neural Information Processing Systems*, pages 321–328. (Cited on page 13)
- Zhou, Z. (2017). A brief introduction to weakly supervised learning. *National Science Review*, 5(1):44–53. (Cited on page 14)
- Zhou, Z., Shin, J., Zhang, L., Gurudu, S., Gotway, M., and Liang, J. (2017). Fine-tuning convolutional neural networks for biomedical image analysis: actively and incrementally. In *Conference on Computer Vision and Pattern Recognition*, pages 7340–7351. (Cited on pages 13 and 15)

Bibliography

- Zhu, W., Lou, Q., Vang, Y., and Xie, X. (2017). Deep Multi-instance Networks with Sparse Label Assignment for Whole Mammogram Classification. In *Conference on Medical Image Computing and Computer Assisted Intervention*, pages 603–611, Cham. Springer International Publishing. (Cited on page 14)
- Zhu, X. and Ghahramani, Z. (2002). Learning from Labeled and Unlabeled Data with Label Propagation. Technical report, Carnegie Mellon University. (Cited on page 13)
- Zoph, B. and Le, Q. (2017). Neural Architecture Search with Reinforcement Learning. In *International Conference on Learning Representations*. (Cited on pages 55 and 58)
- Zoph, B., Vasudevan, V., Shlens, J., and Le, Q. (2018). Learning transferable architectures for scalable image recognition. In *Conference on Computer Vision and Pattern Recognition*, pages 8697–8710. (Cited on page 58)

Róger Bermúdez-Chacón

Doctoral assistant - Computer Vision Laboratory
EPFL - Lausanne, Switzerland

Rue du Maupas 26
1004 Lausanne
Switzerland
+44 7734 408803
rogerberm@gmail.com
[linkedin.com/in/rogerberm](https://www.linkedin.com/in/rogerberm)

Personal information	<i>Date of birth</i> Dec. 16, 1982	<i>Nationality</i> Costa Rican	<i>Marital status</i> Single
Education	École Polytechnique Fédérale de Lausanne (EPFL) • Lausanne, Switzerland Doctoral assistant - Computer Vision / Biomedical Imaging Thesis supervisor: Prof. Pascal Fua		2014 – 2019
	Eidgenössische Technische Hochschule Zürich (ETH Zurich) • Zurich, Switzerland MSc. Computational Biology and Bioinformatics		2011 – 2014
	Instituto Tecnológico de Costa Rica (ITCR) • Cartago, Costa Rica BSc. Computer Science		2000 – 2004
Work experience	Grupo Babel Software & IT Services • Heredia, Costa Rica		Dec 2009 – Aug 2011
	Technical Lead <i>Technical lead and software architect. ☆ Lead the team that developed the business intelligence solution for costarican National Insurance Institute (INS). Developed SQL Server Reporting Services query designer extension to simplify arbitrary report generation for non-expert users.</i>		
	SoftwareFX, Inc. • San José, Costa Rica / Boca Raton, Florida		Sep 2008 – Jun 2009
	Senior Software Developer <i>Software development and integration. ☆ Integrated commercially successful ChartFX visualization suite to Microsoft Sharepoint.</i>		
	StaffDotNet • San José, Costa Rica		Feb 2006 – Aug 2008
	Consultant / Developer <i>Consulting and development of software solutions based on .NET emerging technologies such as J# and Visual Studio Tools for Office. ☆ Developed a .NET Smart documents system for automation of proceedings generation for Banco Nacional de Costa Rica executive board using Microsoft Office directly as user interface.</i>		
	Grupo Babel Software & IT Services • San José, Costa Rica		Mar 2004 – Feb 2006
	Junior Software Developer <i>Developer of custom software solutions for government institutions, banking, and private sector. ☆ Developed a customizable point-of-sale system that integrates sales, inventory, kitchen orders, and a mobile interface for waitstaff, still in use today.</i>		
Publications	<i>Domain Adaptive Multiflow Networks.</i> R. Bermúdez-Chacón , M. Salzmann, P. Fua. (to appear)		
	<i>Visual Correspondences for Unsupervised Domain Adaptation on Electron Microscopy Images.</i> R. Bermúdez-Chacón , O. Altingövde, M. Salzmann, P. Fua. IEEE Transactions on Medical Imaging, 2019		
	<i>A domain-adaptive two-stream U-Net for Electron Microscopy image segmentation.</i> R. Bermúdez-Chacón , P. Márquez-Neila, M. Salzmann, P. Fua. International Symposium of Biomedical Imaging, 2018		
	<i>Scalable Unsupervised Domain Adaptation for Electron Microscopy.</i> R. Bermúdez-Chacón , C. Becker, M. Salzmann, P. Fua. International Conference on Medical Image Computing and Computer-Assisted Intervention, 2016		
Certifications	Microsoft Certified Professional		Jun 2006
	Microsoft Certified Application Developer: Microsoft .NET		Aug 2006
	Microsoft Certified Technology Specialist: SQL Server 2005		Feb 2007
	Microsoft Certified Solution Developer: Microsoft .NET		Feb 2007
	Microsoft Certified Technology Specialist: - Microsoft Office SharePoint Server 2007, Application Development		Nov 2009
Languages	Spanish	Mother tongue	
	English	Full Working Proficiency	
	Greek	Intermediate	
	French	Basic	