On domain-adaptive machine learning

Kouw, Wouter

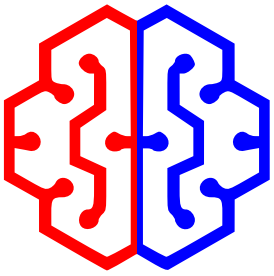
DOI
10.4233/uuid:630ce39a-76d8-49e5-bf5e-aec15fde79b3

Publication date
2018

Document Version
Final published version

Citation (APA)
Kouw, W. (2018). *On domain-adaptive machine learning*. [Dissertation (TU Delft), Delft University of Technology]. https://doi.org/10.4233/uuid:630ce39a-76d8-49e5-bf5e-aec15fde79b3

On domain-adaptive machine learning



On domain-adaptive machine learning

Proefschrift

ter verkrijging van de graad van doctor
aan de Technische Universiteit Delft,
op gezag van de Rector Magnificus prof. dr. ir. T.H.J.J. van der Hagen,
voorzitter van het College voor Promoties,
in het openbaar te verdedigen op dinsdag 5 juni 2018 om 15:00 uur

door

Wouter Marco KOUW
Master of Science in Cognitive and Clinical Neuroscience,
Universiteit Maastricht, Nederland

geboren te Breda, Nederland.

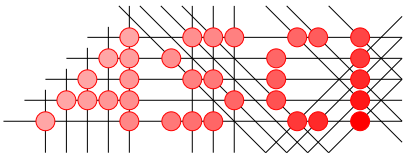
Dit proefschrift is goedgekeurd door de promotoren.

Samenstelling promotiecommissie:

Rector Magnificus,	voorzitter
Prof. dr. ir. M.J.T. Reinders,	Technische Universiteit Delft, promotor
Prof. dr. M. Loog,	Technische Universiteit Delft, Universiteit Kopenhagen, copromotor

Onafhankelijke leden:

Prof. dr. ir. G. Jongbloed	Technische Universiteit Delft
Prof. dr. T.M. Heskes	Radboud Universiteit Nijmegen
Prof. dr. M. van de Wiel	Vrije Universiteit Amsterdam
Prof. dr. C. Igel	Universiteit Kopenhagen, Denemarken
Dr. T.E.J. Mensink	Universiteit van Amsterdam
Prof. dr. E. Eisemann	Technische Universiteit Delft, reservelid



This work was carried out in the ASCI graduate school.
ASCI dissertation series number 390.

Printed by: Ipskamp Printing

ISBN 978-94-028-1048-6

An electronic version of this dissertation is available at
https://repository.tudelft.nl/.

Εἴ τίς με ἐλέγξαι καὶ παραστῆσαί μοι, ὅτι οὐκ
ὀρθῶς ὑπολαμβάνω ἢ πράσσω, δύναται,
χαίρων μεταθήσομαι: ζητῶ γὰρ τὴν ἀλήθειαν,
ὑφ᾽ ἧς οὐδεὶς πώποτε ἐβλάβη, βλάπτεται
δὲ ὁ ἐπιμένων ἐπὶ τῆς ἑαυτοῦ ἀπάτης καὶ ἀγνοίας.

If someone can prove me wrong and
show me my mistake in any thought or action,
I shall gladly change. I seek the truth,
which never harmed anyone: the harm is to
persist in one's own self-deception and ignorance.

- Marcus Aurelius (Meditations 6:21)

Contents

Summary

Artificial intelligence, and in particular *machine learning*, is concerned with teaching computer systems to perform tasks. Tasks such as autonomous driving, recognizing tumors in medical images, or detecting suspicious packages in airports. Such systems learn by observing examples, i.e. data, and forming a mathematical description of what types of variations occur, i.e. a statistical model. For new input, the system computes the most likely output and makes a decision accordingly. As a scientific field, it is situated between statistics and and algorithmics. As a technology, it has become a very powerful tool due to the massive amounts of data being collected and the drop in the cost of computation.

However, obtaining *enough* data is still very difficult. There are often substantial financial, operational or ethical considerations in collecting data. The majority of research in machine learning deals with constraints on the amount, the labeling and the types of data that are available. One such constraint is that it is only possible to collect labeled data from one population, or domain, but the goal is to make decisions for another domain. It is unclear under which conditions this will be possible, which inspires the research question of this thesis: when and how can a classification algorithm generalize from a source domain to a target domain?

My research has looked at different approaches to *domain adaptation*. Firstly, we have asked some critical questions on whether the standard approaches to model validation still hold in the context of different domains. As a result, we have proposed a means to reduce uncertainty in the validation risk estimator, but that does not solve the problem completely. Secondly, we modeled the transfer from source to target domain using parametric families of distributions, which works well in simple contexts such as feature dropout at test time. Thirdly, we looked at a more practical problem: tissue classifiers trained on data from one MRI scanner degrade when applied to data from another scanner due to acquisition-based variations. We tackled this problem by learning a representation for which detrimental variations are minimized while maintaining tissue contrast. Finally, considering that many approaches fail in practice because their assumptions are not met, we designed a parameter estimator that never performs worse than the naive non-adaptive classifier.

Overall, research into domain-adaptive machine learning is still in its infancy, with many interesting challenges ahead. I hope that this work contributes to a better understanding of the problem and will inspire more researchers to tackle it.

Samenvatting

Kunstmatige intelligentie, en in het bijzonder *machinaal leren*, draait om computersystemen die leren om taken uit te voeren. Taken zoals autonoom rijden, tumor herkenning in medische beelden, of detectie van verdachte pakketten op vliegvelden. Zulke systemen leren door het observeren van voorbeelden, i.e. data, en vormen een wiskundige beschrijving van de variaties die voorkomen, i.e. een statistisch model. Voor nieuwe input berekent het systeem de meest waarschijnlijke output en maakt op basis daarvan een beslissing. Als wetenschappelijk veld staat machinaal leren tussen statistiek en algoritmiek. Als technologie is het een krachtig stuk gereedschap vanwege de beschikbaarheid van grote hoeveelheden data en de lage kosten van berekeningen uitvoeren.

Maar *genoeg* data verzamelen is nog steeds erg moeilijk. Er zijn vaak lastige financiële, operationele of ethische overwegingen in data collectie. Onderzoek in machinaal leren draait daarom vooral om het omgaan met beperkingen op de hoeveelheid, de annotatie en de typen data die beschikbaar zijn. Eén zo'n beperking is dat het alleen mogelijk is om data te krijgen van één populatie, oftewel domein, terwijl het doel is om beslissingen te maken voor een andere populatie. Het is onduidelijk onder welke condities dit mogelijk is. Dit leidt tot mijn onderzoeksvraag: wanneer en hoe kan een classificatie algoritme generaliseren van een bron domein naar een doel domein?

Mijn onderzoek heeft gekeken naar verschillende manieren om *domein adaptatie* aan te pakken. Ten eerste, hebben we kritische vragen gesteld over model validatie in de context van verschillen in domeinen. Daaruit is een methode voortgekomen die de onzekerheid van een validatie schatter reduceert, maar daarmee lijkt nog niet alles gezegd te zijn. Ten tweede, hebben we de overgang van bron naar doel domein gemodelleerd met uitvaldistributies, wat goed werkt wanneer informatie in het doel domein wegvalt. Ten derde, hebben we gekeken naar een iets praktischer probleem: weefsel classificeerders getraind op data van één MRI scanner presteren slecht op data van een andere scanner. Om dit op te lossen hebben we een representatie geleerd waarin scanner-gerelateerde variatie minimaal word terwijl weefsel contrast bewaard blijft. Als laatste, omdat veel methoden in de praktijk niet werken vanwege invalide assumpties, hebben we een parameter schatter ontworpen die nooit slechter presteert dan de naïeve non-adaptieve aanpak.

Tot slot, onderzoek naar domein-adaptief machinaal leren staat nog in de kinderschoenen, met vele interessante open vragen. Ik hoop dat dit werk andere onderzoekers aanspoort om deze uitdaging ook aan te gaan.

1

Introduction

In this chapter, I first introduce the concept of computer systems that learn to perform a task. Branching out from the standard framework of supervised learning, I pose my research questions on generalizing across domains. Following those, I discuss a number of theoretical analyses that have proven to be very insightful and present a categorization of approaches including important algorithms. Finally, I briefly discuss the contributions of this thesis to domain-adaptive machine learning.

1

Intelligent systems learn from data to recognize patterns, predict outcomes and make decisions [1, 2]. In data-abundant problem settings, such as recognizing objects in images, these systems achieve super-human levels of performance [3]. Their strength lies in their ability to process huge amounts of examples and obtain a detailed estimate of what *does* and *does not* constitute the object they are looking for. In recent years, the explosion in data collection and open access has thrusted machine learning into the limelight. It is now a key technology in self-driving cars [4], drone guidance [5], computer-assisted diagnosis [6], online commerce [7], satellite cartography [8], exo-planet discovery [9], and machine translation [10], with many more applications on the horizon.

Machine intelligence refers to a computer's ability to *learn* to perform a task [11]. Supervised systems learn through *training*, where the system is rewarded or punished based on whether it produces the right output for a given input [12, 13]. In order to train an intelligent system, one requires a set of matching inputs and outputs. Most often, inputs consist of complicated objects such as images while outputs consists of decisions such as 'yes' or 'no' or classes such as 'apple', 'pear', 'berry', etc. The system will try out many classification functions on the set of inputs and select the function that produced the least errors. If the examples in the dataset are similar to new inputs, then the system will make accurate decisions in the future as well. Classifying new inputs based on a finite set of examples, is called *generalization*. For example, suppose patients are measured on various biometrics such as blood pressure, and have been classified as 'healthy' or 'ill'. Then, a system can be trained by finding the decision function that best diagnoses the patients. If they are an accurate reflection of the population of all possible patients, then the trained system will generalize to new patients as well.

However, if the collected data it is *not* an accurate reflection of the population, then the system will *not* generalize well. Data is *biased* if certain events are observed more frequently than usual while others are observed less frequently. If data is biased, then the system will think that certain outcomes are also more likely to occur. For example, data collected from older patients is biased with respect to the total human population. Researchers in statistics and social sciences have long studied problems with sample biases and have developed a number of techniques to correct for biased data [14–16]. However, there are still fundamental limitations on generalizing towards wider populations. Instead, machine learning researchers are attempting to generalize towards specific *target* populations. For instance, can we use information from *adult* humans to train an intelligent system for diagnosing *infant* heart disease?

In order to target specific populations, we need at least *some* idea of what it looks like. Labeled data, i.e. input-output pairs, is often not available from the target population. But usually there is some unlabeled data as well as some labeled data from another source. Under certain conditions, relationships between populations can be found. Given such a relationship, an intelligent system can now *adapt*, i.e. change its decisions from the source population to generalize more towards the specific target population [17].

A more detailed example of adaptation is the following: in clinical imaging settings, radiologists manually annotate tissues, abnormalities, and pathologies of sets of patients. Biomedical engineers then use these annotations to train systems to perform automatic tissue segmentation or pathology detection in medical images. Now suppose a hospital installs a new MRI scanner. Unfortunately, due to the mechanical configuration, calibration, vendor and acquisition protocol of the scanner, the images it produces will differ from images produced by other scanners [18–20]. Consequently, systems trained on data from other scanners would fail to perform well on the new scanner. However, an adaptive system would find correspondences in images between scanners, and change its decisions accordingly. Thus it avoids the time, money and energy needed to annotate data for the target population (in this case, images from the new scanner) [18, 19]. Chapter 5 of this thesis describes a method that allows for targeted generalization towards a particular MRI scanner.

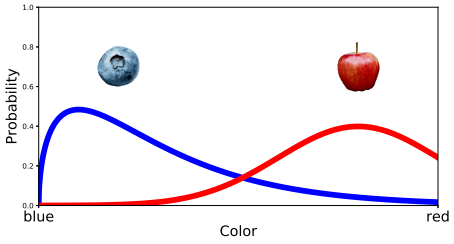
Adaptation is making an impact in a number of other fields as well: in bioinformatics, adaptive approaches have been successful in sequence classification [21, 22], gene expression analysis [23, 24], and biological network reconstruction [25, 26]. There, the types of populations that are predominantly considered are different model organisms or different data-collecting research institutes [27]. In predictive maintenance, every time the fault prognosis system raises an alarm and designates that a component has to replaced, the machine changes its properties [28]. That means that the system will have to adapt to the new setting, until another component is replaced and it will have to adapt again. In search-and-rescue robotics, a system that needs to autonomously navigate wilderness trails will have to adapt to detect concrete structures if it is to be deployed in an urban environment [5, 8]. Computer vision systems that recognize activities have to adapt across different surroundings as well as different groups of people [29–31]. In natural language processing, texts from different publication platforms are tricky to analyze due to different contexts and differences between how authors express themselves. For instance, financial news articles use a vocabulary that differs from the one in biomedical research abstracts [32]. Similarly, online movie reviews are linguistically different from tweets [33]. Sentiment classification relies heavily on context as well; people use different words to express whether they like a book versus whether they like an electronic gadget [34]. Adapting for the target population is very important to online retailers that use sentiment classifiers in their recommender systems. When a new product category is introduced, there is no data available to link users and items. In that case, there is an interest in using online reviews from other product categories to aid in classifying sentiments in the new category [34, 35].

In some situations, the target population is a subpopulation. *Personalization* is an extreme case of this. One of the first types of systems to target subpopulations are spam filters: they are often initialized as general systems but adapt to specific users [36]. Male users receive different kinds of spam than female users for instance, which the system can detect and adapt to based purely on text statistics. Alternatively, in speaker recognition, an initial speaker-independent system can adapt to new speakers [37]. Similarly, general face recognition systems can be adapted to specific persons [38] and person-independent activity recognition algorithms can be specialized to particular individuals [39].

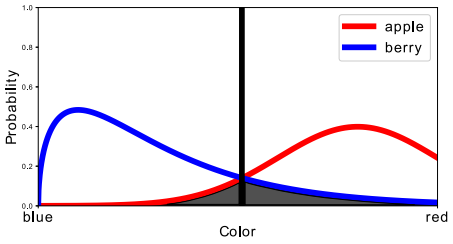
However, the analysis of adaptation is not complete, and it is not clear which conditions have to be fulfilled in order for the system to perform well. It seems that in cases where it is difficult to describe how two populations relate to each other, adaptive systems suffer from high variability: they are highly uncertain about their decisions. In this thesis, several approaches to the adaptation problem are explored. But in order to study it in greater detail, it is necessary to delve into several core concepts from machine learning. The next section gives a short explanation of how intelligent decision-making systems work. Following that, various types and causes of biases are described. The last section of this chapter presents an overview of approaches to adaptation.

1.1. Risk minimization

One of the most thoroughly researched frameworks for the design, construction and analysis of intelligent systems is *risk minimization*. It is part of statistical decision theory and is based on the notion that objects vary [40, 41]. In order to represent an object digitally, we measure one or more *features*. For example, an apple can be described in terms of its overall color. A feature captures information about the object; many apples are red, some are green, but none are blue. These variations over color x can be described by a probability distribution $p(x)$. In order to decide between an apple and say, a berry, the system needs to know which of the two is more probable for a given color, i.e. $p(\text{apple} \mid x) > p(\text{berry} \mid x)$ or $p(\text{apple} \mid x) < p(\text{berry} \mid x)$ [42]. Figure 1.1a describes two probability distributions as a function of color; the red distribution corresponds to apples and the blue to berries.



(a) Probability distributions of apples and berries, as a function of color



(b) Error of the optimal classifier, consisting of the gray area under the distributions

Figure 1.1: Example of a classification problem.

A decision-making problem can be abstractly described as a setting where a system has to assign a class, from a finite set of possible classes, to every possible variation of an object. Decision-making systems are therefore called statistical *classifiers*. In their most basic form they consist purely of a function that takes as input an object, encoded by features, and outputs one of the possible classes, e.g. $h(x) = \text{berry}$. Its output is also called its prediction, as there are problem settings where classification errors are unavoidable. We will refer to the classifier itself as h, while its prediction is denoted by its application to a particular object $h(x)$. Returning to the apple-berry problem, a classifier can be seen as a threshold, illustrated in Figure 1.1b by a black vertical line. It designates everything to the

left as a berry and everything to the right as an apple. Hence, all apples left of the line and all berries to the right are misclassified. The classification error is visualized as the gray region under the distributions and can be written mathematically as:

$$e(h) = \int_{\mathcal{X}} [h(x) \neq \text{apple}]\, p(x \mid \text{apple}) p(\text{apple})\, \mathrm{d}x + \int_{\mathcal{X}} [h(x) \neq \text{berry}]\, p(x \mid \text{berry}) p(\text{berry})\, \mathrm{d}x\,, \tag{1.1}$$

where $h(x)$ refers to the decision made by the classifier. $p(\text{apple})$ and $p(\text{berry})$ refer to the probability of encountering apples and berries in general, while $p(x|\text{apple})$ and $p(x|\text{berry})$ refer to the probabilities of seeing an apple or berry of a given color x (also known as the *class-conditional* distributions). The classifier should be able to make a decision over all possible colors $\mathcal{X}$. Since color is a continuous variable, the decision function is integrated over all possible colors. If the objects were measured on a discrete variable, then the integration would be equivalent to a sum. Essentially, the first term describes how often the classifier will make a mistake in the form of deciding that an actual apple is not an apple and the second term describes how often it thinks that a berry is not a berry. Summing these two terms constitutes the overall classification error $e(h)$.

If apples and berries are encoded into a more general form, as a variable y, then the classification error can be written as follows:

$$e(h) = \sum_{y \in \mathcal{Y}} \int_{\mathcal{X}} [h(x) \neq y]\, p(x, y)\, \mathrm{d}x\,. \tag{1.2}$$

where $p(x, y) = p(x \mid y)p(y)$. $\mathcal{Y}$ numerically represents the set of classes, in this case $\mathcal{Y} = \{\text{apple} = -1, \text{berry} = +1\}$. Objects are often not described by one feature but by multiple measured properties. As such, x is a D-dimensional random vector, and can be continuous, i.e. consisting of real values $\mathcal{X} \subseteq \mathbb{R}^D$, can be discrete, i.e. consisting of integers $\mathcal{X} \subseteq \mathbb{N}^D$, or a mix of both.

1.1.1. Loss functions

The notion of disagreement between the predicted and the true class can be described in a more general form by using a function that describes the numerical cost of correct versus incorrect classification. This function is known as a *loss* function ℓ, which takes as input the classifier h along with the object x and the object's true class y: $\ell(h(x), y) \geq 0$. The pure classification error is known as the $0/1$ *loss*, denoted $\ell_{0/1}$, that has value 0 whenever the prediction is exactly equal to the true label and value 1 whenever they are not equal; $\ell_{0/1}(h(x), y) = [h(x) \neq y]$. Other examples of loss functions are the *quadratic* or *squared* loss, $\ell_{\text{qd}}(h(x), y) = (h(x) - y)^2$, the *logistic loss* $\ell_{\log}(h(x), y) = yh(x) - \log \sum_{y' \in Y} \exp(y' h(x))$ or the *hinge loss* $\ell_{\text{hinge}}(h(x), y) = \max(0, 1 - yh(x))$. These are called *convex surrogate* losses, as they approximate the $0/1$ loss but use a formulation that is easier to work with computationally. Overall, the choice of a loss function has a major impact on the behaviour of the resulting classifier.

Considering that we are integrating the loss function with respect to probabilities, we are actually looking at the expected loss, also called the *risk*, of a particular classifier:

$$R(h) = \mathbb{E}_{X,Y}\, \ell(h(x), y)\,, \tag{1.3}$$

where $\mathbb{E}$ stands for the expectation. Its subscript denotes which variables are being integrated over. Given a risk function, we can evaluate multiple possible classifiers and select the one for which the risk is as small as possible:

$$h^* = \arg\min_h \mathbb{E}_{X,Y}\, \ell(h(x), y)\,. \tag{1.4}$$

The asterisk superscript denotes optimality with respect to the chosen loss function. There are many ways to perform this minimization step, with vastly different computational costs. The main advantage of convex loss functions is that they do not contain local minima and efficient optimization procedures such as gradient descent can be used [43].

1.1.2. Generalization

Up to this point, we have only considered the case where the probability distributions are completely known. In practice, this is rarely the case: only a finite amount of data can be collected. Measurements of objects can be described as a dataset $\mathcal{D}^n = \{(x_i, y_i)\}_{i=1}^n$, where each x_i is an independent sample from the random variable $\mathcal{X}$, and is labeled with its corresponding class y_i. The expected value with respect to the joint distribution of data and labels can be approximated with the sample average:

$$\hat{R}(h \mid \mathcal{D}^n) = \frac{1}{n} \sum_{i=1}^{n} \ell(h(x_i), y_i)\,. \tag{1.5}$$

$\hat{R}$ is called the *empirical risk* function. It evaluates classifiers *given* a particular dataset (the symbol " | " denotes that a function is dependent on something). Note that the true risk R from (1.3) does not depend on a dataset. Minimizing the empirical risk with respect to a classifier for a particular dataset, is called *training* the classifier:

$$\hat{h} = \arg\min_{h \in \mathcal{H}} \hat{R}(h \mid \mathcal{D}^n) \tag{1.6}$$

where $\mathcal{H}$ refers to the collection of all possible classifiers that we consider, also known as the hypothesis space. A risk-minimization system is said to *generalize* if it uses information on specific objects to make decisions for all possible objects.

Generally, more samples lead to better approximations of the risk, and the resulting classifier will be closer to the optimal one. For n samples that are independently drawn and identically distributed, due to the law of large numbers, the empirical risk converges to the true risk [13]:

$$\lim_{n\to\infty} \frac{1}{n} \sum_{i=1}^{n} \ell(h(x_i), y_i) = \mathbb{E}_{X,Y}\, \ell(h(x), y)\,, \tag{1.7}$$

and it can be shown that the resulting classifier will converge to the optimal classifier $\lim_{n\to\infty} \hat{h} \to h^*$ [13, 44]. The minimizer of the empirical risk deviates from the true risk due to the estimation error, i.e. the difference between the sample average and the actual expected value, as well as the optimization error, i.e. the difference between the true minimizer and the one obtained through the optimization procedure [13, 45].

Ultimately, we are not interested in the error of the trained classifier on the given dataset, but in the error on all possible future samples: $e(\hat{h}) = \mathbb{E}_{X,Y}[\hat{h}(x) \neq y]$. This error is known as the *generalization error* [13, 46]. As mistakes are sometimes inevitable, we mostly focus on how much larger the generalization error of the trained classifier is compared to the generalization error of the optimal classifier $e(\hat{h}) - e(h^*)$. Ideally, we would like to know if the generalization error will be small, i.e., less than some small value ϵ. In other words, that our classifier will be *approximately correct*. However, because classifiers are functions of datasets, and datasets are random, we can only describe how probable it is that any classifier is approximately correct. Hence, the Probably Approximately Correct (PAC) bound:

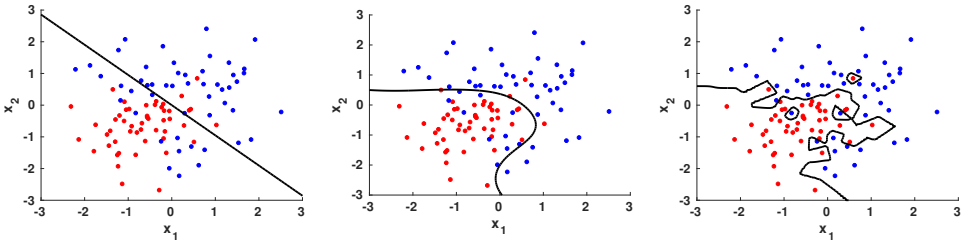
$$\Pr_{\mathcal{D}^n}\left[\, e(\hat{h}) - e(h^*) \;\leq \epsilon \,\right] \geq 1 - \delta\,, \tag{1.8}$$

where δ is a small number [47, 48]. Every dataset leads to a different $\hat{h}$ and we can therefore integrate over trained classifiers by integrating over the probability of drawing any particular dataset (hence the subscript $\mathcal{D}^n$). Essentially, the PAC bound states that, with probability at least $1 - \delta$, the classifier is close to optimal. Specific values for δ and ϵ can be found through plugging in a probability distribution and a function class.

PAC bounds do not study single datasets or choices of algorithms, but describe how the generalization error depends on sample size, the joint distribution and classifier complexity. They avoid the randomness inherent to evaluating specific classifiers on particular datasets, which makes them useful tools for comparisons and analysis. Many variants of PAC bounds have been proposed, some using different measures of complexity, such as Rademacher complexity [49] or Vapnik-Chervonenkis dimensions [50, 51], while others use Bayesian inference [52–54]. Generalization error bounds, as well as learning bounds - inequalities describing how many samples a particular algorithm requires to achieve a specific generalization error - can incorporate assumptions or prior knowledge [13, 55–57]. Bounds with assumptions do not hold universally, but are restricted to the settings specified by the assumption. Due to these restrictions, these bounds are often tighter (there is more certainty whether the classifier will be approximately correct). Such tighter generalization bounds often inspire new algorithms, such as Adaboost or the Support Vector Machine [58, 59].

Learning bounds also tell us that the flexibility, or complexity, of a classifier has to be traded off with the number of available training samples [44, 57, 60]. In particular, a very flexible model can minimize the error on a given dataset completely, but will be too specific to generalize to new samples. This is known as *overfitting*. Figure 1.2c illustrates an example of a classifier that has perfectly fitted to the training set. As can be imagined,

it will not perform as well for new samples. In order to combat overfitting, an additional term is introduced in the empirical risk estimator that punishes model flexibility. This *regularization* term is often a simple additive term in the form of the norm of the classifier's parameters [12, 61]. Figure 1.2b visualizes an example of a properly regularized classifier, that will probably generalize well to new samples. Figure 1.2a shows an example of a too heavily regularized classifier, also known as an "underfitted" classifier.



(a) Underfitted classifier. (b) Well-fitted classifier. (c) Overfitted classifier.

Figure 1.2: Examples of classifier complexities.

1.2. Research question

Normally, samples from one distribution are used to generalize towards new samples from the *same* distribution. However, in practice, new samples are often drawn from a different distribution: the training data might be drawn from a local population, such as a social science experiment where only university students are measured, whereas the test data might be drawn from the national population (an example of a biased sample). Or, it could be that the object of interest (unknowingly) changes over time (an example of a non-stationary data-generating process). Hence, there is a strong interest in developing machine learning methods that can generalize from data from one distribution to data from another.

Such problem settings are known as *domain adaptation* or *transfer learning* settings [17, 62, 63]. The distribution of interest is called the *target domain*, for which labels are usually not available and training a classifier is not possible. However, if a similar domain is available, it could be used as a source of additional information. Now the challenge is to overcome the differences between the domains so that a classifier trained on the source domain generalizes well to the target domain. Such a method is called a *domain-adaptive classifier*. If successful, domain-adaptive classifiers can, for example, make accurate diagnoses for rare forms of cancer based on knowledge from common forms of cancer [64], detect real-world driving lanes from data of high-quality driving simulations [65], or parse part-of-speech tags in literature based on data from news articles [66].

Generalizing across distributions is very difficult and it is not clear under which conditions it is possible. My work therefore focuses on the question:

When and how can a statistical classifier generalize from a source to a target domain?

In the other chapters, I present two analyses (Chapters 2 and 3) and three methods (Chapters 4,5 and 6). Each chapter studies the problem from a different perspective. The discussion chapter reflects on my findings, lists some of the questions that have opened up and presents ideas for future work. For the remainder of this introduction chapter, I will explain domains in greater detail, discuss types of domain shifts and present a categorization of approaches to domain adaptation.

1

1.3. Domain adaptation

Unfortunately, there exists quite a bit of confusion in the literature concerning definitions that are important to the process of generalizing to a different distribution. A clarification is therefore in order. To be precise, we define domains as the combination of an input space $\mathcal{X}$, an output space $\mathcal{Y}$ and an associated probability distribution p. Inputs are subsets of the D-dimensional real space $\mathbb{R}^D$, while outputs are classes. Classes can be binary, in which case $\mathcal{Y}$ corresponds to $\{-1, +1\}$ or can be K-order multi-class, in which case $\mathcal{Y} = \{1, \dots K\}$. Given two domains, we call them different if they are different in at least one of their constituent components, i.e., the input space, the output space, or the probability density function. For example, image caption generators from computer vision generalize from the "image domain" to the "text domain", which would be an example of differences between the input spaces [67, 68]. This thesis is restricted to the case where only the probability distributions differ. We denote the source domain as $(\mathcal{X}, \mathcal{Y}, p_{\mathcal{S}})$ and will sometimes refer to it in shorthand as $\mathcal{S}$. The target domain is denoted $(\mathcal{X}, \mathcal{Y}, p_{\mathcal{T}})$ with the shorthand $\mathcal{T}$. Domain-specific functions will be marked with the subscript $\mathcal{S}$ or $\mathcal{T}$ as well. For example, the expected value with respect to the target domain will be written as: $\sum_{y \in \mathcal{Y}} \int_{\mathcal{X}} f(x, y)\ p_{\mathcal{T}}(x, y)\ \mathrm{d}x = \mathbb{E}_{\mathcal{T}}[f(x, y)]$. With some abuse of notation for the sake of clarity, we will mark marginal and conditional distributions with $\mathcal{S}$ and $\mathcal{T}$ as well; $p_{\mathcal{T}}(x, y)$ for the target joint distribution, $p_{\mathcal{T}}(x)$ for the target data marginal distribution and $p_{\mathcal{T}}(x \mid y)$ for the target class-conditional distribution.

Samples from the source domain are denoted with (x_i, y_i), and the source dataset is referred to as $\mathcal{D}_{\mathcal{S}}^n = \{(x_i, y_i)\}_{i=1}^n$. Note that x refers to an element of the input space $\mathcal{X}$ while x_i refers to a specific observation drawn from the source distribution, $x_i \sim p_{\mathcal{S}}$. Likewise, samples from the target domain are denoted with (z_j, u_j), with its dataset $\mathcal{D}_{\mathcal{T}}^m = \{(z_j, u_j)\}_{j=1}^m$.

Generalizing across domains

The PAC bound from (1.8) describes how much a classifier trained on samples from a distribution will generalize to new samples from that distribution. However, this result is based on Hoeffding's inequality, which only describes deviations of the empirical risk estimator from its *own* true risk, not deviations from *other* risks [13, 69, 70]. Since Hoeffding's inequality does not hold in a cross-domain setting, the standard generalization error bound does not hold either.

However, it is possible to derive generalization error bounds if more is known on the relationship between $\mathcal{S}$ and $\mathcal{T}$ [17, 54, 71–81]. For example, one of the first target generalization error bounds uses the condition that there exists a classification function that can perform well on both domains [17, 72]. This low-joint-domain-error condition is expressed as $\min_{h \in \mathcal{H}} [e_{\mathcal{S}}(h) + e_{\mathcal{T}}(h)] \leq \lambda$. As will be shown later, the deviation between the target generalization error of a classifier trained in the source domain $e_{\mathcal{T}}(\hat{h}_{\mathcal{S}})$ and the target generalization error of the optimal target classifier $e_{\mathcal{T}}(h_{\mathcal{T}}^*)$ depends on this value λ. If λ is too large, then the source trained classifier can never be approximately correct in the target domain.

Additionally, we need some measure of how much two domains differ from each other. For this bound, the *symmetric difference hypothesis* divergence ($\mathcal{H}\Delta\mathcal{H}$-divergence) is used, which takes two classifiers and looks at to what extent they disagree with each other on both domains [17]:

$$d_{\mathcal{H}\Delta\mathcal{H}}(p_{\mathcal{S}}, p_{\mathcal{T}}) = 2 \sup_{h,h' \in \mathcal{H}} \left| \Pr_{\mathcal{S}}[h \neq h'] - \Pr_{\mathcal{T}}[h \neq h'] \right| , \tag{1.9}$$

where the probability Pr can be computed through integration: $\Pr_{\mathcal{S}}[h \neq h'] = \int_{\mathcal{X}} [h(x) \neq h'(x)] p_{\mathcal{S}}(x) \mathrm{d}x$. The sup stands for the *supremum*, which in this context finds the pair of classifiers h, h' for which the difference in probability is largest and returns the value of that difference [17, 72, 82].

Given the condition of low-joint-domain-error and the $\mathcal{H}\Delta\mathcal{H}$-divergence, one can formulate a domain adaptive PAC bound as:

$$\Pr_{\mathcal{D}_{\mathcal{S}}^n} \left[e_{\mathcal{T}}(\hat{h}_{\mathcal{S}}) - e_{\mathcal{T}}(h_{\mathcal{T}}^*) \leq \lambda + \frac{1}{2} d_{\mathcal{H}\Delta\mathcal{H}}(p_{\mathcal{S}}, p_{\mathcal{T}}) + \mathcal{C}(\mathcal{H}) \right] \geq 1 - \delta , \tag{1.10}$$

where $e_{\mathcal{T}}$ is the true error on the target domain, $\hat{h}_{\mathcal{S}}$ is the classifier trained on a sample from the source domain, $h_{\mathcal{T}}^*$ is the optimal classifier in the target domain, and λ describes the maximum joint-domain-error (Theorem 3, [17]). $\mathcal{C}(\mathcal{H})$ describes the complexity of the type of classification functions $\mathcal{H}$ we are using, and comes up in standard generalization error bounds that incorporate classifier complexity [83]. Overall, this bound states that, with probability at least $1 - \delta$, the generalization error of a classifier, with complexity $\mathcal{C}(\mathcal{H})$, trained on source data, will be less than the maximum joint-domain-error and the domain discrepancy. Or simpler said: the larger λ and $d_{\mathcal{H}\Delta\mathcal{H}}$ are for a given domain adaptation problem, the less a source classifier will generalize to the target domain.

In conclusion, in order to generalize from one domain to another, we need some knowledge on how the two domains relate to each other. Sometimes, these relationships are simple in the sense that only some variables have *shifted* while the remainder stay the same across domains. The section on shifts below, section 1.3.1, elaborates on how this information can be exploited. For more general domain discrepancies, there are more complicated conditions that have to be fulfilled. These are shortly discussed in section 1.3.1. Section 1.4 describes proposed methods that make use of one or more of these conditions.

1.3.1. Shifts

We are ultimately interested in minimizing the target risk $R_{\mathcal{T}}$, but we want to do this by making use of the source domain. One of the most straightforward ways to incorporate the source distribution in the target risk is as follows:

$$\begin{aligned} R_{\mathcal{T}}(h) &= \sum_{y \in Y} \int_{\mathcal{X}} \ell(h(x) \mid y)\, p_{\mathcal{T}}(x, y)\, \mathrm{d}x \\ &= \sum_{y \in Y} \int_{\mathcal{X}} \ell(h(x) \mid y)\, \frac{p_{\mathcal{T}}(x, y)}{p_{\mathcal{S}}(x, y)}\, p_{\mathcal{S}}(x, y)\, \mathrm{d}x . \end{aligned} \tag{1.11}$$

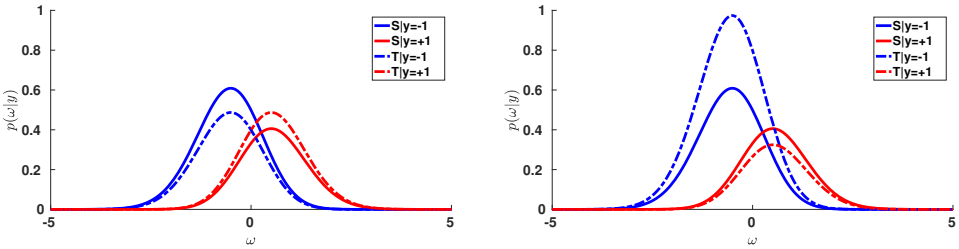
One could now approximate this risk function by plugging in source samples and weighting their loss by the ratio of distributions; $n^{-1} \sum_i^n \ell(h(x_i), y_i) p_{\mathcal{T}}(x_i, y_i) / p_{\mathcal{S}}(x_i, y_i)$ (note that $p_{\mathcal{T}}(x_i, y_i)$ evaluates the probability of a source sample under the target distribution). However, in order to compute the ratio $p_{\mathcal{T}}/p_{\mathcal{S}}$, we would need labeled data from both domains, which is often not available. Fortunately, if the domains are shifted versions of each other, then we do not always need labeled target data. The following subsections discuss three types of shifts: between prior distributions, between data / covariate distributions, and between class-posteriors / concepts. Other types of shifts can occur, for instance mixture component shifts [84], but those are outside the scope of this work.

Prior shift

First of all, there is the case where only the prior probabilities of the classes are different: $p_{\mathcal{S}}(y) \neq p_{\mathcal{T}}(y)$. This can occur in for example fault detection settings, where a new maintenance policy might cause less faults [85], or in the detection of oil spills before versus after an incident [86]. Since only the priors are different, the class-conditional distributions are still the same: $p_{\mathcal{S}\,|\,y}(x \mid y) = p_{\mathcal{T}\,|\,y}(x \mid y)$. We can exploit this information by reducing the ratio of joint probability distributions [87]:

$$
\begin{aligned}
R_{\mathcal{T}}(h) &= \sum_{y \in Y} \int_{\mathcal{X}} \ell(h(x), y) \frac{\cancel{p_{\mathcal{T}}(x \mid y)}\, p_{\mathcal{T}}(y)}{\cancel{p_{\mathcal{S}}(x \mid y)}\, p_{\mathcal{S}}(y)} p_{\mathcal{S}}(x, y)\, \mathrm{d}x \\
&= \mathbb{E}_{\mathcal{S}} \left[\ell(h(x), y)\, w(y) \right] , \qquad (1.12)
\end{aligned}
$$

where the weights $w(y) = p_{\mathcal{T}}(y)/p_{\mathcal{S}}(y)$ represent the change in the balance between classes. Using this approach, we require no unlabeled target samples, only a number of target labels. Figure 1.3a illustrates an example of two class-conditional distributions with imbalanced classes in the source domain (solid lines) and balanced classes in the target domain (dotted lines). Figure 1.3b shows the opposite case; going from an imbalanced class to an even more imbalanced class.



(a) Imbalanced to balanced classes.

(b) Imbalanced to more imbalanced classes.

Figure 1.3: Examples of types of class-prior shift.

Re-weighting each sample from a particular class is very similar to cost-sensitive learning, where we are not correcting for inappropriate priors but are artificially assigning new priors [88]. But prior shifts have also been extensively studied from a different perspective: when

it is more difficult to collect data from one class than the other [89]. For example, in a few countries, the government gives women above a certain age the opportunity to be tested for breast cancer [90]. The vast majority that responds does not show signs of cancerous tissue and only a small minority is tested positive. There is therefore a class imbalance in the data. Furthermore, because the test voluntary, only certain groups of women respond. The sample is therefore biased and there is no guarantee that the class proportions of the sample also hold for the whole population. However, if the general prevalence of a disease is known, then the prior shift can be corrected for [91, 92].

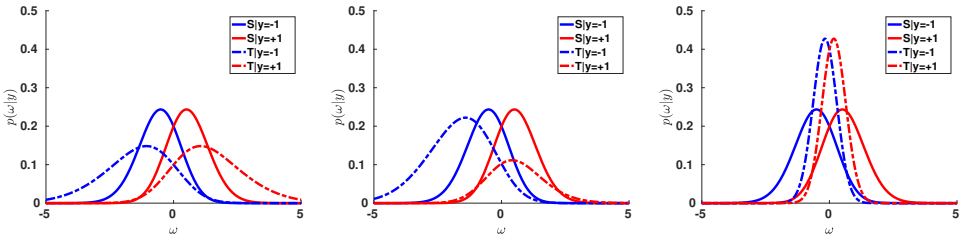
Covariate shift

Covariate shift is one of the most studied means of data shifts. For these cases, we know that $p_{\mathcal{T}}(y \mid x) = p_{\mathcal{S}}(y \mid x)$. This information can be exploited by rewriting the ratio of joint distributions in (1.11) into a ratio of class-posterior times marginal data distributions and canceling out the class-posteriors:

$$R(h) = \sum_{y \in Y} \int_{\mathcal{X}} \ell(h(x), y) \frac{\cancel{p_{\mathcal{T}}(y \mid x)}\, p_{\mathcal{T}}(x)}{\cancel{p_{\mathcal{S}}(y \mid x)}\, p_{\mathcal{S}}(x)} p_{\mathcal{S}}(x, y)\, \mathrm{d}x \tag{1.13}$$

$$= \mathbb{E}_{\mathcal{S}} \left[\ell(h(x), y)\, w(x) \right], \tag{1.14}$$

where the weights $w(x) = p_{\mathcal{T}}(x)/p_{\mathcal{S}}(x)$ indicate how the probability of a source sample should be corrected to reflect the probability under the target distribution.



(a) Wider target domain. (b) Limited support overlap. (c) Narrower target domain.

Figure 1.4: Examples of types of covariate shift.

There are many causes for covariate shifts, with sample selection bias being the most known one [14, 16, 93]. Especially in the social sciences where survey sampling is done locally, i.e. in universities, companies or city centers, the observed data reflects the local population and not the global one [15, 94]. This is often modeled with an additional variable s that denotes how probable it is that x will be selected. For example, suppose we go to a city that is populated according to a normal distribution, i.e., most people live in the center and the habitation density decreases as a function of the distance from the center. Local sampling, in the form of asking people on the main square to fill in a survey, corresponds to setting $p(s = 1 \mid x)$ very high in in the interval close to 0. Applying Bayes' theory, i.e., $p(x \mid s = 1) = p(s = 1 \mid x)p(x)/p(s = 1)$, shows that the collected surveys $p(x \mid s = 1)$ only represent people from the main square instead of the whole city's inhabitants $p(x)$.

From a domain adaptation perspective, the biased sampling defines the source domain $p_S(x) = p(x \mid s = 1)$. As the goal is to correct for the selection bias, the target domain consists of the selection variable being integrated out: $p_T(x) = \sum_{s \in \{0,1\}} p(x \mid s)$.

Similar to sample selection bias, another cause for covariate shift is missing data [95, 96]. In practice, data can be missing as measurement devices fail or because a subject dropped out of the experiment. When there is a consistent mechanism behind how the data went missing, referred to as missing-not-at-random (MNAR), the missingness constitutes an additional variable. This variable acts in the same way as the selection variable, as it decides whether or not a sample will be included in the training set.

The last common cause for covariate shift, is the use of different measurement instruments. For example, using different cameras to take photos of objects [62]. The object itself and how often it occurs, remain constant, which means that the priors and class-posteriors are equivalent in both domains. However, different camera settings lead to different photos, which means the marginal data distributions differ. Considering that these settings are mechanical and have a physical origin, one could argue that there exists a transformation from photos from one camera to photos from another [97, 98].

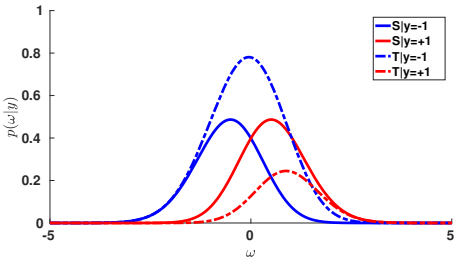
Concept shift

In the case of concept shifts, the definition of the class changes. For instance, [99] consider a medical setting where the aim is to make a prognosis for a patient based on their age, severity of their flu, general health and their socio-economic status. The classes are originally defined as "remission" and "complications", but at test time, other aspects are counted as a form of "complication" and are thusly labeled. Therefore, the classifier trained on the original labeling deteriorates in performance. Alternatively, in computer security, what constitutes an "anomaly" can not only be different for different users but can also change over time [100].

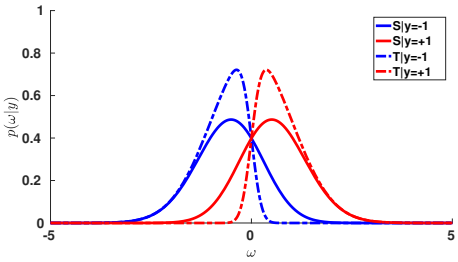
If only the concept has changed, then that means that the marginal data distributions remain the same: $p_S(x) = p_T(x)$. This knowledge can again be exploited through:

$$R_T(h) = \sum_{y \in Y} \int_X \ell(h(x), y) \frac{p_T(y \mid x) \cancel{p_T(x)}}{p_S(y \mid x) \cancel{p_S(x)}} p_S(x, y) \, \mathrm{d}x$$

However, unless there is some prior knowledge on the concept shift, adaptation in this setting is impossible without labeled target data. Unlike the prior and covariate shift cases, where only the data marginal or the class marginal distributions change, in this case a *conditional* distribution changes. To estimate conditional distributions, one requires simultaneous observations of both variables. Figure 1.5a shows an example of a shift in the location of the decision boundary, towards the right, but not in the conditional variance. Figure 1.5b shows the opposite example of a shift in the conditional variance, but not in the position.



(a) Change in position.



(b) Change in variance.

Figure 1.5: Examples of types of concept shift.

Unfortunately, this remarkably difficult setting occurs quite frequently as classifiers are deployed in non-stationary environments [101]. For smoothly varying non-stationarities such as time-series, however, there is again additional information that can be exploited: the shifts are ordered and are relatively small between neighboring time steps. Such a time-dependent setting is often referred to separately as concept *drift*. In many dynamical learning approaches, such as online learning or bandit settings, the classifier receives feedback after every decision it makes [102]. This feedback allows it to detect whether a concept drift has occurred and allows it to estimate how it should adapt accordingly [103–105].

Domain discrepancies

In the most general case, more than one of the above shifts will have occurred. There are many possible ways in which two datasets of the same objects may differ from one another. For example, if one were to search online for images, then one encounters posed objects on white backgrounds on commercial websites, natural photos with highly cluttered backgrounds on travel sites, indoor shots with widely varying lighting conditions on social media, and many more [106]. As can be imagined, this is the most difficult setting and learning will often not be possible at all [78, 107]. In order to generalize well, the domains have to be related in some other exploitable way. Examples of exploitable relationships include: the existence of a single good predictor for both domains [17, 72, 78, 108], constrained worst-case labellings [109, 110], low-data-divergence [17, 72, 78], the existence of a domain manifold [62, 111, 112], conditional independence of class and target given source data [113] and unconfoundedness [114]. This thesis does not explore the case of multiple sources [115, 116], or the related problem settings of multi-task learning [71], online learning [115, 117] or active learning [118, 119].

1.4. Approaches

This section discusses a number of approaches to domain adaptation based on supposed relationships between domains. In order to illustrate the ideas of some of the approaches, we use an example setting. Figure 1.6 visualizes a 2-dimensional scatter plot of red versus blue dots in the source domain (left) and the target domain (right). Training a classifier on the source samples will result in the black line (left), which will probably generalize

well to future source samples. However, applying it directly to the target samples without adaptation, will lead to a number of misclassifications. As can be imagined, in cases where the domains are very far apart, such an approach might lead to worse results than random classification.

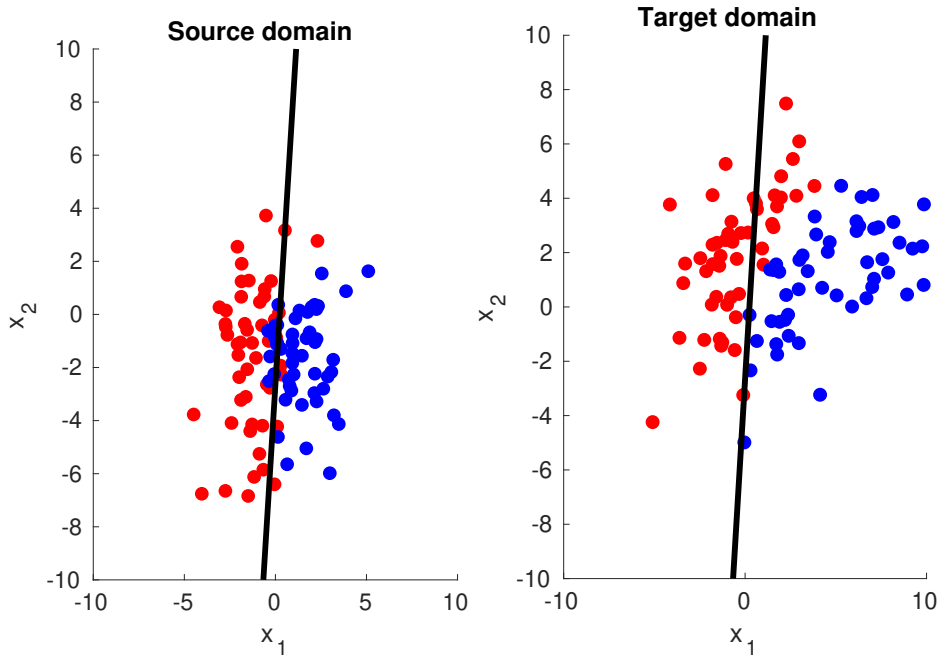


Figure 1.6: Example of a 2D domain adaptation problem. (Left) Data from the source domain, with a classifier (black line) trained to discriminate blue from red dots. (Right) Data from the target domain. Applying the classifier trained on the source domain leads to suboptimal results as it is misclassifying the top red dots.

1.4.1. Importance-weighting

Most importance weighting techniques are designed for covariate shift and most estimate the weights first, before training a weighted classifier. Figure 1.7 shows a scatterplot with weighted source samples. The dotted black line is the adapted classifier, trained on the importance-weighted source samples, and generalizes more to the target domain. Depending on the problem setting, some methods estimate the numerator and denominator of the ratio of probabilities separately, and others estimate the ratio directly. In this section, we discuss several of the most popular techniques.

In the sample selection bias setting, the target domain is the whole population, where each sample has probability 1 of being selected. That means that the numerator in the ratio of probability distributions is constant and it suffices to estimate the selection likelihood for the source samples. There has been a tremendous amount of work from the 1970s onwards in the statistics and social sciences communities that attempts to control for selection biases [14, 93]. Most of these approaches incorporated knowledge of the specific data

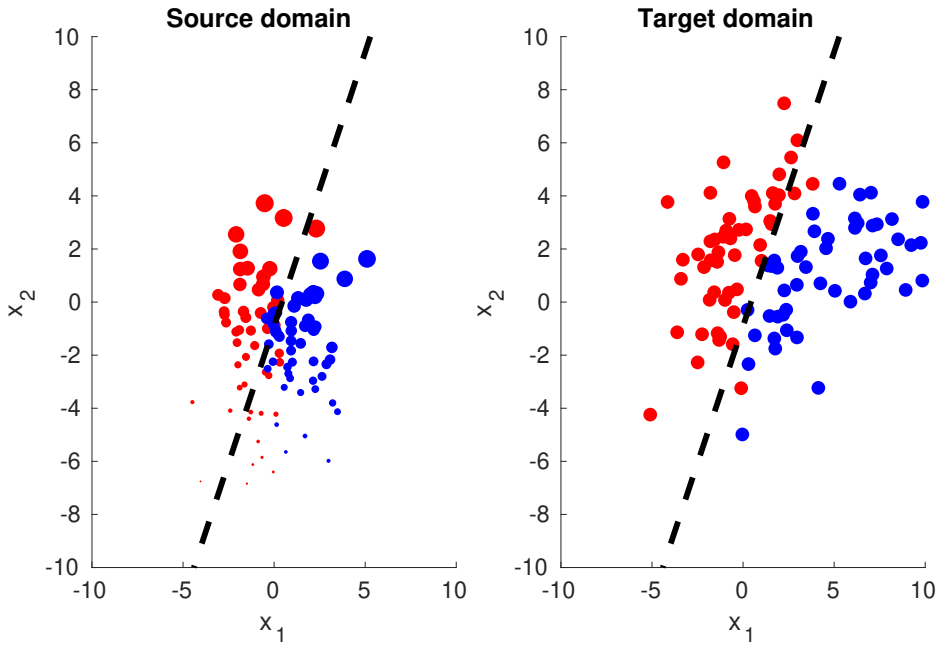


Figure 1.7: Example of importance-weighting. (Left) The source samples from Figure 1.6 have been weighted (larger dot size is larger weight) based on their relative importance to the target domain. The resulting importance-weighted classifier (black dotted line) deviates from the source classifier (solid black line in Figure 1.6). (Right) Applying the adapted classifier to the target samples leads to less misclassifications as compared to the original source classifier.

collection schemes, such as survey sampling , while others focused on estimating probabilities non-parametrically [120]. Knowing exactly how the sample space was discretized, for instance dividing up patients' age into intervals, can directly aid the estimation of the selection bias [121].

In settings with data missing-not-at-random (MNAR), some samples are more likely to be observed than others [95, 96]. This is essentially equivalent to the sample selection bias setting and in this case, one also aims to generalize to the case where all samples would be observed. However, this time, there may be prior knowledge available on what causes the missingness. This may be incorporated separately, with a model of how the data was generated [122, 123]. Given knowledge of how likely a sample is of being observed, also known as its propensity score $e(x) = p(\text{observed} \mid x)$, one can correct for the MNAR bias in the data [114, 124, 125]. Corrections are based on weighing each sample with its inverse propensity score $e(x)^{-1}$. These types of corrections are often employed in the causal inference community, where missingness arises in observational experimental studies [126–128]. From the causal inference community, they are now finding their way into machine learning as counterfactual risk minimization [129–132].

In general cases of covariate shift, the ratio of probability distributions are most often estimated as Gaussian distributions [133]. Unfortunately, closer inspection of families of probability distributions revealed that the use of exponential functions had a negative effect on the variance of the importance weights [74, 75, 134]. For example, if the source distribution is a univariate Gaussian distribution with mean 0 and variance 1, and the target distribution is a univariate Gaussian with mean 0 and variance $\sigma_{\mathcal{T}}^2$, then the weights consist of $p_{\mathcal{T}}(x)/p_{\mathcal{S}}(x) = \mathcal{N}(x \mid 0, \sigma_{\mathcal{T}}^2) \,/\, \mathcal{N}(x \mid 0, 1) = \sigma_{\mathcal{T}}^{-1} \exp(x^2(-1 + \sigma_{\mathcal{T}}^2)/(2\sigma_{\mathcal{T}}^2))$. For this example, if the target variance is larger than 2, then the variance of the weights, $\mathbb{E}_{\mathcal{S}}[(w(x) - \mathbb{E}_{\mathcal{S}}[w(x)])^2]$, diverges to infinity. Large weight variance means that it is highly probable that one sample will receive a very large weight, while the rest will receive very small weights. Consequently, at training time, the classifier will only pay attention to this one important sample and will neglect everything else. The resulting classifier is often pathological and will not generalize well. Alternatively, the distributions are often estimated through kernel density estimation [112, 135, 136].

Methods that directly estimate importance weights w, instead of the source $p_{\mathcal{S}}$ and target $p_{\mathcal{T}}$ distributions separately, are usually based on minimizing some type of discrepancy between the weighted source and the target distributions: $\mathrm{D}\left[w, p_{\mathcal{S}}, p_{\mathcal{T}}\right]$ [137]. However, just minimizing this objective with respect to w might cause highly varying or unusually scaled values, which would not be valid outcomes if we estimated the numerator and denominator separately [138]. This unwanted behaviour can be combated through incorporating a property of the reweighed source distribution:

$$\begin{aligned} 1 &= \int_{\mathcal{X}} p_{\mathcal{T}}(x)\mathrm{d}x \\ &= \int_{\mathcal{X}} w(x) p_{\mathcal{S}}(x)\mathrm{d}x \\ &\approx \frac{1}{n}\sum_{i=1}^{n} w(x_i) \qquad \text{for } x_i \sim p_{\mathcal{S}}\,, \end{aligned} \tag{1.15}$$

where the symbol $\sim$ refers to the fact that x_i are drawn from $p_{\mathcal{S}}$. Restraining the weight average to be close to 1, disfavors large values for weights. The approximate equality can be enforced by constraining the absolute deviation of the weight average to 1 to be less than some small value: $\mid n^{-1}\sum_i^n w(x_i) - 1 \mid \leq \epsilon$. Note that in the sample selection bias case, the inverse selection probability lies in the interval $[1, \infty)$, which will not average to 1. Incorporating the average weight constraint, along with the constraint that the weights should all be non-negative, direct importance weight estimation can be formulated as the following optimization problem:

$$\begin{aligned} \underset{w \in W}{\text{minimize}} \quad & \mathrm{D}\left[\, w,\, p_{\mathcal{S}},\, p_{\mathcal{T}} \,\right] \\ \text{s.t.} \quad & w(x_i) \geq 0 \\ & \mid n^{-1}\sum_{i}^{n} w(x_i) - 1 \mid \leq \epsilon\,. \end{aligned} \tag{1.16}$$

Depending on the choice of discrepancy measure, this optimization problem could be linear, quadratic or contain even more constraints.

One of the most common measures of distribution discrepancies is the Kullback-Leibler Divergence [139–141]. Sugiyama et al. have developed a number of techniques based on this formulation, among which the most famous is called the Kullback-Leibler Importance Estimation Procedure (KLIEP) [142–144]. The KL-divergence between the true target distribution and the importance-weighted source distribution can be simplified as:

$$\begin{aligned} \mathrm{D}_{\mathrm{KL}}\left[w, p_{\mathcal{S}}, p_{\mathcal{T}}\right] &= \int_{\mathcal{X}} p_{\mathcal{T}}(x) \log \frac{p_{\mathcal{T}}(x)}{p_{\mathcal{S}}(x) w(x)} \mathrm{d}x \\ &= \int_{\mathcal{X}} p_{\mathcal{T}}(x) \log \frac{p_{\mathcal{T}}(x)}{p_{\mathcal{S}}(x)} \mathrm{d}x - \int_{\mathcal{X}} p_{\mathcal{T}}(x) \log w(x) \mathrm{d}x \,. \end{aligned} \tag{1.17}$$

Since the first term in the right-hand side of (1.17) is independent of w, only the second term is used as in the optimization objective function. This second term is the expected value of the logarithmic weights with respect to the target distribution, which can be approximated with unlabeled target samples: $\mathbb{E}_{\mathcal{T}}[\log w(x)] \approx m^{-1} \sum_j^m \log w(z_j)$. They formulated w as a functional model consisting of an inner product of weights α and basis functions ϕ, i.e. $w(x) = \alpha^{\mathsf{T}} \phi(x)$ [145]. This allows them to apply the importance-weight function to both the test samples in the KLIEP objective from (1.17) and to the training samples for the constraint in (1.15).

Additionally, the group of Sugiyama has also produced another approach to direct estimation of the importance weights [146, 147]. They formulated the weights as a functional model again and formed an objective function based on minimizing the squared error between the estimated weights and the actual ratio of distributions:

$$\begin{aligned} \mathrm{D}_{\mathrm{LS}}[w, p_{\mathcal{S}}, p_{\mathcal{T}}] &= \frac{1}{2} \int_{\mathcal{X}} \left(w(x) - \frac{p_{\mathcal{T}}(x)}{p_{\mathcal{S}}(x)} \right)^2 p_{\mathcal{S}}(x) \mathrm{d}x \\ &= \frac{1}{2} \int_{\mathcal{X}} w(x)^2 p_{\mathcal{S}}(x) \mathrm{d}x - \int_{\mathcal{X}} w(x) p_{\mathcal{T}}(x) \mathrm{d}x + \text{constant} \,. \end{aligned} \tag{1.18}$$

As this squared error is used as an optimization objective function, the constant term drops out. We are then left with the expected value of the squared weights with respect to the source distribution, and the expected value of the weights with respect to the target distribution. Expanding the weight model, $w(x) = \alpha^{\mathsf{T}} \phi(x)$, gives $1/2\, \alpha^{\mathsf{T}} \mathbb{E}_{\mathcal{S}}[\phi(x)\phi(x)^{\mathsf{T}}]\alpha - \mathbb{E}_{\mathcal{T}}[\phi(x)]$. Replacing the expected values with sample averages allows for plugging in this objective into the nonparametric weight estimator in (1.16). The authors have dubbed this technique the Least-Squares Importance Fitting procedure.

Another very popular measure of domain discrepancy is the Maximum Mean Discrepancy, which is based on the two-sample problem from statistics [148–150]. Fortet and Mourier originally formulated a hypothesis test to see if two sets of samples came from the same distribution. It measures the distance between the means after subjecting the

samples to the continuous function that pulls them maximally apart (hence the name). In order to actually compute the measure, functions from a Reproducing Kernel Hilbert Space (RKHS) are used instead, which, under certain conditions, are able to approximate any continuous function arbitrary well [151–154]. Furthermore, for the subset of functions that are bounded above, the maximization operation can be subsumed in the RKHS norm [150]. As such, the discrepancy measure, including the reweighed source samples, can be expressed as [155]:

$$\mathrm{D}_{\mathrm{MMD}}\left[w, p_{\mathcal{S}}, p_{\mathcal{T}}\right] = \| \mathbb{E}_{\mathcal{S}}\left[w(x)\phi(x)\right] - \mathbb{E}_{\mathcal{T}}\left[\phi(x)\right] \|_{\mathcal{H}}\,, \tag{1.19}$$

where $\| \cdot \|_{\mathcal{H}}$ denotes the norm in RKHS space [154]. Basis functions from RKHS can be infinitely-dimensional, but by taking the square of the MMD one takes their inner products instead, which is again finite-dimensional. This is known as the *kernel trick* [156–159]. Through kernels the squared empirical MMD can be computed as: $\hat{\mathrm{D}}^2_{\mathrm{MMD}}[w, X, Z] = n^{-2}\sum_{i,i'}^{n} w(x_i)\kappa(x_i, x_{i'})w(x_{i'}) - 2/(mn)\sum_{i}^{n}\sum_{j}^{m} w(x_i)\kappa(x_i, z_j) + m^{-2}\sum_{j}^{m}\kappa(z_j, z_{j'})$. Minimizing the empirical MMD with respect to the importance weights, is called Kernel Mean Matching (KMM) [155, 160]. Depending on if, and how, the weights are upper bounded, algorithmic computational complexities and convergence criteria for KMM can be computed as well [136, 150, 155]

Taking a different direction, Bickel et al.'s work focuses on modeling the data generation process and working with domain selection variables [135, 161]. They reformulated the ratio of probability distributions as a selection likelihood ratio, $p(s = 1)/p(s = 1 \mid x)$, for which no explicit modeling of the separate probability distributions is necessary. Modeling this likelihood ratio with a kernel logistic model leads to a consistent estimator for the weights [161]. Through their generative modeling, the authors are able to combine the weight estimation and the weighted classifier training into a single optimization procedure [162]. For some experiments, the integrated models outperformed the two-step approach of estimating the selection likelihood ratio with a classifier and training an importance-weighted classifier [135]. But for other experiments, there was no difference between simultaneous and separate optimization. Their formulation also sheds new light on KMM, as it can also be re-formulated as a selection likelihood ratio estimator [135].

Lastly, directly estimating importance weights can also be done through tessellating the feature space into Voronoi cells [163]. Each cell is a polygon of variable size and denotes an area of equal probability. The cells approximate a probability distribution function in the same way that a multi-dimensional histogram does: with more Voronoi cells, one obtains a more precise description of the change in probability between neighbouring samples. Voronoi tesselations, and more general spacing estimators, have been used as empirical multi-dimensional density and entropy estimators [164, 165]. However, [166] uses them for estimating importance-weights. First, one forms the Voronoi cell V_i of each source sample x_i, which consists of the part of feature space that lies closest to x_i. The ratio of target over source is then approximated by counting the number of target samples z_j that lie within each Voronoi cell: $w(x_i) = |V_i \cap \{z_j\}_{j=1}^{m}|$, where $\cap$ denotes the intersection between the Voronoi cell and the set of target samples and $|\cdot|$ denotes the cardinality of this set.

Voronoi cells can be obtained through a 1-nearest-neighbour classifier, which means it is less computationally expensive than the discrepancy-based direct weight estimators. This is also where it lends its name from: nearest neighbour weighting (NNeW) [166]. It does not require hyperparameter optimization, but one still has the option to perform Laplace smoothing, the simplest one of which adds a 1 to each cell [167]. This counters the variance of the weights and ensures that no source samples are given a weight of 0 and are thus completely discarded.

1.4.2. Subspace mapping

In situations where the acquisition device noisily samples an object, domains may lie in different subspaces [98, 106]. In cases where cameras have the same resolution, and therefore measure the same feature space, there potentially exists a mapping from one domain to the other [62, 68]. For example, the mapping may correspond to a rotation, an affine transformation, or a more complicated nonlinear transformation [97, 98]. Figure 1.8 visualizes a translation and rotation from the source to the target domain, as well as the resulting classifier. Sometimes, such as for online product images and natural images , the domains look completely different from each other and the underlying mapping can be very complicated. Using too flexible transformations can easily lead to overfitting which means these methods will work well on the given target samples but fail for new target samples. Also, any structural relationships between domains, such as equal class-posterior distributions will most likely not be valid anymore after applying subspace mappings. Finally, the techniques for finding these transformations are unsupervised and ignore class information. That can be dangerous because it potentially introduces class overlap.

The simplest technique for finding a subspace mapping is to take the principal components in each domain, $C_{\mathcal{S}}$ and $C_{\mathcal{T}}$, and find the rotation from source to target $C_{\mathcal{S}} C_{\mathcal{S}}^{\top} C_{\mathcal{T}}$ [98]. However, it is likely that a portion of the components are purely based on noise. Including these into the rotation estimation step might cause overfitting. Luckily, this Subspace Alignment (SA) approach can also be used to find an subspace dimensionality parameter; a lower dimension means less parameters which means less overfitting. Additionally, this technique is attractive because its limited flexibility also means that it is quite robust to unusual problem settings. It is computationally not very expensive, easily implemented and intuitive to explain. Because of these attractive properties, it has been extended by other researchers a couple of times. For instance, there is a landmark-based kernelized alignment [168] and a subspace distribution alignment technique [169].

Before Subspace Alignment, there was another method based on principal components [97, 170]. First, the MMD measure is rewritten as a joint domain kernel, $\mathrm{K} = [\kappa_{\mathcal{S},\mathcal{S}}\ \kappa_{\mathcal{S},\mathcal{T}};\ \kappa_{\mathcal{T},\mathcal{S}}\ \kappa_{\mathcal{T},\mathcal{T}}]$ [171]. From this kernel, components are extracted by minimizing the trace of the projection, under the constraint that the projection applied to the centered joint kernel is equivalent to the identity matrix:

$$\begin{aligned} \underset{C}{\text{minimize}} \quad & tr(C^{\top}\mathrm{KLK}C) \\ \text{s.t.} \quad & C^{\top}\mathrm{KHK}C = I\,, \end{aligned} \tag{1.20}$$

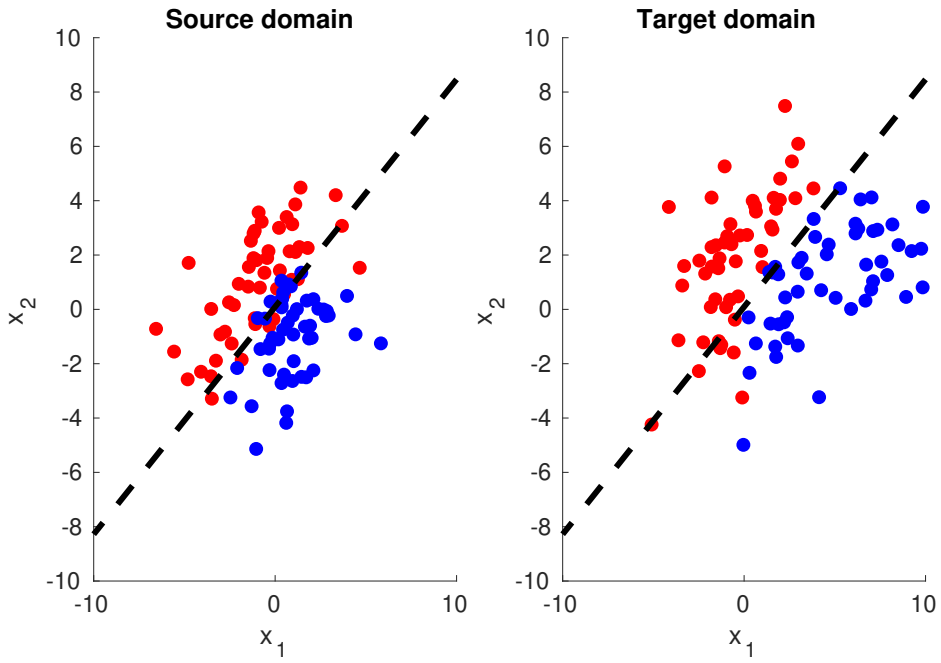


Figure 1.8: Example of subspace mapping. (Left) The source samples from Figure 1.6 have been translated and rotated to match the data from the target domain. Subsequently, a classifier is trained on the mapped source data (black dotted line). (Right) Applying the adapted classifier to the target samples leads to less misclassifications as compared to the original source classifier.

where $tr(\cdot)$ is shorthand for the trace of a matrix, C corresponds to the component matrix, L the normalization matrix that divides each entry in the joint kernel by the sample size of the domain from which it originated, and H is the matrix that centers the joint kernel matrix K [97]. In the original formulation, a regularization term $tr(C^{\top}C)$ along with a trade-off parameter is included as well. Essentially, the projection error is minimized, under the constraint that the projected joint kernel matrix is orthonormal. This formulation resembles kernel PCA and, likewise, its optimization resembles an eigenvalue decomposition [172, 173].

The advantage of principal component based techniques is that it is possible to map data to lower-dimensional representations. Lower dimensionalities mean that these algorithms scale well to large datasets. Furthermore, several researchers have argued that in computer-vision settings there exists a specific lower-dimensional subspace that allows for maximally discriminating target samples based on source samples. The Transfer Subspace Learning approach aims to find the subspace with the minimal Bregman divergence to both domains [174]. Their idea was later generalized by re-formulating the objective as the subspace from which the reconstruction error was minimal [175]. First, the source data is mapped to a lower-dimensional representation, and then mapped back to the original dimensionality. The reconstruction error then consists of the mismatch between the re-

constructed source samples and the target samples, measured through the squared error or the Frobenius norm for instance. Interestingly, this objective is very similar to that of a contractive autoencoder, where the inverse mapping is restricted to be the transpose of the original mapping: $\|\phi(\phi(xW)W^\top) - z\|$ [176]. Contractive and denoising autoencoders are deep learning methods, that can stack multiple layers of projecting and reconstructing samples on top of each other. Stacking allows them to achieve very flexible transformations [177]. [178] implemented a stacked autoencoder to find a mapping from source to target domain. This approach works well when large amounts of source data are available, as the overall transformation can be made more complex. It has not only been successful for adaptive computer vision, but for adaptive natural language processing as well . The computations for reconstructing in each layer were later simplified by through noise marginalization [179].

Outside of principal components based techniques and methods for learning transfer subspaces, there are also a number of methods that aim to find transformations that aid specific subsequent classifiers [180–182]. First, we review Information-Theoretic Metric Learning (ITML), where a Mahalanobis metric, i.e. $d_W^2(x,z) = (x-z)^\top W(x-z)$, is learned for use in a later nearest-neighbour classifier [180, 183]. Metrics describe ways of computing distances between points in vector spaces. If the standard Euclidean metric, $d_E^2(x,z) = (x-z)^\top(x-z)$, states that the distance between points x and z is large, but the Mahalanobis metric states that the distance is small, then one could say that the Mahalanobis metric transformed the space. In fact, first transforming the space and then measuring distances with the Euclidean metric is equivalent to measuring distances with the Mahalanobis metric: $(W^{1/2}x - W^{1/2}z)^\top(W^{1/2}x - W^{1/2}z) = (x-z)^\top W(x-z)$. In order to use the Mahalanobis metric for classification, it is necessary to include some constraints [184]. If a small number of target labels is available, then these correspondence constraints would consist of thresholding the pairwise distance between source and target samples of the same label, $d_W^2(x_k, z_k) \leq u$ with u as an upper bound, as well as thresholding the pairwise distance between source and target samples of different classes, $d_W^2(x_k, z_{k'}) \geq l$ with l as a lower bound. This ensures that the learned metric regards samples of the same class but different domains as similar, while regarding samples of different classes as dissimilar. If no target labels are available, then one is required to encode similarity in other ways.

ITML is restricted to finding transformations between domains in the same feature space. However, sometimes different descriptors are used for different image databases. One descriptor might span a source feature space of dimensionality D_S while another spans a feature space of dimensionality D_T. The symmetric ITML approach can be extended to an asymmetric case, where $d_W(x,z) \neq d_W(z,x)$. Asymmetric Regularized Cross-domain transfer (ARC-t) incorporates non-square metric matrices $W^{D_S \times D_T}$ to find general mappings between feature spaces of different dimensionalities [181, 185].

Instead of finding a unsupervised metric and constraining it with respect to class similarity, one could learn a supervised metric as well. The Fisher criterion consists of the ratio of between-class scatter, $S_B = \sum_k^K \pi_k(\mu_k - \bar{\mu})(\mu_k - \bar{\mu})^\top$ with μ_k the mean of the k-th class,

1

$\bar{\mu}$ the overall mean and π_k the class-prior, and average within-class scatter, $S_W = \sum_k^K \pi_k \Sigma_k$ with Σ_K as the covariance matrix of the k-th class [186, 187]. The Fisher criterion can be used to extract a set of basis vectors that maintain class separability, much like a supervised form of PCA [188]. It is often used in feature extraction, where the criterion is maximized with respect to a mapping to a lower-dimensional representation that maintains as much class separability as possible. They have been adapted to account for domain shift in the form of FIDOS, a FIsher based feature extraction method for DOmain Shift [189]. FIDOS incorporates multiple source domains and creates a between-class scatter matrix of the weighted average of the between-class scatter matrices in each domain as well as a within-class scatter matrix of the weighted average of each domains within-class scatter. It aims to both maximize class-separability and minimize domain differences, with a trade-off parameter to fine-tune the balance between these two objectives for particular settings. Besides metrics for subsequent metric-using classifiers, there are also techniques that align class margins for subsequent maximum-margin classifiers [190].

1.4.3. Domain manifolds

One can make stronger assumptions than the existence of a subspace mapping: that there exists a domain manifold [112, 191–193]. A manifold is a curved lower-dimensional subspace embedded in a larger vector space. Every point on a domain manifold generates a single domain. For example, the domain manifold might consist of a set of camera optics parameters, where each setting would measure the data in one vector space basis [111, 112]. This assumption is useful because it signifies that there exists a path along that manifold, going from the source domain to the target domain [194, 195]. In turn, this implies that there exists an intermediate domain for every step along the path [196–198]. Figure 1.9 visualizes what a classifier trained on interpolated domains might look like. Additional samples have been drawn from the supposed intermediate domains.

One of the first approaches to incorporate manifolds looked at the idea of learning incremental small subspace transformations instead of a single large transformation [111]. This idea of *incremental learning* was originally explored in situations with time-dependent concept shifts [199]. In the domain adaptation context, the goal is to learn the most likely intermediate subspaces between the source and target domain. The space of all possible D-dimensional subspaces in an n-dimensional real-valued vector space $\mathbb{R}^n$ can be described by the *Grassmann* manifold [200–203]. Each point on the Grassmannian generates a basis that forms a subspace [196, 197, 203]. One can move from one subspace to another by following the geodesic path. Computing the direction and speed of geodesic flow is performed based on the matrix exponential flow of the starting subspace [204]. Another option would be to compute the sampling spline flow [195]. Given the geodesic flow, all intermediate subspaces are computed and the source data is projected onto each of them separately. A classifier then trains on labeled data from a starting subspace and predicts labels for the next subspace, which are used as the labeled data in the next step. This process of inferring labels for every following step resembles *self-learning* in semi-supervised learning, where one iteratively labels unlabeled samples with the classifiers prediction and incorporates them in the next training stage [205]. The iteration was later proven redundant by a technique called Geodesic Flow Kernel (GFK), that incorporates the projections

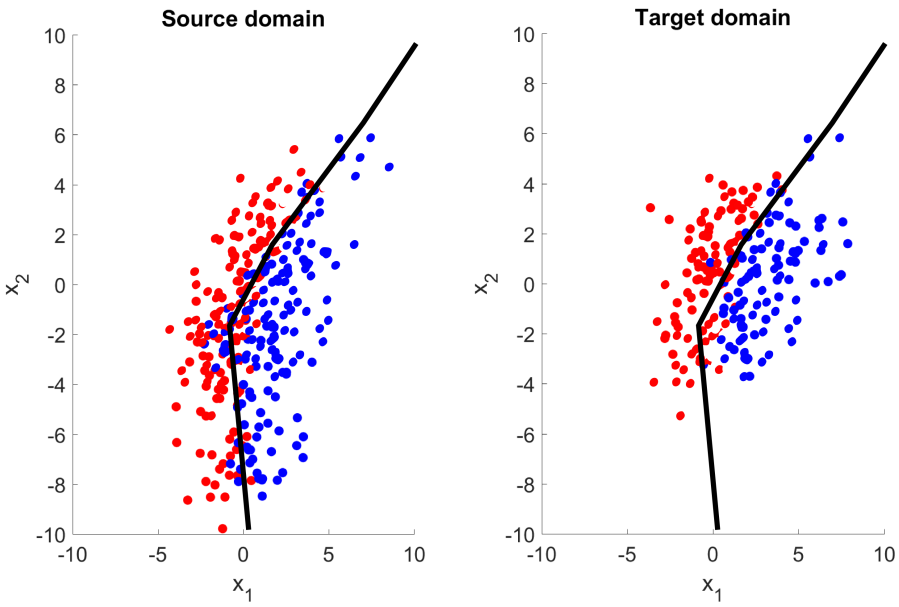


Figure 1.9: Example of interpolating along a domain manifold. (Left) Additional samples have been drawn from distributions that are believed to be intermediate domains between the source and the target domain. A classifier is now trained on both the source and the additional samples (black line). The resulting classifier (black dotted line) does not deviate much from the original classifier in the part of feature space where the source domain resides, but does deviate along the interpolated domains. (Right) Applying the adapted classifier to the target samples leads to less misclassifications around the target domain, but results in a few misclassifications around the source domain part of feature space.

on all subspaces into a single training stage [196, 197]. Naturally, it will be hard to recover the true path, but having multiple source domains or a small number of target labels will benefit geodesic flow estimation [193].

Working with Grassmann manifolds for subspace mappings is just one option. Alternatively, one could look at statistical manifolds, where each point generates a probability distribution [112]. Especially for rich families of distributions, such as the exponential family, a path on the statistical manifold may describe a complicated process of turning one distributions into another. In this case, the length of the geodesic path along the statistical manifold, called the Hellinger distance, is used as a measure of domain discrepancy [104, 206]. The Hellinger distance is closely related to the total variation distance between two distributions, which is used in a number of other domain adaptation works without reference to the statistical manifold [17, 72, 207]. Adaptation consists of either importance-weighting samples or transforming parameters to minimize the Hellinger distance [112, 208].

1.4.4. Domain-invariance

The problem with transformations between domains or moving along a domain manifold is that the classifier remains in a domain-specific representation. But variation due to do-

mains is often more of a nuisance instead of an interesting factor. Ideally, we would like to represent the data in a space which is invariant to specific domains. Figure 1.10 shows an example of such a space: in this case the data can be mapped to a line (new 1-dimensional representation) such that the source and target distributions vary only minimally, while the variation between classes (red versus blue) is still the same. Note that a domain-invariant representation need not be lower dimensional. The advantage of this approach is that classification is now the same as in standard supervised learning (training on source data and applying to target data).

Most domain-invariant projection techniques stem from the computer vision and (biomedical) image processing communities, where domains are often caused by different acquisition methods. For instance, in computer vision, camera-specific variation between sets of photos is an unwanted factor of variation [190, 194]. Furthermore, we could argue that there exists a true representation of the object and that each camera is a different noisy representation of it. Similarly, in medical imaging, there exists a true representation of a patient and each MRI scanner is a different noisy representation [18, 20]. Since there is extensive knowledge of the acquisition device, it is possible to design specific techniques that work quite well.

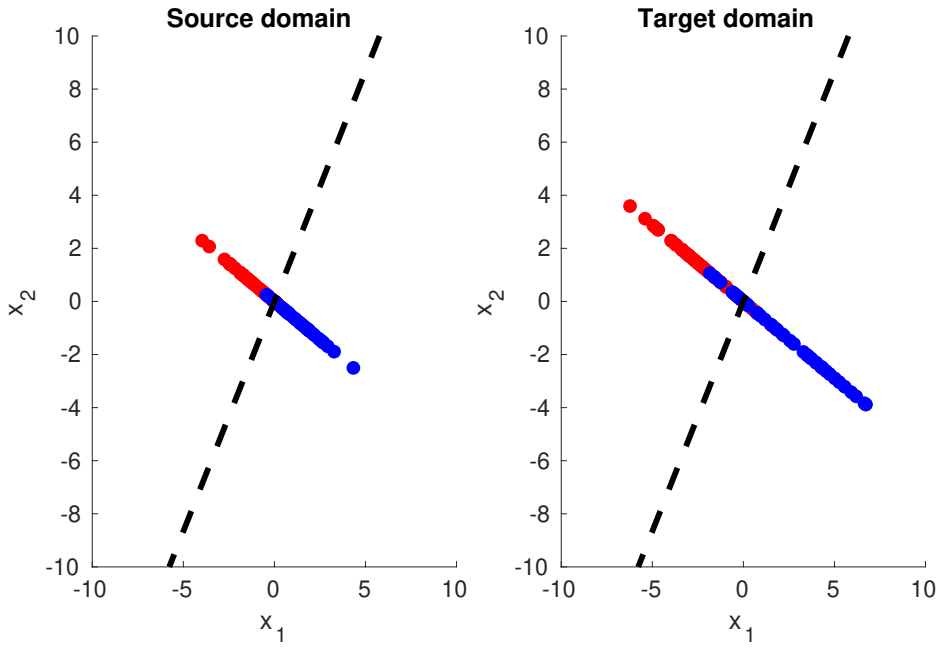


Figure 1.10: Example of a domain-invariant representation. (Left) The source data is mapped to a representation in which the variation over domains is minimal; the source data and target lie on the same 1-dimensional (line). The variation over classes is maintained; the red and blue dots are still as separable as before. The classifier trained on the source data in the domain-invariant representation (dotted black line) can now directly be applied to the target samples (right).

Learning the projection to the domain-invariant can be done in a number of ways. The Domain-Invariant Projection (DIP) approach from [194] , later renamed to Distribution-Matching Embedding (DME) [208], aims to find a projection matrix that minimizes the MMD:

$$\mathrm{D}_{\mathrm{DME}}[W, p_{\mathcal{S}}, p_{\mathcal{T}}] = \| \mathbb{E}_{\mathcal{S}}[\phi(W^{\mathsf{T}}x)] - \mathbb{E}_{\mathcal{T}}[\phi(W^{\mathsf{T}}x)] \|_{\mathcal{H}} , \tag{1.21}$$

where W is the projection matrix that is being minimized over, with the additional constraint that it remains orthonormal; $W^{\mathsf{T}}W = I$. This constraint is necessary to avoid pathological solutions to the minimization problem. However, although the MMD encourages moments of distributions to be similar, it does not encourage smoothness in the new space. To this end, it is possible to add a regularization term that punishes the within-class variance in the domain-invariant space, to encourage class clustering. Alternatively, the same authors have also proposed the same technique, but with the Hellinger distance instead of the MMD [208]. This approach resembles Transfer Component Analysis, but minimizes discrepancy instead of maximizing joint domain variance [97].

DME is still limited by its use of a linear projection matrix; unless the specific acquisition device only causes linear noise, a linear projection matrix will not be able to find the true underlying domain-invariant representation. A nonlinear projection is much more flexible and much more likely to recover the true domain-invariant space. The Nonlinear Distribution-Matching Embedding achieves this additional flexibility by performing the linear projection in kernel space; $W^{\mathsf{T}}\phi(x)$ [208]. However, using a kernel-within-kernel approach is expensive in terms of memory and computational resources.

Alternatively, [209] proposed to learn the kernel for MMD itself: instead of weighting or projecting samples and then using a universal kernel to measure their discrepancy, it is also possible to find a basis function for which the two sets of distributions are as similar as possible. The space spanned by this learned kernel then corresponds to the domain-invariant space. Considering that different distributions generate different means in kernel space, it is possible to describe a distribution of kernel means [154, 210]. The variance of this meta-distribution, termed *distributional variance*, should then be minimized. This is achieved by incorporating a lower-dimensional orthogonal transform into the inner product of the basis functions, also known as the Gram matrix, and minimizing the empirical distributional variance with respect to this transform matrix [211]. However, this is fully unsupervised and could introduce class overlap. The functional relationship between the input and the classes can be preserved by incorporating a central subspace in which the input and the classes are conditionally independent [212, 213]. Constraining the optimization objective with maintaining this central subspace, ensures that classes remain separable in the new domain-invariant space. Overall, as this approach is interpreted as finding kernel components that minimize distributional-variance while maintaining the functional relationship, it is coined Domain-Invariant Component Analysis (DICA) [209]. It has been expanded on for the specific case of spectral kernels by [214].

Although the kernel approaches have the capacity to recover any nonlinear mapping, they require multiple computations of $n \times n$ matrices and therefore do not scale well with

respect to the number of samples. For larger datasets, one might employ neural networks since they also have the capacity to recover any nonlinear mapping but scale much better in terms of the number of samples [215, 216]. Neural networks are layered, and when going from one layer to the next, the input representation is transformed using a linear operation and pushed through a nonlinear activation function. By increasing the complexity of a layer and stacking multiple layers on top of each other, under certain conditions, any possible transformation can be achieved [55, 217–219]. Its optimization procedure, known as backpropagation, pushes the network to find a transformation that maps the data into a space in which it is maximally linearly separable. Fortunately, by using different loss functions in the top layer, we can achieve different forms of transformations [220]. Domain-Adverserial Neural Networks (DANN) have one classification-error minimizing top layer and one domain-error maximizing top layer [221]. Essentially, the network finds a domain-invariant space when its domain classifier cannot recognize from which domain a new sample has come, without introducing class overlap. The idea of maximizing domain-confusion while minimizing classification error has been taken up and applied to various settings by a number of approaches [222, 223].

Finally, if we were to have a small collection of target labels, then it might be possible to compare the classifiers found in each domain and transform the space such that these become as similar as possible [224]. The formulation here consists of encoding an additional variable denoting to which domain each sample belongs, along with its corresponding modeled class-posterior distribution, and marginalizing this variable out. As more domains imply more restrictions on the possible transformations, this method benefits from incorporating multiple sources.

Mapping data to a domain-invariant space resembles the setting where we are correcting for sample selection bias. Essentially, the domain-invariant space corresponds to the super-population and our data mapped to the domain-invariant space corresponds to an unbiased sampling from this super-population. They differ perhaps, in that, in the selection bias setting, certain samples are very common in one dataset and rare in another, while, in the domain-invariance setting, an object might be equally common in both datasets, but look different in each of them.

1.4.5. Feature augmentation

In natural language processing (NLP), domains present themselves in the form of differences between word frequency statistics over text corpora. This happens mostly because people express themselves differently in different contexts [225, 226]. For instance, the word ‘useful’ occurs more often to denote positive sentiment in kitchen appliance reviews than in book reviews [34]. It can even happen that certain words occur *only* in particular contexts, such as ’opioid receptors’ in abstracts of biomedical papers but not in financial news [108]. Hence, domains present a large problem for the field.

Fortunately, NLP systems can exploit the fact that words tend to signal each other; in a *bag-of-words* (BoW) encoding each document is described by a vocabulary and a set of word counts [227]. Words that signal each other, tend to occur together in documents.

Co-occurring words lead to correlating features in a BoW encoding. Correlating features can be exploited as follows: suppose a particular word is a strong indicator of positive or negative sentiment and only occurs in the source domain. Then one could find a correlating "pivot" word that occurs frequently in both domains, and find the word in the target domain that correlates most with the pivot word. This target domain word, is most likely the corresponding word to the original source domain word and will be a good indicator of positive versus negative sentiment as well [32]. Thus, by augmenting the bag-of-words encoding with pivot words and learning correspondences, it is possible to overcome domain differences [32, 108, 228, 229]. Figure 1.11 shows an idealized example of augmenting a feature space. x_3 is added to both the source (left) and the target (right) domain. This additional feature allows for training a classifier (grey plane) that will perform as well on the source domain as on the target domain, regardless of the initial differences between them.

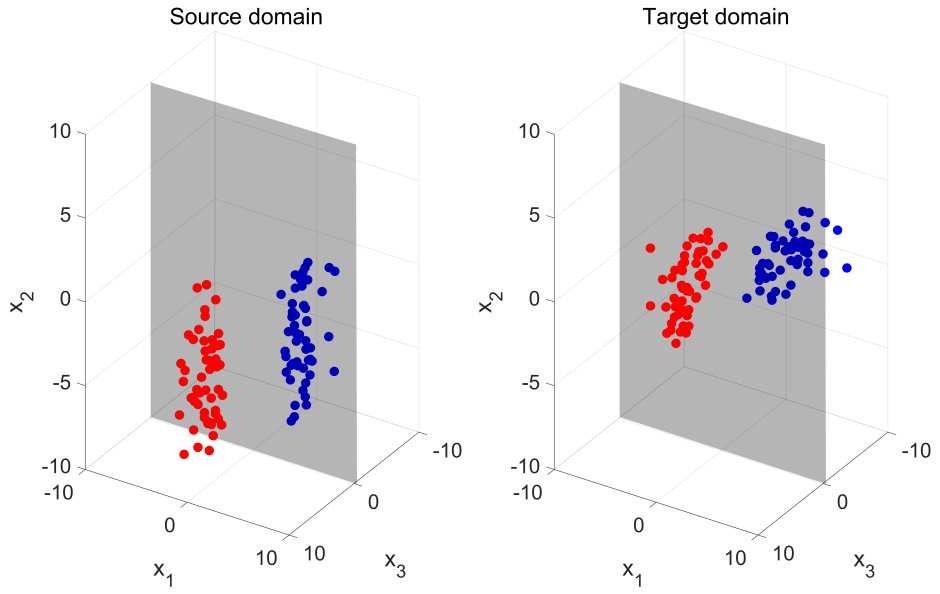


Figure 1.11: Example of an augmented feature space. (Left) The source domain is augmented with an additional feature (x_3) that does not vary over domains. (Right) The target domain is augmented with the same feature. Now, regardless of the initial difficulty of adaptation, a classifier (grey plane) can be trained that generalizes well.

How to find corresponding features, or in general how to couple subspaces, is an open question. Note that with more features, there is a larger chance to find a good pivot feature. The earliest approaches have extracted pivot features through joint principal components or maximizing cross-correlation [34, 108]. However, such techniques are linear and can only model linear relationships between features. Later approaches are more nonlinear through the use of kernelization or by employing "co-training" approaches [228, 229].

Feature augmentation shares a lot of overlap with the approach of finding a domain-invariant space, as the augmentation is essentially an artificially created invariant space. The key difference lies in the fact that retaining the original features can be helpful when there indeed exist correspondences between features. Additionally, feature augmentation also shares overlap with the subspace mapping setting as the domain-specific features are now different subspaces of the overall feature space. For example, in bag-of-word encodings, target samples essentially have value 0 on words that only occur in the source domain. If we were to encode all samples in the union of words, then the source domain could be mapped to the target domain by a transformation through the domain-specific part of the total feature space.

1.4.6. Robust adaptation

When any of the aforementioned approaches are applied to settings where their assumed shift is actually not happening, then they tend to mis-estimate how to adapt and perform terribly [230, 231]. In short, they are not robust to unexpected changes. Although some methods are more robust to invalid assumptions, such as maximum-margin based classifiers, others can be made more robust [232]. To ensure a robust level of performance, a few approaches assume worst-case settings. Worst-case settings are often formalized as minimax optimization problems [40, 233]. However, as there are no target labels, the worst-case is not unique and no optimization algorithm will converge. Considering that there is additional information in the form of the source domain, it is possible to constrain the worst-case setting. One of the most interesting approaches here is the use of worst-case importance weights [109]:

$$(\hat{h}, \hat{w}) = \arg\min_{h \in \mathcal{H}} \arg\max_{w \in W} \frac{1}{n} \sum_{i=1}^{n} \ell(h(x_i), y_i) w(x_i) \,. \tag{1.22}$$

By training a classifier under worst-case weights, it should be more robust against the adverse effects of density ratio estimation errors.

Another minimax strategy, dubbed the robust bias-aware classifier [110], plays a game between a risk minimizing target classifier and a risk maximizing target class-posterior distribution, where the adversary is constrained to pick posteriors that match the moments of the source distribution statistics. The constraint is important, as the adversary would otherwise be able to design posterior probabilities that result in degenerate classifiers (e.g. assign all class-posterior probabilities to 1 for one class and 0 for the other). Effectively, this means the minimax estimator returns high confidence predictions in regions with large probability mass of the source density and uniform class predictions in regions with small source probability mass. This behaviour nicely reflects the larger difficulty of an adaptation problem with a larger domain dissimilarity, but restricts this approach to problems where the probability masses of both domains overlap to some extent. However, it also means that their approach loses predictive power in areas of feature space where the source distribution has limited support, and thus is not suited very well for problems where the domains are very different.

Conservatism can also be expressed through *algorithmic robustness* [234]. An algorithm is deemed robust if it can separate a labeled feature space into disjoint sets such that the variation of the classifier is bounded by a factor dependent on the training set. Intuitively, a robust classification algorithm does not change its predictions much whenever a training sample is changed. Separating the space with a robust algorithm implies that the loss is bounded in each partition, regardless of the distribution of samples. [207] employs this notion to construct a robust adaptive algorithm. They introduce λ-shift, a measure of how far the value of the target class posterior probability differs from the source class posterior probability, which is used as a constraint on the loss on target samples in a support vector machine formulation. The classifier finds a separating hyperplane such that the hinge loss on the source set is similar to the loss of the target domain. The downside of this approach is that if the class posterior distributions of both domains are very different (e.g. orthogonal decision boundaries), it will not perform well on *both* sets.

1

1.5. Contribution

Each chapter in this thesis builds upon different approaches to domain adaptation.

Chapters 2 and 3 are concerned with the popular importance-weighting technique and address two open questions regarding cross-validation under covariate shift. Chapter 2 shows that importance weighting the source validation dataset is not sufficient to obtain hyperparameters that are optimal in the target domain, while Chapter 3 extends this work by hypothesizing that this insufficiency is due to problems with unbounded variance of the importance weights. Experiments with reducing the sampling variance of the importance-weighted risk estimator show that the estimator improves, but still does not find the optimal regularization parameter.

Chapter 4 switches to the case of subspace mappings. We formulated a condition that, when fulfilled, allows for recovering the optimal target classifier. The difficulty lies in finding the correct parameterization of what we call a *transfer model*. Looking at a simple case of transfer, namely dropout (prevalent in cases of missing data at test time and bag-of-word encodings), we present an algorithm that estimates the transfer model's parameters and trains a classifier that ignores features that are not important in the target domain.

Chapter 5 combines machine learning with medical imaging. Tissue classifiers do not generalize well across MRI-scanners due to unknown acquisition-related variations. We tackle this problem by mapping images from different MRI-scanners to a domain-invariant representation. We used an MR simulator that allows us to vary scan sequence parameters on the same subjects, thereby isolating acquisition-related factors of variation. A Siamese convolutional neural network is used to learn the acquisition-invariant representation. Using a measure of distance between datasets, the proxy $\mathcal{A}$-distance, we are able to show that, in some cases, it can be very beneficial to add data from another scanner, while, in other cases, it can be very disruptive to training a tissue classifier.

Finally, Chapter 6 formulates a robust parameter estimator that is guaranteed to never perform worse than the source classifier on the target domain. When applied to the discriminant analysis framework, it even ensures that the resulting adaptive classifier will *always* perform better than the source classifier, in terms of model likelihood. No performance guarantees of this kind have been proposed before.

In closing, Chapter 7 reflects on some of the most important findings and discusses promising avenues for further research.

References

[1] M. I. Jordan and T. M. Mitchell, *Machine learning: trends, perspectives, and prospects*, Science **349**, 255 (2015).

[2] Z. Ghahramani, *Probabilistic machine learning and artificial intelligence*, Nature **521**, 452 (2015).

[3] K. He, X. Zhang, S. Ren, and J. Sun, *Delving deep into rectifiers: surpassing human-level performance on imagenet classification*, in *International Conference on Computer Vision* (2015) pp. 1026–1034.

[4] J. Wang, L. Zhang, D. Zhang, and K. Li, *An adaptive longitudinal driving assistance system based on driver characteristics*, IEEE Transactions on Intelligent Transportation Systems **14**, 1 (2013).

[5] A. Giusti, J. Guzzi, D. C. Cireşan, F.-L. He, J. P. Rodríguez, F. Fontana, M. Faessler, C. Forster, J. Schmidhuber, G. Di Caro, *et al.*, *A machine learning approach to visual perception of forest trails for mobile robots*, IEEE Robotics and Automation Letters **1**, 661 (2016).

[6] I. Kononenko, *Machine learning for medical diagnosis: history, state of the art and perspective*, Artificial Intelligence in Medicine **23**, 89 (2001).

[7] J. Lu, D. Wu, M. Mao, W. Wang, and G. Zhang, *Recommender system application developments: a survey*, Decision Support Systems **74**, 12 (2015).

[8] M. Wieland and M. Pittore, *Performance evaluation of machine learning algorithms for urban pattern recognition from multi-spectral satellite images*, Remote Sensing **6**, 2912 (2014).

[9] N. M. Ball and R. J. Brunner, *Data mining and machine learning in astronomy*, International Journal of Modern Physics D **19**, 1049 (2010).

[10] J. Zhang and C. Zong, *Deep neural networks in machine translation: an overview*, IEEE Intelligent Systems **30**, 16 (2015).

[11] S. J. Russell and P. Norvig, *Artificial intelligence: a modern approach*, (2009).

[12] C. M. Bishop, *Pattern recognition and machine learning* (Springer, 2006).

[13] M. Mohri, A. Rostamizadeh, and A. Talwalkar, *Foundations of Machine Learning* (MIT Press, 2012).

[14] J. J. Heckman, *Sample selection bias as a specification error (with an application to the estimation of labor supply functions)*, (1977).

[15] J. Heckman, *Varieties of selection bias*, American Economic Review **80**, 313 (1990).

[16] J. C. Helton, J. D. Johnson, C. J. Sallaberry, and C. B. Storlie, *Survey of sampling-based methods for uncertainty and sensitivity analysis*, Reliability Engineering & System Safety **91**, 1175 (2006).

[17] S. Ben-David, J. Blitzer, K. Crammer, A. Kulesza, F. Pereira, and J. W. Vaughan, *A theory of learning from different domains*, Machine Learning **79**, 151 (2010).

[18] A. van Opbroek, M. Ikram, M. Vernooij, and M. De Bruijne, *Transfer learning improves supervised image segmentation across imaging protocols*, IEEE Transactions on Medical Imaging **34**, 1018 (2015).

[19] M. Ghafoorian, A. Mehrtash, T. Kapur, N. Karssemeijer, E. Marchiori, M. Pesteie, C. R. Guttmann, F.-E. de Leeuw, C. M. Tempany, B. van Ginneken, *et al.*, *Transfer learning for domain adaptation in MRI: application in brain lesion segmentation*, arXiv preprint arXiv:1702.07841 (2017).

[20] W. Kouw, M. Loog, L. Bartels, and A. Mendrik, *Mr acquisition-invariant representation learning*, ArXiv (2017).

[21] C. Widmer, N. C. Toussaint, Y. Altun, O. Kohlbacher, and G. Rätsch, *Novel machine learning methods for MHC Class I binding prediction*, in *International Conference on Pattern Recognition in Bioinformatics* (Springer, 2010) pp. 98–109.

[22] S. Mei, W. Fei, and S. Zhou, *Gene ontology based transfer learning for protein subcellular localization*, BMC Bioinformatics **12**, 1 (2011).

[23] A. Chen and Z.-W. Huang, *A new multi-task learning technique to predict classification of leukemia and prostate cancer*, Medical Biometrics , 11 (2010).

[24] Q. Xu, H. Xue, and Q. Yang, *Multi-platform gene-expression mining and marker gene analysis*, International Journal of Data Mining and Bioinformatics **5**, 485 (2011).

[25] M. Nassar, R. Abdallah, H. A. Zeineddine, E. Yaacoub, and Z. Dawy, *A new multitask learning method for multiorganism gene network estimation*, in *International Symposium on Information Theory* (IEEE, 2008) pp. 2287–2291.

[26] T. Kato, K. Okada, H. Kashima, and M. Sugiyama, *A transfer learning approach and selective integration of multiple types of assays for biological network inference*, in *Computational Knowledge Discovery for Bioinformatics Research* (IGI Global, 2012) pp. 188–202.

[27] Q. Xu and Q. Yang, *A survey of transfer and multitask learning in bioinformatics*, Journal of Computing Science and Engineering **5**, 257 (2011).

[28] W. Caesarendra, G. Niu, and B.-S. Yang, *Machine condition prognosis based on sequential monte carlo method*, Expert Systems with Applications **37**, 2412 (2010).

[29] T. Van Kasteren, G. Englebienne, and B. J. Kröse, *Transferring knowledge of activity recognition across sensor networks*. in *Pervasive Computing*, Vol. 10 (Springer, 2010) pp. 283–300.

[30] H. Hachiya, M. Sugiyama, and N. Ueda, *Importance-weighted least-squares probabilistic classifier for covariate shift adaptation with application to human activity recognition*, Neurocomputing **80**, 93 (2012).

[31] E. Gedik and H. Hung, *Speaking status detection from body movements using transductive parameter transfer*, in *International Joint Conference on Pervasive and Ubiquitous Computing* (ACM, 2016) pp. 69–72.

[32] J. Blitzer, R. McDonald, and F. Pereira, *Domain adaptation with structural correspondence learning*, in *Empirical Methods in Natural Language Processing* (2006) pp. 120–128.

[33] V. Peddinti and P. Chintalapoodi, *Domain adaptation in sentiment analysis of twitter*, in *AAAI Conference on Artificial Intelligence* (2011).

[34] J. Blitzer, M. Dredze, F. Pereira, *et al.*, *Biographies, bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification*, in *Association for Computational Linguistics*, Vol. 7 (2007) pp. 440–447.

[35] R. He and J. McAuley, *Ups and downs: modeling the visual evolution of fashion trends with one-class collaborative filtering*, in *International Conference on World Wide Web* (International World Wide Web Conferences Steering Committee, 2016) pp. 507–517.

[36] M. T. Nuruzzaman, C. Lee, and D. Choi, *Independent and personal SMS spam filtering*, in *International Conference on Computer and Information Technology* (IEEE, 2011) pp. 429–435.

[37] C. Leggetter and P. Woodland, *Maximum likelihood linear regression for speaker adaptation of continuous density hidden markov models*, Computer Speech & Language **9**, 171 (1995).

[38] G. Zen, E. Sangineto, E. Ricci, and N. Sebe, *Unsupervised domain adaptation for personalized facial emotion recognition*, in *International Conference on Multimodal Interaction* (2014) pp. 128–135.

[39] E. Gedik and H. Hung, *Personalised models for speech detection from body movements using transductive parameter transfer*, Personal and Ubiquitous Computing , 1 (2017).

[40] J. O. Berger, *Statistical decision theory and Bayesian analysis* (Springer, 2013).

[41] J. Friedman, T. Hastie, and R. Tibshirani, *The elements of statistical learning*, Vol. 1 (Springer, 2001).

[42] S. Theodoridis, A. Pikrakis, K. Koutroumbas, and D. Cavouras, *Introduction to pattern recognition: a matlab approach* (Academic Press, 2010).

[43] S. Boyd and L. Vandenberghe, *Convex optimization* (Cambridge University Press, 2004).

[44] V. Vapnik, *The nature of statistical learning theory* (Springer, 2000).

[45] O. Bousquet and L. Bottou, *The tradeoffs of large scale learning*, in *Advances in Neural Information Processing Systems* (2008) pp. 161–168.

[46] P. Bartlett, *Prediction algorithms: complexity, concentration and convexity*, in *IFAC Symposium on System Identification* (2003) pp. 1507–1517.

[47] L. G. Valiant, *A theory of the learnable*, Communications of the ACM **27**, 1134 (1984).

[48] M. J. Kearns and U. V. Vazirani, *An introduction to computational learning theory* (MIT Press, 1994).

[49] P. L. Bartlett and S. Mendelson, *Rademacher and gaussian complexities: risk bounds and structural results*, Journal of Machine Learning Research **3**, 463 (2002).

[50] V. Vapnik and A. Y. Chervonenkis, *On the uniform convergence of relative frequencies of events to their probabilities*, Theory of Probability and its Applications **16**, 264 (1971).

[51] S. Bendavid, N. Cesabianchi, D. Haussler, and P. M. Long, *Characterizations of learnability for classes of (0,..., n)-valued functions*, Journal of Computer and System Sciences **50**, 74 (1995).

[52] D. McAllester, *Simplified PAC-Bayesian margin bounds*, Lecture Notes in Computer Science , 203 (2003).

[53] J. Langford and J. Shawe-Taylor, *PAC-Bayes & margins*, in *Advances in Neural Information Processing Systems* (2003) pp. 439–446.

[54] L. Bégin, P. Germain, F. Laviolette, and J.-F. Roy, *PAC-Bayesian theory for transductive learning*, in *Artificial Intelligence and Statistics* (2014) pp. 105–113.

[55] P. L. Bartlett, *The sample complexity of pattern classification with neural networks: the size of the weights is more important than the size of the network*, IEEE Transactions on Information Theory **44**, 525 (1998).

[56] P. L. Bartlett, M. I. Jordan, and J. D. McAuliffe, *Convexity, classification, and risk bounds*, Journal of the American Statistical Association **101**, 138 (2006).

[57] L. Devroye, L. Györfi, and G. Lugosi, *A probabilistic theory of pattern recognition*, (2013).

[58] Y. Freund and R. E. Schapire, *A desicion-theoretic generalization of on-line learning and an application to boosting*, in *European Conference on Computational Learning Theory* (Springer, 1995) pp. 23–37.

[59] C. Cortes and V. Vapnik, *Support vector machine*, Machine learning **20**, 273 (1995).

[60] A. Ehrenfeucht, D. Haussler, M. Kearns, and L. Valiant, *A general lower bound on the number of examples needed for learning*, Information and Computation **82**, 247 (1989).

[61] A. N. Tikhonov, *Solution of incorrectly formulated problems and the regularization method*, in *Soviet Mathematics Doklady*, Vol. 5 (1963) pp. 1035–1038.

[62] V. M. Patel, R. Gopalan, R. Li, and R. Chellappa, *Visual domain adaptation: a survey of recent advances*, IEEE Signal Processing Magazine **32**, 53 (2015).

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

[64] B. Saha, S. Gupta, D. Phung, and S. Venkatesh, *Transfer learning for rare cancer problems via discriminative sparse gaussian graphical model*, in *International Conference on Pattern Recognition* (2016) pp. 537–542.

[65] J. Kim and C. Park, *End-to-end ego lane estimation based on sequential transfer learning for self-driving cars*, in *Computer Vision and Pattern Recognition Workshops* (2017) pp. 1194–1202.

[66] D. McClosky, E. Charniak, and M. Johnson, *Automatic domain adaptation for parsing*, in *Conference of the North American Chapter of the Association for Computational Linguistics* (2010) pp. 28–36.

[67] A. Karpathy and L. Fei-Fei, *Deep visual-semantic alignments for generating image descriptions*, in *Conference on Computer Vision and Pattern Recognition* (2015) pp. 3128–3137.

[68] R. Gopalan, R. Li, V. M. Patel, R. Chellappa, *et al.*, *Domain adaptation for visual recognition*, Foundations and Trends ® in Computer Graphics and Vision **8**, 285 (2015).

[69] W. Hoeffding, *Probability inequalities for sums of bounded random variables*, Journal of the American Statistical Association **58**, 13 (1963).

[70] S. Boucheron, G. Lugosi, and P. Massart, *Concentration inequalities: a nonasymptotic theory of independence* (Oxford University Press, 2013).

[71] R. K. Ando and T. Zhang, *A framework for learning predictive structures from multiple tasks and unlabeled data*, Journal of Machine Learning Research **6**, 1817 (2005).

[72] S. Ben-David, J. Blitzer, K. Crammer, and F. Pereira, *Analysis of representations for domain adaptation*, in *Advances in Neural Information Processing Systems* (2007) pp. 137–144.

[73] J. Blitzer, K. Crammer, A. Kulesza, F. Pereira, and J. Wortman, *Learning bounds for domain adaptation*, in *Advances in Neural Information Processing Systems* (2008) pp. 129–136.

[74] C. Cortes, M. Mohri, M. Riley, and A. Rostamizadeh, *Sample selection bias correction theory*, in *Algorithmic Learning Theory* (Springer, 2008) pp. 38–53.

[75] C. Cortes, Y. Mansour, and M. Mohri, *Learning bounds for importance weighting*, in *Advances in Neural Information Processing Systems* (2010) pp. 442–450.

[76] C. Cortes, M. Mohri, and A. Muñoz Medina, *Adaptation algorithm and theory based on generalized discrepancy*, in *International Conference on Knowledge Discovery and Data Mining* (2015) pp. 169–178.

[77] Y. Mansour, M. Mohri, and A. Rostamizadeh, *Domain adaptation: Learning bounds and algorithms*, in *Conference on Learning Theory* (2009).

[78] S. Ben-David, T. Lu, T. Luu, and D. Pál, *Impossibility theorems for domain adaptation*, in *International Conference on Artificial Intelligence and Statistics* (2010) pp. 129–136.

[79] C. Zhang, L. Zhang, and J. Ye, *Generalization bounds for domain adaptation*, in *Advances in Neural Information Processing Systems* (2012) pp. 3320–3328.

[80] P. Germain, A. Habrard, F. Laviolette, and E. Morvant, *A PAC-Bayesian approach for domain adaptation with specialization to linear classifiers*, in *International Conference on Machine Learning* (2013) pp. 738–746.

[81] P. Germain, A. Habrard, F. Laviolette, and E. Morvant, *A new PAC-Bayesian perspective on domain adaptation*, in *International Conference on Machine Learning* (2016) pp. 859–868.

[82] D. Kifer, S. Ben-David, and J. Gehrke, *Detecting change in data streams*, in *International Conference on Very Large Data Bases* (VLDB Endowment, 2004) pp. 180–191.

[83] V. Vapnik and V. Vapnik, *Statistical learning theory*, Vol. 2 (Wiley New York, 1998).

[84] A. Storkey, *When training and test sets are different: characterizing learning transfer*, Dataset Shift in Machine Learning , 3 (2009).

[85] N. Japkowicz, C. Myers, M. Gluck, *et al.*, *A novelty detection approach to classification*, in *International Joint Conference on Artificial Intelligence*, Vol. 1 (1995) pp. 518–523.

[86] M. Kubat, R. C. Holte, and S. Matwin, *Machine learning for the detection of oil spills in satellite radar images*, Machine Learning **30**, 195 (1998).

[87] M. Saerens, P. Latinne, and C. Decaestecker, *Adjusting the outputs of a classifier to new a priori probabilities: a simple procedure*, Neural Computation **14**, 21 (2002).

[88] C. Elkan, *The foundations of cost-sensitive learning*, in *International Joint Conference on Artificial Intelligence*, Vol. 17 (2001) pp. 973–978.

[89] H. He and E. A. Garcia, *Learning from imbalanced data*, IEEE Transactions on Knowledge and Data Engineering **21**, 1263 (2009).

[90] W. Scaf-Klomp, *Screening for breast cancer*, Attendance and psychological consequences. Rijksuniversiteit Groningen, Groningen **142** (1997).

[91] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, *Smote: synthetic minority over-sampling technique*, Journal of Artificial Intelligence Research **16**, 321 (2002).

[92] A. Estabrooks, T. Jo, and N. Japkowicz, *A multiple resampling method for learning from imbalanced data sets*, Computational Intelligence **20**, 18 (2004).

[93] L.-F. Lee, *Some approaches to the correction of selectivity bias*, Review of Economic Studies **49**, 355 (1982).

[94] W. G. Cochran and D. B. Rubin, *Controlling bias in observational studies: a review*, Sankhyā: Indian Journal of Statistics, Series A , 417 (1973).

[95] D. Rubin, *Inference and missing data*, Biometrika **63**, 581 (1976).

[96] R. J. Little and D. B. Rubin, *Statistical analysis with missing data* (John Wiley & Sons, 2014).

[97] S. Pan, I. Tsang, J. Kwok, and Q. Yang, *Domain adaptation via transfer component analysis*, Neural Networks **22**, 199 (2011).

[98] B. Fernando, A. Habrard, M. Sebban, and T. Tuytelaars, *Unsupervised visual domain adaptation using subspace alignment*, in *International Conference on Computer Vision* (2013) pp. 2960–2967.

[99] R. Alaiz-Rodríguez and N. Japkowicz, *Assessing the impact of changing environments on classifier performance*, in *Conference of the Canadian Society for Computational Studies of Intelligence* (Springer, 2008) pp. 13–24.

[100] T. Lane and C. E. Brodley, *Approaches to online learning and concept drift for user identification in computer security*, in *Knowledge Discovery and Data Mining* (1998) pp. 259–263.

[101] G. Widmer and M. Kubat, *Learning in the presence of concept drift and hidden contexts*, Machine Learning **23**, 69 (1996).

[102] T. G. Dietterich, *Machine learning for sequential data: a review*, in *Joint IAPR International Workshops on Statistical Techniques in Pattern Recognition (SPR) and Structural and Syntactic Pattern Recognition (SSPR)* (Springer, 2002) pp. 15–30.

[103] J. Gama and G. Castillo, *Learning with local drift detection*, in *International Conference on Advanced Data Mining and Applications* (Springer, 2006) pp. 42–55.

[104] G. Ditzler and R. Polikar, *Hellinger distance based drift detection for nonstationary environments*, in *IEEE Symposium on Computational Intelligence in Dynamic and Uncertain Environments* (2011) pp. 41–48.

[105] Y. Chen, S. H. Hassani, A. Karbasi, and A. Krause, *Sequential information maximization: When is greedy near-optimal?* in *Conference on Learning Theory* (2015) pp. 338–363.

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

[107] S. Ben-David and R. Urner, *On the hardness of domain adaptation and the utility of unlabeled target samples*, in *Algorithmic Learning Theory* (Springer, 2012) pp. 139–153.

[108] J. Blitzer, S. Kakade, and D. Foster, *Domain adaptation with coupled subspaces*, in *International Conference on Artificial Intelligence and Statistics* (2011) pp. 173–181.

[109] J. Wen, C.-N. Yu, and R. Greiner, *Robust learning under uncertain test distributions: relating covariate shift to model misspecification*, in *International Conference on Machine Learning* (2014) pp. 631–639.

[110] A. Liu and B. Ziebart, *Robust classification under sample selection bias*, in *Advances in Neural Information Processing Systems* (2014) pp. 37–45.

[111] R. Gopalan, R. Li, and R. Chellappa, *Domain adaptation for object recognition: an unsupervised approach*, in *International Conference on Computer Vision* (IEEE, 2011) pp. 999–1006.

[112] M. Baktashmotlagh, M. Harandi, B. Lovell, and M. Salzmann, *Domain adaptation on the statistical manifold*, in *Conference on Computer Vision and Pattern Recognition* (2014) pp. 2481–2488.

[113] W. M. Kouw, L. J. Van Der Maaten, J. H. Krijthe, and M. Loog, *Feature-level domain adaptation*, Journal of Machine Learning Research **17**, 1 (2016).

[114] G. W. Imbens and D. B. Rubin, *Causal inference in statistics, social, and biomedical sciences* (Cambridge University Press, 2015).

[115] M. Dredze and K. Crammer, *Online methods for multi-domain learning and adaptation*, in *Conference on Empirical Methods in Natural Language Processing* (Association for Computational Linguistics, 2008) pp. 689–697.

[116] Y. Mansour, M. Mohri, and A. Rostamizadeh, *Domain adaptation with multiple sources*, in *Advances in Neural Information Processing Systems* (2009) pp. 1041–1048.

[117] V. Jain and E. Learned-Miller, *Online domain adaptation of a pre-trained cascade of classifiers*, in *Conference on Computer Vision and Pattern Recognition* (IEEE, 2011) pp. 577–584.

[118] D. Cohn, Z. Ghahramani, and M. Jordan, *Active learning with statistical models*, Journal of Artificial Intelligence Research (1996).

[119] A. Beygelzimer, S. Dasgupta, and J. Langford, *Importance weighted active learning*, in *International Conference on Machine Learning* (2009) pp. 49–56.

[120] P. J. Bickel, V. N. Nair, and P. C. Wang, *Nonparametric inference under biased sampling from a finite population*, Annals of Statistics , 853 (1992).

[121] D. G. Horvitz and D. J. Thompson, *A generalization of sampling without replacement from a finite universe*, Journal of the American Statistical Association **47**, 663 (1952).

[122] B. M. Marlin, R. S. Zemel, S. T. Roweis, and M. Slaney, *Collaborative filtering and the missing at random assumption*, in *Conference on Uncertainty in Artificial Intelligence* (2007).

[123] B. M. Marlin and R. S. Zemel, *Collaborative prediction and ranking with non-random missing data*, in *ACM Conference on Recommender Systems* (ACM, 2009) pp. 5–12.

[124] P. R. Rosenbaum and D. B. Rubin, *The central role of the propensity score in observational studies for causal effects*, Biometrika **70**, 41 (1983).

[125] P. R. Rosenbaum and D. B. Rubin, *Reducing bias in observational studies using subclassification on the propensity score*, Journal of the American Statistical Association **79**, 516 (1984).

[126] J. D. Angrist, G. W. Imbens, and D. B. Rubin, *Identification of causal effects using instrumental variables*, Journal of the American Statistical Association **91**, 444 (1996).

[127] J. Pearl, *Causality* (Cambridge University Press, 2009).

[128] N. Kallus, *A framework for optimal matching for causal inference*, in *Conference on Artificial Intelligence and Statistics* (2017) pp. 372–381.

[129] M. A. Hernán and J. M. Robins, *Estimating causal effects from epidemiological data*, Journal of Epidemiology & Community Health **60**, 578 (2006).

[130] L. Bottou, J. Peters, J. Quiñonero-Candela, D. X. Charles, D. M. Chickering, E. Portugaly, D. Ray, P. Simard, and E. Snelson, *Counterfactual reasoning and learning systems: the example of computational advertising*, Journal of Machine Learning Research **14**, 3207 (2013).

[131] A. Swaminathan and T. Joachims, *Counterfactual risk minimization: learning from logged bandit feedback*, in *International Conference on Machine Learning* (2015) pp. 814–823.

[132] T. Schnabel, A. Swaminathan, A. Singh, N. Chandak, and T. Joachims, *Recommendations as treatments: debiasing learning and evaluation*, in *International Conference on Machine Learning* (2016) pp. 1670–1679.

[133] H. Shimodaira, *Improving predictive inference under covariate shift by weighting the log-likelihood function*, Journal of Statistical Planning and Inference **90**, 227 (2000).

[134] C. Cortes and M. Mohri, *Domain adaptation and sample bias correction theory and algorithm for regression*, Theoretical Computer Science **519**, 103 (2014).

[135] S. Bickel, M. Brückner, and T. Scheffer, *Discriminative learning under covariate shift*, Journal of Machine Learning Research **10**, 2137 (2009).

[136] Y.-L. Yu and C. Szepesvári, *Analysis of kernel mean matching under covariate shift*, in *International Conference on Machine Learning* (2012) pp. 1147–1154.

[137] M. Sugiyama, T. Suzuki, and T. Kanamori, *Density ratio estimation in machine learning*, (2012).

[138] Y. Tsuboi, H. Kashima, S. Hido, S. Bickel, and M. Sugiyama, *Direct density ratio estimation for large-scale covariate shift adaptation*, Journal of Information Processing **17**, 138 (2009).

[139] S. Kullback and R. A. Leibler, *On information and sufficiency*, Annals of Mathematical Statistics **22**, 79 (1951).

[140] T. M. Cover and J. A. Thomas, *Elements of information theory* (John Wiley & Sons, 2012).

[141] D. J. MacKay, *Information theory, inference and learning algorithms* (Cambridge University Press, 2003).

[142] M. Sugiyama and K.-R. Müller, *Model selection under covariate shift*, in *International Conference on Artificial Neural Networks* (Springer, 2005) pp. 235–240.

[143] M. Sugiyama, M. Krauledat, and K.-R. Müller, *Covariate shift adaptation by importance weighted cross validation*, Journal of Machine Learning Research **8**, 985 (2007).

[144] M. Sugiyama, S. Nakajima, H. Kashima, P. V. Buenau, and M. Kawanabe, *Direct importance estimation with model selection and its application to covariate shift adaptation*, in *Advances in Neural Information Processing Systems* (2008) pp. 1433–1440.

[145] M. Sugiyama and K.-R. Müller, *Input-dependent estimation of generalization error under covariate shift*, Statistics & Decisions **23**, 249 (2005).

[146] T. Kanamori, S. Hido, and M. Sugiyama, *A least-squares approach to direct importance estimation*, Journal of Machine Learning Research **10**, 1391 (2009).

[147] T. Kanamori, S. Hido, and M. Sugiyama, *Efficient direct density ratio estimation for non-stationarity adaptation and outlier detection*, in *Advances in Neural Information Processing Systems* (2009) pp. 809–816.

[148] R. Fortet and E. Mourier, *Convergence de la répartition empirique vers la répartition théorique*, in *Annales Scientifiques de l'École Normale Supérieure*, Vol. 70 (1953) pp. 267–285.

[149] K. M. Borgwardt, A. Gretton, M. J. Rasch, H.-P. Kriegel, B. Schölkopf, and A. J. Smola, *Integrating structured biological data by kernel maximum mean discrepancy*, Bioinformatics **22**, e49 (2006).

[150] A. Gretton, K. M. Borgwardt, M. Rasch, B. Schölkopf, and A. J. Smola, *A kernel method for the two-sample-problem*, in *Advances in Neural Information Processing Systems* (2007) pp. 513–520.

[151] N. Aronszajn, *Theory of reproducing kernels*, Transactions of the American Mathematical Society **68**, 337 (1950).

[152] B. Schölkopf and A. J. Smola, *Learning with kernels: support vector machines, regularization, optimization, and beyond* (MIT Press, 2002).

[153] J. Shawe-Taylor and N. Cristianini, *Kernel methods for pattern analysis* (Cambridge University Press, 2004).

[154] A. Berlinet and C. Thomas-Agnan, *Reproducing kernel Hilbert spaces in probability and statistics* (Springer, 2011).

[155] A. Gretton, A. Smola, J. Huang, M. Schmittfull, K. Borgwardt, and B. Schölkopf, *Covariate shift by kernel mean matching*, in *Dataset Shift in Machine Learning*, edited by Q. Candela, M. Sugiyama, A. Schwaighofer, and N. Lawrence (MIT Press, 2009) pp. 131–160.

[156] J. Mercer, *Functions of positive and negative type, and their connection with the theory of integral equations*, Philosophical Transactions of the Royal Society of London. Series A **209**, 415 (1909).

[157] M. Aizerman, *Theoretical foundations of the potential function method in pattern recognition learning*, Automation and Remote Control **25**, 821 (1964).

[158] B. Schölkopf, R. Herbrich, and A. Smola, *A generalized representer theorem*, in *Computational Learning Theory* (Springer, 2001) pp. 416–426.

[159] F. Cucker and S. Smale, *On the mathematical foundations of learning*, Bulletin of the American Mathematical Society **39**, 1 (2002).

[160] J. Huang, A. Gretton, K. M. Borgwardt, B. Schölkopf, and A. J. Smola, *Correcting sample selection bias by unlabeled data*, in *Advances in Neural Information Processing Systems* (2007) pp. 601–608.

[161] S. Bickel, M. Brückner, and T. Scheffer, *Discriminative learning for differing training and test distributions*, in *International Conference on Machine Learning* (2007) pp. 81–88.

[162] C. F. Manski and S. R. Lerman, *The estimation of choice probabilities from choice based samples*, Econometrica: Journal of the Econometric Society , 1977 (1977).

[163] A. Okabe, *Spatial tessellations* (Wiley Online Library, 1992).

[164] E. G. Miller, *A new class of entropy estimators for multi-dimensional densities*, in *International Conference on Acoustics, Speech, and Signal Processing*, Vol. 3 (IEEE, 2003) pp. III–297.

[165] E. Learned-Miller, *Hyperspacings and the estimation of information theoretic quantities*, University of Massachusetts-Amherst Technical Report: Amherst, MA, USA (2004).

[166] M. Loog, *Nearest neighbor-based importance weighting*, in *International Workshop on Machine Learning for Signal Processing* (IEEE, 2012) pp. 1–6.

[167] D. A. Field, *Laplacian smoothing and delaunay triangulations*, International Journal for Numerical Methods in Biomedical Engineering **4**, 709 (1988).

[168] R. Aljundi, R. Emonet, D. Muselet, and M. Sebban, *Landmarks-based kernelized subspace alignment for unsupervised domain adaptation*, in *Conference on Computer Vision and Pattern Recognition* (2015) pp. 56–63.

[169] B. Sun and K. Saenko, *Subspace distribution alignment for unsupervised domain adaptation*. in *British Machine Vision Conference* (2015) pp. 24–1.

[170] S. J. Pan, X. Ni, J.-T. Sun, Q. Yang, and Z. Chen, *Cross-domain sentiment classification via spectral feature alignment*, in *International Conference on World wide web* (ACM, 2010) pp. 751–760.

[171] S. J. Pan, J. T. Kwok, and Q. Yang, *Transfer learning via dimensionality reduction*. in *Association for the Advancement of Artificial Intelligence*, Vol. 8 (2008) pp. 677–682.

[172] B. Schölkopf, A. Smola, and K.-R. Müller, *Kernel principal component analysis*, in *International Conference on Artificial Neural Networks* (Springer, 1997) pp. 583–588.

[173] B. Schölkopf, A. Smola, and K.-R. Müller, *Nonlinear component analysis as a kernel eigenvalue problem*, Neural Computation **10**, 1299 (1998).

[174] S. Si, D. Tao, and B. Geng, *Bregman divergence-based regularization for transfer subspace learning*, IEEE Transactions on Knowledge and Data Engineering **22**, 929 (2010).

[175] M. Shao, D. Kit, and Y. Fu, *Generalized transfer subspace learning through low-rank constraint*, International Journal of Computer Vision **109**, 74 (2014).

[176] G. E. Hinton and R. S. Zemel, *Autoencoders, minimum description length and helmholtz free energy*, in *Advances in Neural Information Processing Systems* (1994) pp. 3–10.

[177] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P.-A. Manzagol, *Stacked denoising autoencoders: learning useful representations in a deep network with a local denoising criterion*, Journal of Machine Learning Research **11**, 3371 (2010).

[178] X. Glorot, A. Bordes, and Y. Bengio, *Domain adaptation for large-scale sentiment classification: a deep learning approach*, in *International Conference on Machine Learning* (2011) pp. 513–520.

[179] M. Chen, Z. Xu, F. Sha, and K. Weinberger, *Marginalized denoising autoencoders for domain adaptation*, in *International Conference on Machine Learning* (2012) pp. 767–774.

[180] K. Saenko, B. Kulis, M. Fritz, and T. Darrell, *Adapting visual category models to new domains*, European Conference on Computer Vision , 213 (2010).

[181] B. Kulis, K. Saenko, and T. Darrell, *What you saw is not what you get: domain adaptation using asymmetric kernel transforms*, in *Conference on Computer Vision and Pattern Recognition* (IEEE, 2011) pp. 1785–1792.

[182] B. Geng, D. Tao, and C. Xu, *Daml: domain adaptation metric learning*, IEEE Transactions on Image Processing **20**, 2980 (2011).

[183] J. V. Davis, B. Kulis, P. Jain, S. Sra, and I. S. Dhillon, *Information-theoretic metric learning*, in *International Conference on Machine Learning* (2007) pp. 209–216.

[184] A. Bar-Hillel, T. Hertz, N. Shental, and D. Weinshall, *Learning a Mahalanobis metric from equivalence constraints*, Journal of Machine Learning Research **6**, 937 (2005).

[185] J. Hoffman, E. Rodner, J. Donahue, B. Kulis, and K. Saenko, *Asymmetric and category invariant feature transformations for domain adaptation*, International Journal of Computer Vision **109**, 28 (2014).

[186] R. A. Fisher, *The statistical utilization of multiple measurements*, Annals of Human Genetics **8**, 376 (1938).

[187] G. McLachlan, *Discriminant analysis and statistical pattern recognition*, Vol. 544 (John Wiley & Sons, 2004).

[188] A. M. Martínez and A. C. Kak, *PCA versus LDA*, IEEE Transactions on Pattern Analysis and Machine Intelligence **23**, 228 (2001).

[189] C. V. Dinh, R. P. Duin, I. Piqueras-Salazar, and M. Loog, *Fidos: A generalized fisher based feature extraction method for domain shift*, Pattern Recognition **46**, 2510 (2013).

[190] J. Hoffman, E. Rodner, J. Donahue, T. Darrell, and K. Saenko, *Efficient learning of domain-invariant image representations*, International Conference on Learning Representations (2013).

[191] B. Gong, K. Grauman, and F. Sha, *Reshaping visual datasets for domain adaptation*, in *Advances in Neural Information Processing Systems* (2013) pp. 1286–1294.

[192] J. Hoffman, T. Darrell, and K. Saenko, *Continuous manifold based adaptation for evolving visual domains*, in *Conference on Computer Vision and Pattern Recognition* (2014) pp. 867–874.

[193] K. Zhang, M. Gong, and B. Schölkopf, *Multi-source domain adaptation: a causal view.* in *Association for the Advancement of Artificial Intelligence* (2015) pp. 3150–3157.

[194] M. Baktashmotlagh, M. Harandi, B. Lovell, and M. Salzmann, *Unsupervised domain adaptation by domain invariant projection*, in *International Conference on Computer Vision* (2013) pp. 769–776.

[195] R. Caseiro, J. F. Henriques, P. Martins, and J. Batista, *Beyond the shortest path: unsupervised domain adaptation by sampling subspaces along the spline flow*, in *Conference on Computer Vision and Pattern Recognition* (2015) pp. 3846–3854.

[196] B. Gong, Y. Shi, F. Sha, and K. Grauman, *Geodesic flow kernel for unsupervised domain adaptation*, in *Conference on Computer Vision and Pattern Recognition* (IEEE, 2012) pp. 2066–2073.

[197] B. Gong, K. Grauman, and F. Sha, *Connecting the dots with landmarks: discriminatively learning domain-invariant features for unsupervised domain adaptation*, in *International Conference on Machine Learning* (2013) pp. 222–230.

[198] R. Gopalan, R. Li, and R. Chellappa, *Unsupervised adaptation across domain shifts by generating intermediate data representations*, IEEE Transactions on Pattern Analysis and Machine Intelligence **36**, 2288 (2014).

[199] J. C. Schlimmer and R. H. Granger, *Incremental learning from noisy data*, Machine Learning **1**, 317 (1986).

[200] Y.-C. Wong, *Differential geometry of grassmann manifolds*, Proceedings of the National Academy of Sciences **57**, 589 (1967).

[201] P.-A. Absil, R. Mahony, and R. Sepulchre, *Riemannian geometry of grassmann manifolds with a view on algorithmic computation*, Acta Applicandae Mathematicae **80**, 199 (2004).

[202] P. Turaga, A. Veeraraghavan, and R. Chellappa, *Statistical analysis on stiefel and grassmann manifolds with applications in computer vision*, in *Conference on Computer Vision and Pattern Recognition* (IEEE, 2008) pp. 1–8.

[203] J. Zheng, M.-Y. Liu, R. Chellappa, and P. J. Phillips, *A grassmann manifold-based domain adaptation approach*, in *International Conference on Pattern Recognition* (IEEE, 2012) pp. 2095–2099.

[204] K. A. Gallivan, A. Srivastava, X. Liu, and P. Van Dooren, *Efficient algorithms for inferences on grassmann manifolds*, in *IEEE Workshop on Statistical Signal Processing* (2003) pp. 315–318.

[205] G. J. McLachlan, *Iterative reclassification procedure for constructing an asymptotically optimal rule of allocation in discriminant analysis*, Journal of the American Statistical Association **70**, 365 (1975).

[206] E. Hellinger, *Neue begründung der theorie quadratischer formen von unendlichvielen veränderlichen.* Journal für die Reine und Angewandte Mathematik **136**, 210 (1909).

[207] Y. Mansour and M. Schain, *Robust domain adaptation*, Annals of Mathematics and Artificial Intelligence **71**, 365 (2014).

[208] M. Baktashmotlagh, M. Harandi, and M. Salzmann, *Distribution-matching embedding for visual domain adaptation*, Journal of Machine Learning Research **17**, 3760 (2016).

[209] K. Muandet, D. Balduzzi, and B. Schölkopf, *Domain generalization via invariant feature representation*, in *International Conference on Machine Learning* (2013) pp. 10–18.

[210] A. Smola, A. Gretton, L. Song, and B. Schölkopf, *A hilbert space embedding for distributions*, in *Algorithmic Learning Theory* (2007) pp. 13–31.

[211] B. Scholkopf and A. J. Smola, *Learning with kernels: support vector machines, regularization, optimization, and beyond* (MIT Press, 2001).

[212] Q. Gu and J. Zhou, *Learning the shared subspace for multi-task clustering and transductive transfer classification*, in *International Conference on Data Mining* (IEEE, 2009) pp. 159–168.

[213] M. Kim and V. Pavlovic, *Central subspace dimensionality reduction using covariance operators*, IEEE Transactions on Pattern Analysis and Machine Intelligence **33**, 657 (2011).

[214] M. Long, J. Wang, J. Sun, and S. Y. Philip, *Domain invariant transfer kernel learning*, IEEE Transactions on Knowledge and Data Engineering **27**, 1519 (2015).

[215] C. M. Bishop, *Neural networks for pattern recognition* (Oxford University Press, 1995).

[216] R. Rojas, *Neural networks: a systematic introduction* (Springer, 2013).

[217] G. Cybenko, *Approximation by superposition of sigmoidal functions*, Mathematics of Control, Signals and Systems **2**, 303 (1989).

[218] K. Hornik, M. Stinchcombe, and H. White, *Multilayer feedforward networks are universal approximators*, Neural Networks **2**, 359 (1989).

[219] K. Hornik, *Approximation capabilities of multilayer feedforward networks*, Neural Networks **4**, 251 (1991).

[220] Y. Ganin and V. Lempitsky, *Unsupervised domain adaptation by backpropagation*, in *International Conference on Machine Learning* (2015) pp. 1180–1189.

[221] Y. Ganin, E. Ustinova, H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, M. Marchand, and V. Lempitsky, *Domain-adversarial training of neural networks*, Journal of Machine Learning Research **17**, 1 (2016).

[222] E. Tzeng, J. Hoffman, T. Darrell, and K. Saenko, *Simultaneous deep transfer across domains and tasks*, in *International Conference on Computer Vision* (2015) pp. 4068–4076.

[223] K. Kamnitsas, C. Baumgartner, C. Ledig, V. Newcombe, J. Simpson, A. Kane, D. Menon, A. Nori, A. Criminisi, D. Rueckert, *et al.*, *Unsupervised domain adaptation in brain lesion segmentation with adversarial networks*, in *International Conference on Information Processing in Medical Imaging* (Springer, 2017) pp. 597–609.

[224] J. Hoffman, B. Kulis, T. Darrell, and K. Saenko, *Discovering latent domains for multisource domain adaptation*, in *European Conference on Computer Vision* (Springer, 2012) pp. 702–715.

[225] J. Blitzer, *Domain adaptation of natural language processing systems*, Ph.D. thesis, University of Pennsylvania (2008).

[226] J. Jiang, *Domain adaptation in natural language processing* (University of Illinois at Urbana-Champaign, 2008).

[227] G. Sidorov, F. Velasquez, E. Stamatatos, A. Gelbukh, and L. Chanona-Hernández, *Syntactic n-grams as machine learning features for natural language processing*, Expert Systems with Applications **41**, 853 (2014).

[228] M. Chen, K. Q. Weinberger, and J. Blitzer, *Co-training for domain adaptation*, in *Advances in Neural Information Processing Systems* (2011) pp. 2456–2464.

[229] W. Li, L. Duan, D. Xu, and I. Tsang, *Learning with augmented features for supervised and semi-supervised heterogeneous domain adaptation*, IEEE Transactions on Pattern Analysis and Machine Intelligence **36**, 1134 (2014).

[230] F. Provost and T. Fawcett, *Robust classification for imprecise environments*, Machine Learning **42**, 203 (2001).

[231] W. M. Kouw and M. Loog, *Target contrastive pessimistic risk for robust domain adaptation*, ArXiv (2017).

[232] B. Zadrozny, *Learning and evaluating classifiers under sample selection bias*, in *International Conference on Machine Learning* (ACM, 2004) p. 114.

[233] P. D. Grünwald and A. P. Dawid, *Game theory, maximum entropy, minimum discrepancy and robust Bayesian decision theory*, Annals of Statistics , 1367 (2004).

[234] H. Xu and S. Mannor, *Robustness and generalization*, Machine Learning **86**, 391 (2012).

2

Cross-validation under covariate shift

This chapter identifies a problem with the usual procedure for L^2-regularization parameter estimation in a domain adaptation setting. In such a setting, there are differences between the distributions generating the training data (source domain) and the test data (target domain). The usual cross-validation procedure requires validation data, which can not be obtained from the unlabeled target data. The problem is that if one decides to use source validation data, the regularization parameter is underestimated. One possible solution is to scale the source validation data through importance weighting, but we show that this correction is not sufficient. The chapter is concluded with an empirical analysis of the effect of several importance weight estimators on the estimation of the regularization parameter.

This chapter is based on the paper "On regularization parameter estimation under covariate shift".

2.1. Introduction

In supervised learning, there is a (mostly implicit) assumption that the training data is an unbiased sampling of the underlying distribution of interest. However, that may not be the case. In a variety of problems there is often an unknown bias in the sampling procedure. These arise due to environmental effects, such as temperature in different genome sequencing centers [1–3], or due to the use of particular measuring instruments, such as types of cameras in computer vision [4, 5]. This means the training dataset (source domain) and the test dataset (target domain) are technically generated by different distributions and generalization might no longer be possible. The challenge lies in using the labeled source data and the unlabeled target data to classify new target data; a problem setting often referred to as domain adaptation, transfer learning or sample selection bias [6–11]. Most research focuses on classifiers that incorporate information on the difference between the data in both domains, but unfortunately most of these approaches overlook the role of the regularization parameter.

Regularization is used to combat overfitting of complex models and is a vital component in most classifiers to ensure they generalize to unseen data. It consists of a trade-off between how well the classifier can discriminate training samples and how complex it must become to do so. This balance is described by the regularization parameter which is usually estimated by holding out a small subset of unseen labeled data and evaluating the trained classifier (cross-validation). However, since there are no labeled target samples available, it is not possible to construct a target validation set. If one were to alternatively construct a validation set from source data, the estimator converges in distribution to the source risk and not the target risk [12].

We study how the generalization performance of a classifier behaves as a function of the regularization parameter and the domain dissimilarity. There are many factors that influence the value of the optimal regularization parameter, such as the moments of the class-conditional distributions in each domain (differences in variance, skewness, etc.), concept drift (different class priors in each domain), types of adapting classifiers (some require less regularization than others) and high-dimensional distribution estimation errors, but we focus on differences in variance between domains. The first correction that comes to mind consists of scaling the source validation risk with importance weights and although this remedies the problem somewhat, we show that the optimal regularization parameter for the target domain remains underestimated.

2.2. Estimation problem

Domains are different biased samplings, which correspond to different joint probability distributions over the same input space $\mathcal{X}$ and output space $\mathcal{Y} = \{-1, +1\}$. We will refer to the source domain with $\mathcal{S}$ and the target domain with $\mathcal{T}$. Source data X with labels y consists of n samples from $p_{\mathcal{S}}(x, y)$, denoted as a data set $\{(x_i, y_i)\}_{i=1}^{n}$, and target data Z with labels u consists of m samples from $p_{\mathcal{T}}(x, y)$, denoted as a data set $\{(z_j, u_j)\}_{j=1}^{m}$. The input space is a D-dimensional feature space, which means that x_i and z_j are vectors: $x_i = (x_{i1}, \ldots, x_{iD})$ and $z_j = (z_{j1}, \ldots, z_{jD})$. A classifier is a function that takes as input data

and outputs a class prediction, $h : \mathcal{X} \to \mathcal{Y}$.

2.2.1. Regularized risk

The risk minimization framework allows one to construct classifiers through searching a class of hypothetical functions $\mathcal{H}$ (e.g., linear) and selecting the one that minimizes the expected loss $\ell : \mathbb{R} \times \mathcal{Y} \to \mathbb{R}^+$. The source and target risk are defined respectively as:

$$R_S(h) = \int_{\mathcal{X}} \sum_{y \in \mathcal{Y}} \ell(h(x), y)\, p_S(x, y)\, \mathrm{d}x \tag{2.1}$$

$$R_T(h) = \int_{\mathcal{X}} \sum_{y \in \mathcal{Y}} \ell(h(x), y)\, p_T(x, y)\, \mathrm{d}x\,. \tag{2.2}$$

Note that for any h, the source and target risks differ only through the joint probabilities $p_S(x, y)$ and $p_T(x, y)$. The goal is to find the classification function h that will minimize the target risk, based on source data.

Unfortunately, minimizing the empirical source risk with respect to h directly, often leads to a solution that does not generalize well to other samples (overfitting), let alone samples from another distribution. In order to prevent the classifier from becoming too specific for the training data set, a complexity term is added to the empirical risk during training. Most often, the L^2-norm of the classifier's parameters θ is chosen as the complexity term. The regularized empirical risk can be written as:

$$\hat{R}_T(\theta_\lambda) = \frac{1}{|T|} \sum_{t \in T} \ell(h(x_t \mid \theta), y_t) + \lambda \|\theta\|_2^2 \tag{2.3}$$

where the subscript T denotes the set of indices indicating which source samples are used for *training* $T \subset \{1, \ldots, n\}$ (not the *target* samples), $|\cdot|$ denotes the cardinality and $\|.\|_2$ denotes the L^2-norm. Note that the empirical risk is now a function of the classifier's parameters θ, instead of h, and that it has received the subscript T to indicate that it is the empirical risk with respect to the source training samples. In the following, we will use other subscripts to indicate empirical risks with respect to other data sets.

The regularization parameter λ trades off the empirical risk and the L^2-norm. It is usually estimated by defining a set of values Λ, training a classifier for each and selecting $\lambda \in \Lambda$ with the minimal risk according to an evaluation on a disjoint validation dataset. The set of regularized classifiers can be denoted as:

$$\theta_\Lambda = \{\theta_\lambda = \arg\min_{\theta \in \Theta} \hat{R}_T(\theta_\lambda) \mid \lambda \in \Lambda\}\,. \tag{2.4}$$

where θ_λ refers to the classifier that is trained using λ. Θ is the classifier parameter space, which for linear classifiers is, for instance, the set of $D + 1$-dimensional real vectors $\mathbb{R}^{D+1}$. The regularization parameter space Λ is often taken to be an exponentially increasing set of nonnegative values; for example $\{0, 0.01, 0.1, 1, 10, 100, 1000\}$.

If we choose a quadratic loss function, $\ell(h(x \mid \theta), y) = (h(x \mid \theta) - y)^2$, with a linear hypothesis class, then the solution to minimizing equation 2.3 with respect to the classifier parameters is $\theta_\lambda = (X_T^\top X_T + \lambda I)^{-1}(X_T^\top y_T)$, where X refers to the $n \times D$ data matrix. Note that for the same training data X_T, θ_λ varies due to different choices of λ.

2

2.2.2. Evaluation measure

Evaluating a classifier consists of computing its empirical risk on a novel dataset. We will be studying two validation sets, the first being held-out source data, and the second being target data. We will incorporate the quadratic loss in the risk function for validation as well. The held-out source validation data will be marked with the subscript V, which indicates the set of indices that are disjoint from the training set $V \cap T = \emptyset$). Plugging in source validation $\{(X_V, y_V)\}$ and target validation data $\{(Z, u)\}$, the empirical risks are:

$$\hat{R}_V(\theta_\lambda) = 1 - \frac{2}{|V|} y_V^\top X_V \theta_\lambda + \frac{1}{|V|} \theta_\lambda^\top X_V^\top X_V \theta_\lambda \tag{2.5}$$

$$\hat{R}_Z(\theta_\lambda) = 1 - \frac{2}{|Z|} u^\top Z \theta_\lambda + \frac{1}{|Z|} \theta_\lambda^\top Z^\top Z \theta_\lambda \,. \tag{2.6}$$

Cross-validation consists of holding out each source sample at least once, training a classifier on the remainder and evaluating on the held out validation set. One round of cross-validation is performed for each $\theta_\lambda \in \theta_\Lambda$ and the minimizer of the set with respect to the empirical risk corresponds to the estimated regularization parameter.

2.2.3. Problem setting

For any h, the empirical source validation risk $\hat{R}_V$ converges to the true source risk R_S by independently sampling validation data sets infinitely many times [13]. Unfortunately, this is not equal to the target risk R_T. Hence, selecting a regularization parameter based on source validation data will not be equivalent to selecting a regularization parameter based on target validation data. Furthermore, the larger the difference between $p_S(x, y)$ and $p_T(x, y)$, the larger the difference between the selected regularization parameters. In order to obtain a regularization parameter estimate that is closer to the one found by validating on the target risk, we need a way to match the validation empirical risks.

2.3. Covariate Shift

A natural approach to designing a corrected validation procedure, would be to employ some functional relation between the source and target risks. Fortunately, such a relation exists for a subset of the class of domain adaptation problems: if one makes the *covariate shift* assumption that the class posterior distributions are equivalent in both domains,

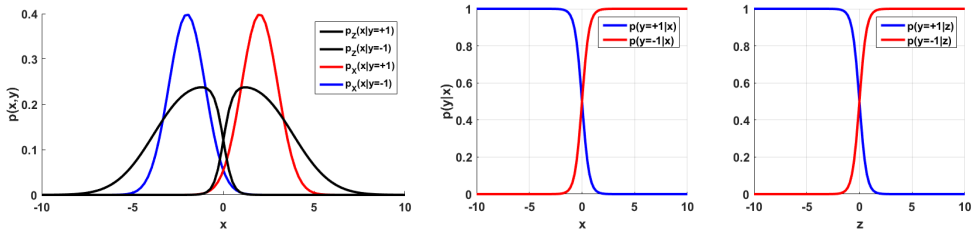
$p_{\mathcal{T}}(y \mid x) = p_{\mathcal{S}}(y \mid x)$, then the target risk can be rewritten into a weighted source risk:

$$
\begin{aligned}
R_{\mathcal{W}}(h) &= \int_{\mathcal{X}} \sum_{y \in \mathcal{Y}} \ell(h(x), y) \, \frac{p_{\mathcal{T}}(y, x)}{p_{\mathcal{S}}(y, x)} p_{\mathcal{S}}(x, y) \, \mathrm{d}x \\
&= \int_{\mathcal{X}} \sum_{y \in \mathcal{Y}} \ell(h(x), y) \, \frac{p_{\mathcal{T}}(y \mid x) \, p_{\mathcal{T}}(x)}{p_{\mathcal{S}}(y \mid x) \, p_{\mathcal{S}}(x)} p_{\mathcal{S}}(x, y) \, \mathrm{d}x \\
&= \int_{\mathcal{X}} \sum_{y \in \mathcal{Y}} \ell(h(x), y) \, \frac{p_{\mathcal{T}}(x)}{p_{\mathcal{S}}(x)} p_{\mathcal{S}}(x, y) \, \mathrm{d}x \, .
\end{aligned}
$$

The functional relation thus consists of weighting the source samples appropriately. It can be shown that under the additional assumption of a small domain discrepancy, this problem setting is learnable [14].

2.3.1. Generating a covariate shift setting

Since we are restricting the analysis to covariate shift settings, we need to generate such a problem. First, we choose a set of source class-conditional distributions $p_{\mathcal{S}}(x \mid y)$, a set of priors $p_{\mathcal{S}}(y)$ and compute the class posterior distributions $p_{\mathcal{S}}(y \mid x)$ through Bayes' rule. Then, by choosing a different target distribution $p_{\mathcal{T}}(x)$, multiplying by the derived class-posterior distributions $p_{\mathcal{T}}(y \mid x) = p_{\mathcal{S}}(y \mid x)$ and inverting Bayes' rule, the class-conditional target distributions $p_{\mathcal{T}}(x \mid y)$ are obtained. Note that this also implies that the priors are equal in both domains: $p_{\mathcal{S}}(y) = p_{\mathcal{T}}(y)$. Figure 2.1 (left) visualizes an example of this problem for Gaussian class-conditional distributions. We plotted the labeled source distributions in red and blue with the unlabeled target distributions in black. The class posteriors of this problem are plotted in Figure 2.1 (right), and are equivalent. An artificial dataset can be generated by sampling from these distributions, either through inverse transform sampling or rejection sampling.



(a) Class-conditional distributions of each domain.

(b) Class-posterior distributions of the source (left) and the target domain (right).

Figure 2.1: An artificially generated 1-dimensional covariate shift problem.

If we fix the source class-conditional distributions to be Gaussian distributions, with the blue class as $\mathcal{N}(x \mid -1, 1)$ and the red class as $\mathcal{N}(x \mid 1, 1)$, then we can generate 5 problem settings by choosing 5 different target distributions. Figure 2.2 (left) shows 5 Gaussian target distributions with equal means as the source distributions but with different variances

$\sigma_T^2 \in \{0.5, 1, 2, 3, 4\}$. If we train a classifier based on the source class-conditional distributions and evaluate it using the target empirical risk, then it becomes apparent that the difference between the minimizer of the source risk and the target risk starts to increase as the difference between the distributions start to increase. Figure 2.2 (right) plots the empirical risk as a function of θ_λ for the 5 covariate shift problems, with the minimum for each marked with a black square. Note that for $\sigma_T^2 = 1$ the distributions are equivalent and its minimizer is equivalent to the minimizer of the source risk. The curves show a gradual increase in the minimizers as the variance increases.

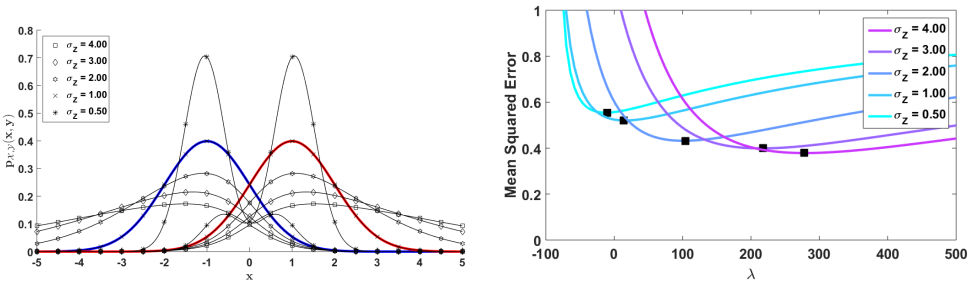


Figure 2.2: (Left) 5 covariate shift problems, with the target variance $\sigma_T^2 \in \{0.5, 1, 2, 3, 4\}$. (Right) The corresponding target empirical risk curves. The black squares denote the minima of these curves.

2.3.2. Difference in error curves

If we minimize the empirical risk curves of the source validation data (2.5) with respect to the trained regularized classifier θ_λ, we obtain:

$$\begin{aligned}\theta_{\hat{\lambda}_V} &= \arg\min_{\theta_\lambda} \hat{R}_V(\theta_\lambda) \\ &= (X_V^\top X_V)^{-1}(X_V^\top y_V)\,.\end{aligned}$$

The subscript $\hat{\lambda}_V$ is used to signify that this is the regularization parameter chosen by validating on the held-out source data; since the same training data X_T is used, θ_λ only differs through the choice of λ. Similarly, minimizing the empirical risk on the target validation data produces:

$$\begin{aligned}\theta_{\hat{\lambda}_Z} &= \arg\min_{\theta_\lambda} \hat{R}_Z(\theta_\lambda) \\ &= (Z^\top Z)^{-1}(Z^\top u)\end{aligned}$$

where $\hat{\lambda}_Z$ denotes the optimal regularization parameter we would have chosen, had we been able to validate on labeled target data.

Studying these two forms, we see that these estimates of λ differ mainly through the data inner products (i.e., the uncentered, unnormalized covariance matrices). To illustrate this point, we can decompose the data through a singular value decomposition, allowing

us to express the minimizers as:

$$\theta_{\hat{\lambda}_V} = (V_V D_V U_V^\top) y_V$$
$$\theta_{\hat{\lambda}_Z} = (V_Z D_Z U_Z^\top) u$$

where the diagonal matrices D_V and D_Z consist of the normalized singular values $D_{V,ii} = \alpha_{V,i}/\alpha_{V,i}^2$ and $D_{Z,ii} = \alpha_{Z,i}/\alpha_{Z,i}^2$. Apart from a change of basis from V_V to V_Z and U_V to U_Z, the difference lies mainly in the scale of the eigenvalues.

If we were to apply a scaling operation to the validation risk, then the difference between these curves can be minimized. Finding the optimal regularization parameter for the target domain will then be equivalent to finding the optimal regularization parameter for the scaled validation risk.

2.3.3. Importance-weighted validation

Sugiyama et al. (2007) employ just such a scaling transformation in the form of importance weighting the validation risk, with weights w as estimates of the ratio of data marginals $p_{\mathcal{T}}(x)/p_{\mathcal{S}}(x)$ [12]. These weights scale the risk of each individual validation sample separately. This leads to an importance weighted source validation risk as follows:

$$\hat{R}_W(\theta_\lambda) = 1 - \frac{2}{|V|} y_V^\top W X_V \theta_\lambda + \frac{1}{|V|} \theta_\lambda^\top X_V^\top W X_V \theta_\lambda$$

where W is a matrix with the importance weights for the validation samples on its diagonals. This formulation has the following minimizer:

$$\theta_{\hat{\lambda}_W} = (X_V^\top W X_V)^{-1} (X_V^\top W y_V)\,.$$

The ratio of probabilities can have a very large variance, depending on how likely it is to encounter either extremely large target probabilities or extremely small source probabilities. Furthermore, in the small sample size setting, estimation errors increase the possibility of a numerical explosion, such as when 2 samples are drawn that lie so close together that the estimated target distribution resembles a Dirac distribution. Lastly, the cross-validation estimator has its own variance [15] which is now directly affected by the variance of the importance weight estimator. For a better understanding of the behavior of an importance weighted cross-validation estimator, we performed a number of experiments with a large diversity of weight estimators in the following section.

2.4. Experiments

We conducted an experiment on an artificial problem setting and one on a typical real-world domain adaption problem where there is no knowledge on whether the covariate shift assumption holds. Our goal is to evaluate the ability of a number of both parametric and nonparametric importance weight estimators to correctly estimate the optimal regularization parameter in the target domain. These experiments illustrate that a large diversity of existing estimators tends to underestimate the optimal target parameter.

2.4.1. Importance weight estimators

We selected four importance weight estimators with a diverse set of behaviors.

Ratio of Gaussians

A baseline method of estimating the marginal data ratio through modeling each sample set with a separate Gaussian distribution [16]:

$$\hat{w}_{\text{rG}} = \frac{\mathcal{N}(x \mid \hat{\mu}_{\mathcal{T}}, \hat{\sigma}^2_{\mathcal{T}})}{\mathcal{N}(x \mid \hat{\mu}_{\mathcal{S}}, \hat{\sigma}^2_{\mathcal{S}})},$$

where $\mathcal{N}$ denotes the Gaussian distribution function, $\hat{\mu}_{\mathcal{S}}$ denotes the estimated mean of the source data set, $\hat{\mu}_{\mathcal{T}}$ the estimated mean of the target data set, $\hat{\sigma}^2_{\mathcal{S}}$ denotes the estimated variance of the source set and $\hat{\sigma}^2_{\mathcal{T}}$ the estimated variance of the target set. Note that the data marginals in our problem are actually Gaussian and that this is thus a correctly specified model.

Kullback-Leibler importance estimation procedure

This popular method is based on minimizing the Kullback-Leibler divergence between the re-weighted source samples and the target samples [17]:

$$\hat{w}_{\text{KLIEP}} = \arg\max_{w \in W} \sum_{j=1}^{m} \log \sum_{i}^{n} w_i \kappa(x_i, z_j),$$

$$\text{s.t.} \sum_{i}^{n} w_i \kappa(x_i, z_j) = n,$$

where κ is a kernel function, in this case between the source samples x and the target samples z. We chose a Gaussian kernel, with the kernel width estimated through a separate 3-fold cross-validation procedure [17].

Kernel Mean Matching

Another popular weight estimator that is motivated by assigning weights that minimize the Maximum Mean Discrepancy (MMD) between the re-weighted source and the target samples [18]. The MMD is the distance between the means of two sets of samples under a worst-case transformation (one that pushes them as far away as possible):

$$\hat{w}_{\text{KMM}} = \arg\min_{w \in W} \frac{1}{2} w^{\top} \kappa(x, x') w - \frac{n}{m} \sum_{j=1}^{m} \kappa(z_j, x) w,$$

$$\text{s.t.} \quad w_i \in [0,\, B]$$

$$\left| \frac{1}{n} \sum_{i=1}^{n} w_i - 1 \right| \leq \epsilon$$

where the constraints ensure that the weights are non-negative, bounded above and roughly average to 1. For the kernel function κ, we selected a radial basis function with Silverman's

rule of thumb for bandwidth selection. Huang et al. recommend setting epsilon to $B/\sqrt{n}$, which ensures that the allowed deviation from the sample size depends on both the upper bound for each weight and the sample size itself.

Nearest Neighbour

Lastly, we have a nonparametric estimator based on a Voronoi tessellation of the space [19]. The procedure consists of assigning a weight to each source sample based on the number of target samples that are nearest neighbors of it. It is proportional, up to the ratio of sample sizes, to the ratio of marginal distributions [20, 21]. It is expressed as:

$$\hat{w}_{\text{NN}} = |C_i \cap \{z_j\}_{j=1}^{m}| + 1\,,$$

where C_i refers to the Voronoi cell of sample x_i. The tessellation can be smoothed by adding a value of 1 to each cell, a technique also known as Laplace smoothing.

2.4.2. Artificial data

Our first experiment consists of an evaluation of different importance weight estimators and their resulting minimizers of θ_λ. The set θ_λ was constructed with a least-squares classifier $\theta_\lambda = (X_T^\top X_T + \lambda I)^{-1}(X_T^\top y_T)$. Λ was taken from -100 to 500 in 101 steps. For the source data, we drew 100 samples from two Gaussian class-conditional distribution with means $\mu_S \in \{-1, 1\}$ and unit variances $\sigma_S^2 = 1$. The target class-conditional distributions have the same mean $\mu_T \in \{-1, 1\}$, but with a different set of variances $\sigma_T^2 \in \{0.1, 0.5, 1, 2, 3, 4\}$. The ratio of the marginal distributions is sensitive in regions of low probability of the source distribution: really small probabilities in the denominator explode the weight value. Therefore, we expect the minimizers of the importance weight estimators to be close to the target minimizer for smaller target variances $\sigma_T^2 < \sigma_S^2$. Consequently, we expect erratic behavior for target variance larger than the source variance $\sigma_T^2 > \sigma_S^2$. Table 2.1 displays the estimated regularization parameters for the source validation risk, the importance-weight estimators, the actual ratio of marginals $p_T(x)/p_S(x)$, and for the empirical target risk. Shown are the means and standard errors over 100 repetitions.

Table 2.1: The mean and standard error of the estimated regularization parameter $\hat{\lambda}$ for different importance weight estimators and an increasingly larger target variance in a covariate shift problem.

σ_T^2	0.1	0.5	1.0	2.0	3.0	4.0
$h_{\hat{\lambda}_V}$	4 (19)	3 (20)	5 (20)	3 (20)	4 (20)	4 (19)
$\hat{w}_{\text{rG}}$	-15 (20)	-10 (19)	6 (17)	30 (24)	55 (41)	58 (36)
$\hat{w}_{\text{KLIEP}}$	-23 (33)	-2 (20)	3 (20)	0 (17)	4 (24)	17 (25)
$\hat{w}_{\text{KMM}}$	22 (24)	17 (26)	11 (25)	-1 (24)	-15 (21)	-14 (19)
$\hat{w}_{\text{NN}}$	-18 (28)	-13 (21)	4 (23)	33 (24)	53 (24)	64 (26)
p_T/p_S	-46 (47)	-33 (21)	3 (18)	46 (44)	72 (50)	77 (50)
$h_{\hat{\lambda}_Z}$	179 (137)	-24 (21)	5 (21)	102 (28)	207 (38)	296 (45)

It seems that all importance weight estimators as well as the true ratio of marginals underestimate the target risk minimizer. Furthermore, it seems that $\hat{w}_{\text{KMM}}$ leads to increasingly smaller minimizers for an increasing target variance. Even though $\hat{w}_{\text{KLIEP}}$ is increasing, it still underestimates the true value the most. $\hat{w}_{\text{rG}}$ is the most accurate one, but that will probably not be the case if the marginal distributions are not Gaussian anymore (i.e., model misspecification). $\hat{w}_{\text{NN}}$ is the other most accurate one and lies closest to true importance weights. Considering that it does not rely on an assumption of normality, it might be the preferred estimator in a more general setting.

2.4.3. Heart disease

The artificial data represents a case where we know exactly what the dissimilarity is between domains and whether assumptions are valid. However, it is also interesting to evaluate on data where we do not have this knowledge. For this we have selected a UCI dataset [22] on medical data where the domain dissimilarity is caused by a geographically biased sampling of patients. The goal is to classify the presence of a heart disease based on symptoms. The four domains correspond to hospitals in 'Cleveland', 'Virginia', 'Hungary' and 'Switzerland', containing 303, 200, 294 and 123 samples each respectively. There are a total of 14 symptoms, but 2 contained so much missing data ($> 99\%$) that these were removed from the set. All other missing data was imputed with 0 values after z-scoring, i.e. subtracting the mean of each feature and normalizing by its standard deviation. Table 2.2 displays the minimizers found by the importance weight estimators compared with those found by the unweighted source validation risk $h_{\hat{\lambda}_V}$ and the target validation risk $h_{\hat{\lambda}_Z}$, for all combinations of treating one hospital as the source domain and another as the target. Shown are the means and standard errors over 10 repetitions.

Table 2.2: Heart disease dataset. Mean and standard error of the estimated regularization parameter $\hat{\lambda}$ for different importance weight estimators. The letters are abbreviations of the 4 hospitals: C='Cleveland', V='Virginia', H='Hungary' and S='Switzerland.

$\mathcal{S}$	$\mathcal{T}$	$h_{\hat{\lambda}_V}$	$\hat{w}_{\text{rG}}$	$\hat{w}_{\text{KLIEP}}$	$\hat{w}_{\text{KMM}}$	$\hat{w}_{\text{NN}}$	$h_{\hat{\lambda}_Z}$
C	V	1 (5)	-1 (8)	1 (5)	9 (13)	2 (5)	500 (0)
C	H	1 (4)	4 (6)	1 (6)	2 (14)	4 (5)	500 (0)
C	S	4 (6)	7 (9)	0 (5)	9 (12)	-1 (9)	500 (0)
V	H	5 (5)	9 (13)	3 (6)	2 (5)	7 (9)	417 (66)
V	S	3 (4)	-1 (12)	3 (6)	2 (3)	7 (8)	-60 (284)
H	S	3 (6)	34 (48)	3 (8)	44 (40)	4 (6)	500 (0)
V	C	4 (5)	-1 (9)	2 (4)	2 (3)	4 (4)	500 (0)
H	C	1 (5)	0 (7)	2 (7)	31 (29)	0 (6)	500 (0)
S	C	2 (4)	-1 (4)	2 (4)	1 (3)	2 (4)	488 (30)
H	V	4 (4)	-15 (14)	4 (7)	25 (43)	4 (9)	500 (0)
S	V	2 (4)	-1 (4)	1 (7)	1 (3)	4 (4)	-95 (253)
S	H	0 (4)	4 (8)	2 (6)	0 (5)	4 (4)	289 (89)

The results show that also for real datasets all importance weight estimators underestimate the optimal target regularization parameter. Note that the standard errors are 0 for all $h_{\hat{\lambda}_Z}$ with value 500, because 500 is the right boundary of the set Λ. Extending the boundary would produce even larger values for the optimal target regularization parameter. It seems that $\hat{w}_{\mathrm{KMM}}$ is the best performing estimator here. $\hat{w}_{\mathrm{rG}}$ also produces reasonable results, but that would probably not be the case if we had not z-scored each feature first. That ensures an overlap of the regions with high probability mass in each domain. The other estimators seem to find weight values close to 1, as they are not very different from the unweighted source validation risk.

2.5. Discussion

Considering the significance of regularization to generalization, it would be interesting to further study factors that influence the difference between the risk minimizers in each domain. At the moment we assume that no concept drift has occurred (a difference between class priors in each domain), but if this assumption is violated then the difference in scale depends on the two dominant classes in each domain. The minimizers of the empirical risk would be dominated by the proportions of samples that belong to one class, which can get very complicated in the multi-class setting. Furthermore, it would be interesting to describe the minimizers in terms of general measures of domain dissimilarity, such as the discrepancy distance [23] or the $\mathcal{H}$-divergence [8].

The main difficulty in estimating the appropriate weights lies in the fact that it is hard to estimate exactly how the two domains differ from each other. Most adaptation approaches are sensitive to only a particular type of relation between domains or rely on assumptions that can not be checked in advance. Furthermore, estimation errors tend to propagate. For instance, if the distributions of each domain's data marginals are poorly estimated, then the importance weights explode, leading to a more erroneous estimate of the optimal target regularization parameter. In domain adaptation settings with so many sources of uncertainty, it seems that simple methods work best.

2.6. Conclusion

We have shown an empirical analysis of regularization parameter estimation in the context of differing variances in covariate shift problems. It seems that the generalization performance of an unadapted source classifier can be improved by importance weighting the source validation risk. However, most popular weight estimators underestimate the optimal target regularization parameter.

References

[1] N. H. Shah, C. Jonquet, A. P. Chiang, A. J. Butte, R. Chen, and M. A. Musen, *Ontology-driven indexing of public datasets for translational bioinformatics*, BMC Bioinformatics **10**, 1 (2009).

[2] S. Mei, W. Fei, and S. Zhou, *Gene ontology based transfer learning for protein subcellular localization*, BMC Bioinformatics **12**, 1 (2011).

[3] Q. Xu and Q. Yang, *A survey of transfer and multitask learning in bioinformatics*, Journal of Computing Science and Engineering **5**, 257 (2011).

[4] K. Saenko, B. Kulis, M. Fritz, and T. Darrell, *Adapting visual category models to new domains*, European Conference on Computer Vision , 213 (2010).

[5] J. Hoffman, E. Rodner, J. Donahue, T. Darrell, and K. Saenko, *Efficient learning of domain-invariant image representations*, International Conference on Learning Representations (2013).

[6] C. Cortes, M. Mohri, M. Riley, and A. Rostamizadeh, *Sample selection bias correction theory*, in *Algorithmic Learning Theory* (Springer, 2008) pp. 38–53.

[7] J. Quionero-Candela, M. Sugiyama, A. Schwaighofer, and N. D. Lawrence, *Dataset shift in machine learning* (MIT Press, 2009).

[8] S. Ben-David, J. Blitzer, K. Crammer, A. Kulesza, F. Pereira, and J. W. Vaughan, *A theory of learning from different domains*, Machine Learning **79**, 151 (2010).

[9] A. Margolis, *A literature review of domain adaptation with unlabeled data*, Rapport Technique, University of Washington , 35 (2011).

[10] J. G. Moreno-Torres, T. Raeder, R. Alaiz-RodríGuez, N. V. Chawla, and F. Herrera, *A unifying view on dataset shift in classification*, Pattern Recognition **45**, 521 (2012).

[11] W. M. Kouw, L. J. Van Der Maaten, J. H. Krijthe, and M. Loog, *Feature-level domain adaptation*, Journal of Machine Learning Research **17**, 1 (2016).

[12] M. Sugiyama, M. Krauledat, and K.-R. Müller, *Covariate shift adaptation by importance weighted cross validation*, Journal of Machine Learning Research **8**, 985 (2007).

[13] J. Friedman, T. Hastie, and R. Tibshirani, *The elements of statistical learning*, Vol. 1 (Springer, 2001).

[14] S. Ben-David, T. Lu, T. Luu, and D. Pál, *Impossibility theorems for domain adaptation*, in *International Conference on Artificial Intelligence and Statistics* (2010) pp. 129–136.

[15] M. Markatou, H. Tian, S. Biswas, and G. M. Hripcsak, *Analysis of variance of cross-validation estimators of the generalization error*, Journal of Machine Learning Research **6**, 1127 (2005).

[16] H. Shimodaira, *Improving predictive inference under covariate shift by weighting the log-likelihood function*, Journal of Statistical Planning and Inference **90**, 227 (2000).

[17] M. Sugiyama, S. Nakajima, H. Kashima, P. V. Buenau, and M. Kawanabe, *Direct importance estimation with model selection and its application to covariate shift adaptation*, in *Advances in Neural Information Processing Systems* (2008) pp. 1433–1440.

[18] J. Huang, A. Gretton, K. M. Borgwardt, B. Schölkopf, and A. J. Smola, *Correcting sample selection bias by unlabeled data*, in *Advances in Neural Information Processing Systems* (2007) pp. 601–608.

[19] M. Loog, *Nearest neighbor-based importance weighting*, in *International Workshop on Machine Learning for Signal Processing* (IEEE, 2012) pp. 1–6.

[20] E. G. Miller, *A new class of entropy estimators for multi-dimensional densities*, in *International Conference on Acoustics, Speech, and Signal Processing*, Vol. 3 (IEEE, 2003) pp. III–297.

[21] E. Learned-Miller, *Hyperspacings and the estimation of information theoretic quantities*, University of Massachusetts-Amherst Technical Report: Amherst, MA, USA (2004).

[22] M. Lichman, *UCI machine learning repository*, (2013).

[23] Y. Mansour, M. Mohri, and A. Rostamizadeh, *Domain adaptation: Learning bounds and algorithms*, in *Conference on Learning Theory* (2009).

3

Sampling variance of importance-weighted risks

Covariate shift classification problems can in principle be tackled by importance-weighting training samples. However, the sampling variance of the risk estimator is often scaled up dramatically by the weights. This means that during cross-validation - when the importance-weighted risk is repeatedly evaluated - suboptimal hyperparameter estimates are produced. We study the sampling variances of the importance-weighted versus the oracle estimator as a function of the relative scale of the training data. We show that introducing a control variate can reduce the variance of the importance-weighted risk estimator, which leads to superior regularization parameter estimates when the training data is much smaller in scale than the test data.

This chapter is based on the paper "Reducing sampling variance in covariate shift using control variates".

3.1. Introduction

In many real-world classification problems, training data is not gathered in a completely unbiased manner. An *unbiased* sample refers to events that were observed according to their true probabilities, whereas a *biased* sample refers to events that were observed more/less frequently [1–3]. For example, clinical data collected from a single hospital will be biased because the local patient population deviates from the global patient population. Consequently, a computer-aided diagnosis system trained on data from that hospital will not generalize well to hospitals in other countries. Unfortunately, collecting completely unbiased data can be extremely difficult. Instead, we are interested in statistical models that correct for biased samplings and generalize to target populations [4–8]. In particular, we propose an adjusted correction procedure that will aid hyperparameter optimization.

In classification settings, bias corrections are often performed based on individual sample probabilities: each sample is weighed by a factor that matches its current probability to the probability of encountering it in the target population. For example, if a particular event occurs very frequently in the training set but rarely in the target population, then it is not deemed important. Vice versa, if it occurs very rarely in the training set but often in the target population, then it is deemed important. As such, this correction is known as importance weighting [9]. Sample importance originates from Monte Carlo (MC) simulation, where it is used to draw samples from rare yet interesting regions of a distribution [10, 11]. The main difference between importance sampling in Monte Carlo simulation and importance weighing in a classification setting is that in the former case the importance sampling distribution is *designed*, whereas, in the latter case, it is *fixed*; it consists of the already collected biased training data. Although importance weighting can be very useful in controlling for biases in data, there are also a number of practical problems. The predominant one is weight bimodality: a small number of samples are assigned a very large weight while the remainder is assigned a near-zero weight. Essentially, only a handful of samples are deemed important, which greatly reduces effective sample size [12].

We focus on cross-validation in the face of biased data. More specifically, we consider the example of selecting a regularization parameter for a least-squares classifier [13]. If the collected training data were unbiased, a classifier can be evaluated by holding out a portion of the training data, training on the remainder and validating on the held-out set. Splitting the dataset into k parts where each is hold out once, is called k-fold cross-validation [14, 15]. By repeating this procedure for different values of hyperparameters, such as regularization parameters, the parameter can be selected that generalizes best to unseen data. However, since the training data is biased, the hyperparameter estimate that is obtained through cross-validation will not be optimal with respect to the whole population [16, 17]. It is essentially over-fitted to the biased training data [18]. One could correct for the discrepancy caused by the biased data by assigning importance-weights to the validation set [9]. However, the weight variance scales the sampling variance of the cross-validation estimator, which affects its ability to select the optimal hyperparameter [19]. This chapter proposes an adjustment to the importance-weighted cross-validation estimator that counteracts the increase in sampling variance due to the importance weights.

The sampling variance of an estimator describes the variation in its estimates for different datasets. This sampling variance depends on the size of the sample: as the estimator gets more samples, it will return more accurate estimates. However, the size of a given dataset is often fixed. Instead of increasing the number of samples in order to obtain a more accurate estimate, it is possible to directly reduce the sampling variance of the estimator [20]. In fact, there are many variance reduction techniques that were designed to make MC simulation more efficient and practical [11, 12]. These techniques incorporate additional information on the data distribution. For example, with antithetic variates one has the knowledge that the data-generating distribution is symmetric around some point. This knowledge can be exploited by mirroring the existing samples and augmenting the dataset [21]. Alternatively, a control variate consists of a function that is known to correlate strongly with the estimand. By subtracting a value from the estimand when the control variate rises and adding a value when the control variate shrinks, one reduces the estimator's deviation from its mean. It essentially returns more accurate estimates using the same dataset [22].

We show how we can use control variates to reduce the sampling variance of importance-weighted cross-validation (see Section 3.4). For the correlating function, we chose the importance weights themselves. Instead of scaling up the sampling variance of the estimator whenever the weight variance increases, it now helps us to perform more accurate estimations. Furthermore, we show that this improved risk estimator can be used to evaluate classifiers and leads to better hyperparameters when employed in cross-validation (see Section 3.5). In the next section we first introduce the problem setting, known as covariate shift [3], in more detail (see Section 3.2).

3.2. Covariate shift

In this section, we introduce some concepts and notation, followed by an explanation of covariate shift along with an example that will be used throughout this chapter.

3.2.1. Notation

Biased training data that stems from local sampling and unbiased test data that stems from global sampling can be described as different *domains*. A domain in this context is defined as the combination of an input space $\mathcal{X}$, an output space $\mathcal{Y}$ and an associated probability distribution p. Given two domains, we call them different if they are different in at least one of their constituent components, i.e., the input space, the output space, or the probability density function.

We focus on the case where only the probability distributions differ. Inputs remain the same, namely the D-dimensional real space $\mathbb{R}^D$ and outputs stay the same as well, namely the classes $\mathcal{Y} = \{-1, +1\}$. We denote the source domain as $(\mathcal{X}, \mathcal{Y}, p_{\mathcal{S}})$ and will refer to it as $\mathcal{S}$. The target domain is denoted $(\mathcal{X}, \mathcal{Y}, p_{\mathcal{T}})$ with the shorthand $\mathcal{T}$. The challenge is to use information from the source domain to generalize to the target domain.

Domain-specific functions will be marked with the subscript $\mathcal{S}$ or $\mathcal{T}$ as well, for example $\mathbb{E}_{\mathcal{T}}$. With some abuse of notation for the sake of clarity, we will mark marginal and conditional distributions with $\mathcal{S}$ and $\mathcal{T}$ as well: $p_{\mathcal{T}}(x, y)$ for the target joint distribution, $p_{\mathcal{T}}(x)$

for the target data marginal distribution and $p_{\mathcal{T}}(x \mid y)$ for the target class-conditional distribution. The source data is denoted as the set $\{(x_i, y_i)\}_{i=1}^{n}$. Note that x refers to an element of the input space $\mathcal{X}$, while x_i refers to a specific observation drawn from the source distribution, $x_i \sim p_{\mathcal{S}}$. Likewise, the target domain data consists of the set $\{(z_j, u_j)\}_{j=1}^{m}$.

3.2.2. Specifics of covariate shift

Covariate shift refers to the case where the class-posterior distributions remain equal, $p_{\mathcal{S}}(y \mid x) = p_{\mathcal{T}}(y \mid x)$. Furthermore, it is assumed that the class-priors are equal in both domains as well, $p_{\mathcal{S}}(y) = p_{\mathcal{T}}(y)$. It is therefore called covariate shift because only the covariates - the marginal data distributions - have shifted; $p_{\mathcal{S}}(x) \neq p_{\mathcal{T}}(x)$.

Throughout the chapter, we will use a running example of a basic covariate shift setting to illustrate several concepts: the target data distribution is set to be a normal distribution with mean 0 and standard deviation 1, $p_{\mathcal{T}}(x) = \mathcal{N}(x \mid 0, 1)$, its priors are set equal $p_{\mathcal{T}}(y) = 1/2$, and its class-posterior distribution is set to a cumulative normal distribution with mean 0 and standard deviation 1, $p_{\mathcal{T}}(y \mid x) = \Phi(yx)$. As the goal is to create a covariate shift setting, the target's class-posterior distribution is set to be equal to the source's: $p_{\mathcal{T}}(y \mid x) = p_{\mathcal{S}}(y \mid x)$. The source's priors are set to be equal as well, $p_{\mathcal{S}}(y) = 1/2$, but its data marginal distribution is set to be a normal distribution with mean 0 and standard deviation γ, $p_{\mathcal{S}}(x) = \mathcal{N}(x \mid 0, \gamma)$. γ controls the scale of the source domain. The further γ deviates away from 1 (the target domain's scale in this example setting) in either direction, the further the domain dissimilarity increases.

Figure 3.1 plots the distributions of the example; the left column corresponds to the source domain and the right column to the target domain. The top row corresponds to the data distributions $p(x)$, the middle row to the class-posteriors $p(y \mid x)$ and the bottom row to the class-conditional distributions $p(x \mid y)$. Red lines represent the negative class, while blue lines represent the positive class. For this figure, we visualized the case of $\gamma = 2$.

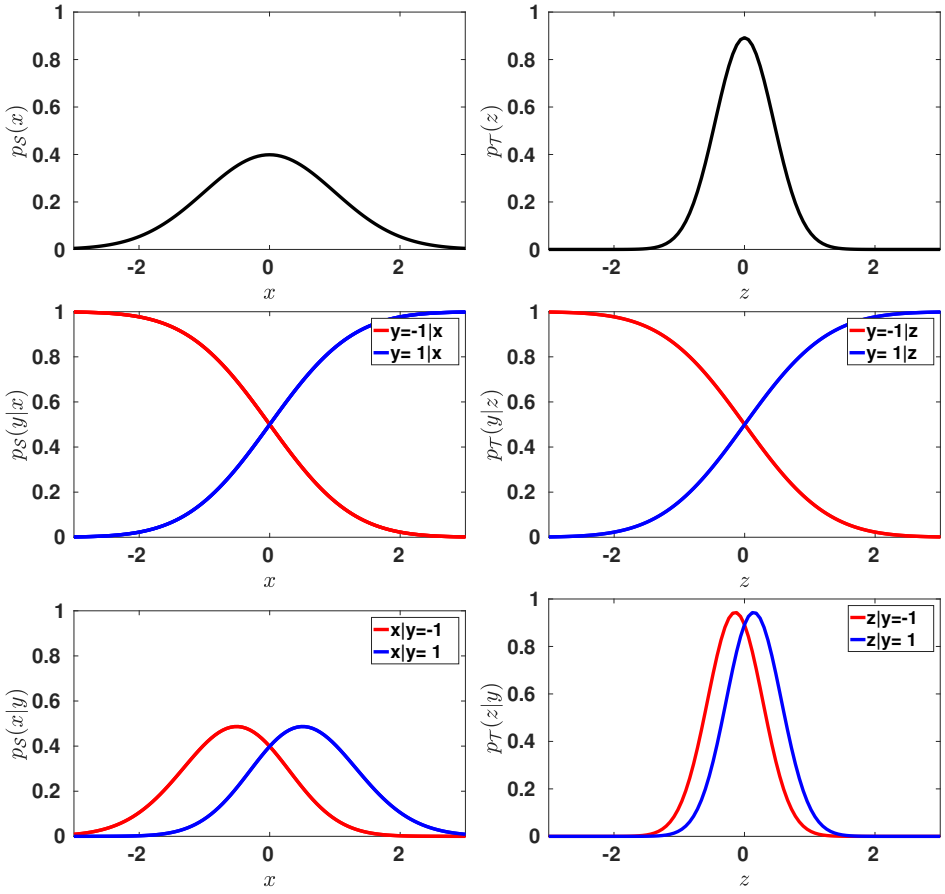


Figure 3.1: Example of a covariate shift setting: the left column shows the source domain and the right column the target domain. The top row plots the data distributions $p(x)$, the middle row the class-posterior distributions $p(y \mid x)$ and the right column the class-conditional distributions $p(x \mid y)$. The only difference between the domains is their standard deviation, which is set to 1 in the target domain and 2 in the source domain.

3.3. Importance-weighting

The empirical risk minimization framework describes a classifiers performance by its expected loss. The risk function integrates the loss ℓ of the classifiers parameters θ over the joint distribution p and is hence domain-specific. We are interested in generalizing to the target domain, which is another way of saying that we are interested in the classifier that minimizes the target risk $R_{\mathcal{T}}$:

$$R_{\mathcal{T}}(\theta) = \int_{\mathcal{X}} \sum_{y \in \mathcal{Y}} \ell\left(h\left(x \mid \theta\right), y\right) p_{\mathcal{T}}\left(x, y\right) \, \mathrm{d}x \,.$$

This integral is an expected value, $R_{\mathcal{T}}(\theta) = \mathbb{E}_{\mathcal{S}}\left[\ell\left(h\left(x \mid \theta\right), y\right)\right]$, which can be esti-

mated through the sample average of data drawn from the target domain:

$$\hat{R}_{\mathcal{T}}(\theta) = \frac{1}{m}\sum_{j=1}^{m} \ell\left(h\big(z_j \mid \theta\big), u_j\right) . \tag{3.1}$$

We will refer to estimators with a $\hat{}$ symbol. By the law of large numbers, the estimated value will converge to the true target risk: $\lim_{m\to\infty} \hat{R}_{\mathcal{T}}(\theta) = R_{\mathcal{T}}(\theta)$ for all θ [23].

3

Note that target labels $\{u_j\}_{j=1}^{m}$ are required for this risk estimator. Unfortunately, these are not available in a covariate shift problem setting. Consequently, we are interested in estimators of the target risk that do *not* depend on the target labels. One of the most popular ones is the importance-weighted risk estimator. It starts by multiplying and dividing the target distribution with the source distribution as follows:

$$R_{\mathcal{W}}(\theta) = \int_{\mathcal{X}} \sum_{y\in\mathcal{Y}} \ell\left(h\left(x \mid \theta\right), y\right) \frac{p_{\mathcal{T}}(x,y)}{p_{\mathcal{S}}(x,y)} p_{\mathcal{S}}(x,y)\, \mathrm{d}x\,.$$

Under the assumption that the class-posterior distributions are equivalent, $p_{\mathcal{T}}(y \mid x) = p_{\mathcal{S}}(y \mid x)$, the importance-weighted risk simplifies to [24]:

$$R_{\mathcal{W}}(\theta) = \int_{\mathcal{X}} \sum_{y\in\mathcal{Y}} \ell\left(h\left(x \mid \theta\right), y\right) \frac{p_{\mathcal{T}}(x)}{p_{\mathcal{S}}(x)} p_{\mathcal{S}}\left(x, y\right)\, \mathrm{d}x\,.$$

For this risk we can again formulate an estimator based on the sample average. Except this time, data from the source domain is used:

$$\hat{R}_{\mathcal{W}}(\theta) = \frac{1}{n}\sum_{i=1}^{n} \ell(h(x_i \mid \theta), y_i)\, \frac{p_{\mathcal{T}}(x_i)}{p_{\mathcal{S}}(x_i)}\,. \tag{3.2}$$

Note that this estimator does not depend on target labels u.

We will abbreviate the ratio of probability distributions through $p_{\mathcal{T}}(x)/p_{\mathcal{S}}(x) = w(x)$. Equation 3.2 already shows why importance-weighting can be problematic: 1 over a small probability equals a very large weight. In the example setting laid out in Section 3.2.2, the weight function can be derived: $w(x) = \gamma \exp\left(-x^2(\gamma^2-1)/(2\gamma^2)\right)$. In this case, the importance weights are an exponential function of the domain dissimilarity $(\gamma^2 - 1)$ and can become very large, very quickly. In particular, if we take the variance of the weights with respect to the source distribution, $\mathbb{V}_{\mathcal{S}}[w(x)] = -1 + \gamma^2/\sqrt{2\gamma^2 - 1}$, then we can identify two scenario's: for $\gamma > 1$ the variance rises slowly, while for $\gamma < 1$ the variance diverges to infinity as γ approaches $1/\sqrt{2}$ (see Figure 3.2). The former scenario corresponds to the case where the source domain is larger in scale and the goals is to generalize to a particular subset. The latter scenario corresponds to the case where the source domain is smaller in scale and the goal is to generalize to a larger population. Based on the weight variance, it seems that the latter case is far less feasible than the former.

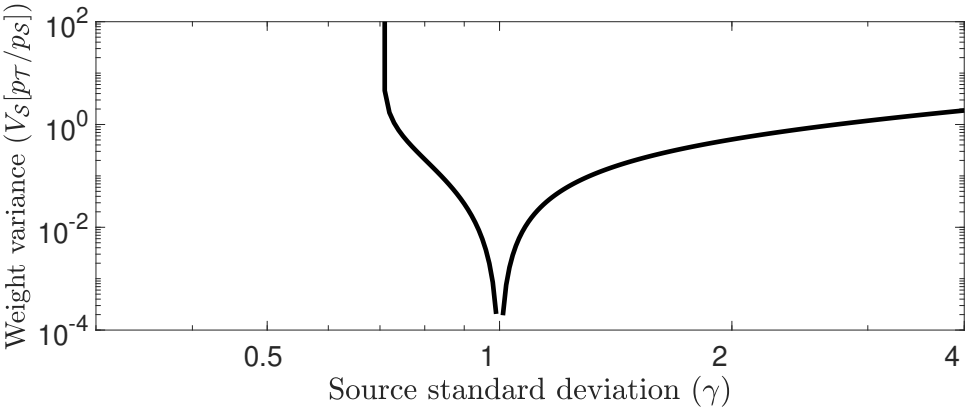


Figure 3.2: Variance of the importance weights as a function of domain dissimilarity, for the example problem setting described in Section 3.2.2.

3.3.1. Sampling variances

Although the expected values of $\hat{R}_T$ and $\hat{R}_W$ are the same, they behave differently for finite sample sizes. It is much more difficult to estimate the target risk using samples from another domain; $\hat{R}_W$ estimates tend to vary much more than $\hat{R}_T$'s ones for a fixed sample size. The variance of an estimator with respect to its samples is known as the *sampling variance* (not to be confused with sample variance, which is the variance between samples in a set). In the following, we will compare the sampling variance of $\hat{R}_T$ versus that of $\hat{R}_W$. The sampling variance with respect to a set of samples consists of the average squared deviation of the estimator from its true risk:

$$\mathbb{V}_T[\hat{R}_T] = \mathbb{E}_T\Big[\big(\hat{R}_T - R_T\big)^2\Big]. \tag{3.3}$$

Using the fact that samples are drawn independently and are identically distributed (iid), (3.3) can be simplified by pulling the sum over samples outside of the expectation:

$$\begin{aligned}\mathbb{E}_T\Big[\big(\hat{R}_T - R_T\big)^2\Big] &= \mathbb{E}_T[(\frac{1}{m}\sum_{j=1}^{m}\ell(h(z_j \mid \theta), u_j) - R_T)^2] \\ &= \frac{1}{m^2}\sum_{j=1}^{m}\mathbb{E}_T[(\ell(h(z_j \mid \theta), u_j) - R_T)^2] \\ &= \frac{1}{m}\mathbb{E}_T[(\ell(h(x \mid \theta), y) - R_T)^2]. \end{aligned} \tag{3.4}$$

The sampling variance with respect to a single sample, $\mathbb{E}_T[(\ell(h(x \mid \theta), y) - R_T)^2]$, will be referred to as σ_T^2.

The source data is drawn iid as well. That means that the same simplification holds for the importance-weighted risk estimator:

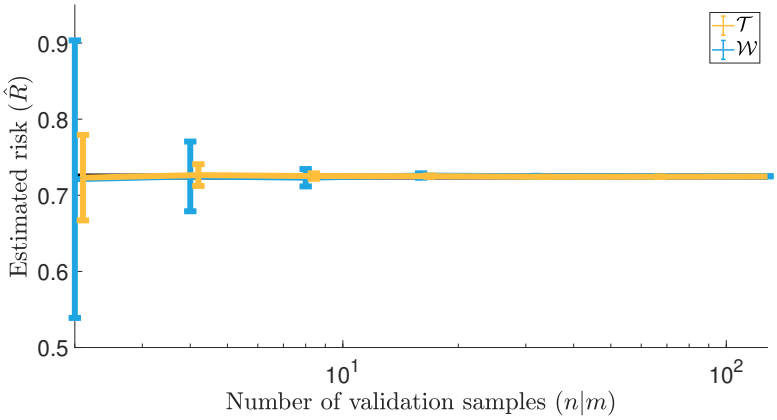
$$\begin{aligned}\mathbb{V}_{\mathcal{S}}[\hat{R}_{\mathcal{W}}] &= \mathbb{E}_{\mathcal{S}}\Big[\left(\hat{R}_{\mathcal{W}} - R_{\mathcal{W}}\right)^2\Big] \\ &= \frac{1}{n}\mathbb{E}_{\mathcal{S}}[(\ell(h(x \mid \theta), y)w(x) - R_{\mathcal{W}})^2]\,. \end{aligned} \tag{3.5}$$

Similar to before, the sampling variance with respect to a single sample will be referred to as $\sigma^2_{\mathcal{W}}$.

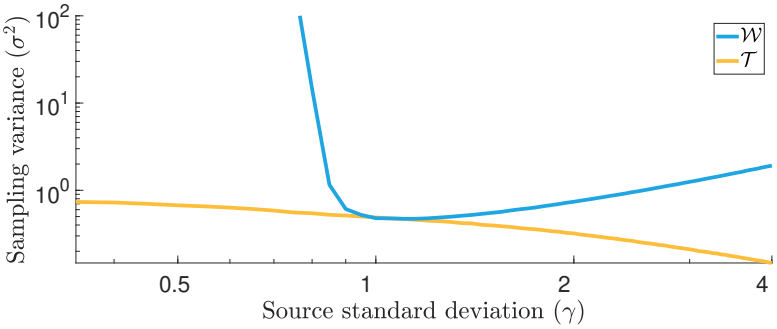
Expanding the squares in (3.4) leads to $\mathbb{E}_{\mathcal{T}}[\ell(h(x \mid \theta), y)^2] - R^2_{\mathcal{T}}$ and expanding (3.5) leads to $\mathbb{E}_{\mathcal{T}}[\ell(h(x \mid \theta), y)^2 w(x)] - R^2_{\mathcal{W}}$. Note that $R_{\mathcal{W}} = R_{\mathcal{T}}$ and that the only difference between $\sigma^2_{\mathcal{T}}$ and $\sigma^2_{\mathcal{W}}$ is the addition of the importance weights. Thus, the weights directly scale the sampling variance of the estimator. So, even though $\hat{R}_{\mathcal{W}}$ and $\hat{R}_{\mathcal{T}}$ are estimators of the same risk, the fact that $\hat{R}_{\mathcal{W}}$ is based on data from another distribution makes it a less accurate estimator.

Figure 3.3a computes the estimators for the running example using a least-squares classifier: $\ell(h(x \mid \theta), y) = \left(x\theta^\mathsf{T} - y\right)^2$ with $\theta = [\theta_1, \theta_0]$ [25]. Since the probability distributions are known, the Bayes optimal classifier for the source domain can be derived: $\theta^* = [\sqrt{2/\pi}/\sqrt{1+\gamma^2},\ \ 0]$. This θ^* was used to compute the risk estimates. The figure shows a learning curve of 10^5 repetitions of the estimated risk as a function of the size of the validation set (m and n for the target and the importance-weighted risk estimators respectively). A source standard deviation of $\gamma = 2$ was chosen for this visualization. Note that the importance-weighted risk varies much more than the target risk.

Figure 3.3b displays the sampling variance of $\hat{R}_{\mathcal{T}}$ and $\hat{R}_{\mathcal{W}}$ as a function of the domain dissimilarity. For $\gamma = 1$, the domains are the same and the sampling variances are equal. For $\gamma > 1$, the sampling variance of $\hat{R}_{\mathcal{T}}$ drops off, while the sampling variance of $\hat{R}_{\mathcal{W}}$ slowly increases. For $\gamma < 1$, the $\hat{R}_{\mathcal{T}}$'s variance remains relatively steady, while the $\hat{R}_{\mathcal{W}}$'s variance diverges to infinity at $\gamma = 1/\sqrt{2}$. The shape of this curve reflects the influence of the variance of the importance weights, as shown in Figure 3.2.



(a) Estimated risk of the target (yellow) versus the importance-weighted (blue) estimator as a function of the number of validation set samples, for the example setting with $\gamma = 2$.



(b) Sampling variance of the target (yellow) versus importance-weighted (blue) risk estimators as a function of the source domain standard deviation γ.

Figure 3.3: Comparison of the importance-weighted source versus the target risk estimator.

3.4. Reducing sampling variance

The increased sampling variance of the importance-weighted risk estimator is problematic for procedures that rely on accurate estimates of the target risk. One such procedure is cross-validation, which we discuss in Section 3.5. Our goal is to reduce the sampling variance of $\hat{R}_{\mathcal{W}}$. To that end, we will introduce a control variate [12, 22]. A control variate is a function that correlates with the estimator and whose expected value is known: $(\hat{\alpha} - \mathbb{E}[\hat{\alpha}])$. These two properties mean that it essentially contains additional information on the function of interest, which can be used to reduce sampling variance. Whenever the correlating function's value rises above its expected value, $(\hat{\alpha} - \mathbb{E}[\hat{\alpha}]) > 0$, so does the risk estimator's value rise above its expected value (the true risk), $(\hat{R} - R) > 0$. By subtracting the control variate from the risk estimate, $\hat{R} - (\hat{\alpha} - \mathbb{E}[\hat{\alpha}])$, the estimator's deviation from the true risk is reduced. Hence, its variance is reduced. It is however important that the control variate is appropriately scaled, as subtracting a too large value can increase the sampling variance

as well. A parameter β is used to control the scaling: $\hat{R} - \beta(\hat{\alpha} - \mathbb{E}[\hat{\alpha}])$.

We chose the importance-weights $w(x)$ themselves as the control variate, since their expected value is known: $\mathbb{E}_{\mathcal{S}}[w(x)] = \int_{\mathcal{X}} w(x) p_{\mathcal{S}}(x) \mathrm{d}x = \int_{\mathcal{X}} p_{\mathcal{T}}(x) \mathrm{d}x = 1$. Adding the weight-based control variate to the importance-weighted risk, forms the following estimator:

$$\hat{R}_\beta = \frac{1}{n} \sum_{i=1}^{n} \ell(h(x_i \mid \theta), y_i)\, w(x_i) - \beta(w(x_i) - 1)\,.$$

Note that the added term does not bias the overall estimator: the expected value of the control variate $\mathbb{E}_{\mathcal{S}}[\beta(w(x_i) - 1)] = \beta(\mathbb{E}_{\mathcal{S}}[w(x_i)] - 1) = 0$ for all values of β. This means that the expected value of the controlled estimator is the same as that of the importance-weighted estimator: $\mathbb{E}_{\mathcal{S}}[\hat{R}_\beta] = R_{\mathcal{W}}$.

3.4.1. Sampling variance of the controlled estimator

The effect of the control variate on the sampling variance of the importance-weighted risk estimator can be described exactly [22]:

$$\begin{aligned}
\mathbb{V}_{\mathcal{S}}[\hat{R}_\beta] &= \mathbb{E}_{\mathcal{S}}\Big[\big(\hat{R}_\beta - R_\beta\big)^2\Big] \\
&= \mathbb{E}_{\mathcal{S}}\Big[\Big(\frac{1}{n}\sum_{i=1}^{n} \ell(h(x_i \mid \theta), y_i) w(x_i) - \beta(w(x_i) - 1) - R_{\mathcal{W}}\Big)^2\Big] \\
&= \frac{1}{n}\mathbb{E}_{\mathcal{S}}\Big[\Big(\ell(h(x \mid \theta), y) w(x)\Big)^2 - \ell(h(x \mid \theta), y) w(x) \beta(w(x) - 1) \\
&\quad - \ell(h(x \mid \theta), y) w(x) R_{\mathcal{W}} - \beta(w(x) - 1)\ell(h(x \mid \theta), y) w(x) \\
&\quad + \beta^2 (w(x) - 1)^2 + \beta(w(x) - 1) R_{\mathcal{W}} - R_{\mathcal{W}} \ell(h(x \mid \theta), y) w(x) \\
&\quad + R_{\mathcal{W}}\, \beta(w(x) - 1) + R_{\mathcal{W}}^2\Big] \\
&= \frac{1}{n}\mathbb{E}_{\mathcal{S}}\Big[\Big(\ell(h(x \mid \theta), y) w(x) - R_{\mathcal{W}}\Big)^2 \\
&\qquad - 2\beta\Big(w(x) - 1\Big)\Big(\ell(h(x \mid \theta), y) w(x) - R_{\mathcal{W}}\Big) \\
&\qquad + \beta^2 (w(x) - 1)^2\Big] \\
&= \frac{1}{n}\Big[\sigma^2_{\mathcal{W}} - 2\beta\, \mathbb{C}_{\mathcal{S}}\Big[\ell(h(x \mid \theta), y) w(x), w(x)\Big] + \beta^2\, \mathbb{V}_{\mathcal{S}}[w(x)]\Big]. \qquad (3.6)
\end{aligned}$$

The $\mathbb{C}_{\mathcal{S}}$ stands for the covariance, in this case between the weighted loss and the weights themselves. The scale parameter β of the control variate can be optimized to minimize the overall sampling variance of the estimator:

$$\begin{aligned}
\frac{\partial}{\partial \beta^*}\Big[\frac{\sigma^2_{\mathcal{W}}}{n} - \frac{2\beta^*}{n}\mathbb{C}_{\mathcal{S}}\Big[\ell(h(x \mid \theta), y) w(x), w(x)\Big] + \frac{\beta^{*2}}{n}\mathbb{V}_{\mathcal{S}}[w(x)]\Big] &= 0 \\
-\frac{2}{n}\,\mathbb{C}_{\mathcal{S}}\Big[\ell(h(x \mid \theta), y) w(x), w(x)\Big] + \frac{2}{n}\beta^*\, \mathbb{V}_{\mathcal{S}}[w(x)] &= 0 \\
\mathbb{C}_{\mathcal{S}}\left[\ell(h(x \mid \theta), y) w(x), w(x)\right] \;/\; \mathbb{V}_{\mathcal{S}}\left[w(x)\right] &= \beta^*
\end{aligned}$$

where β^* is the minimizer. Plugging β^* back in to (3.6) simplifies the sampling variance to:

$$\begin{aligned}\sigma_\beta^2 &= \sigma_W^2 - 2\,\mathbb{C}_S\Big[\ell(h(x \mid \theta), y)w(x), w(x)\Big] \,/\, \mathbb{V}_S\Big[w(x)\Big]\,\mathbb{C}_S\Big[\ell(h(x \mid \theta), y)w(x), w(x)\Big] \\ &\quad + \Big(\,\mathbb{C}_S\Big[\ell(h(x \mid \theta), y)w(x), w(x)\Big] \,/\, \mathbb{V}_S\Big[w(x)\Big]\Big)^2\,\mathbb{V}_S[w(x)] \\ &= \sigma_W^2 - \mathbb{C}_S\Big[\ell(h(x \mid \theta), y)w(x), w(x)\Big]^2 \,/\, \mathbb{V}_S\Big[w(x)\Big]. \end{aligned} \tag{3.7}$$

Considering that both the squared covariance term and the variance term are non-negative, the sampling variance of a controlled estimator is never larger than that of the standard estimator [26]. In particular, multiplying $\mathbb{C}_S[\ell(h(x \mid \theta), y)w(x), w(x)]^2$ $/\ \mathbb{V}_S[w(x)]$ with σ_W^2/σ_W^2, allows (3.7) to be written as:

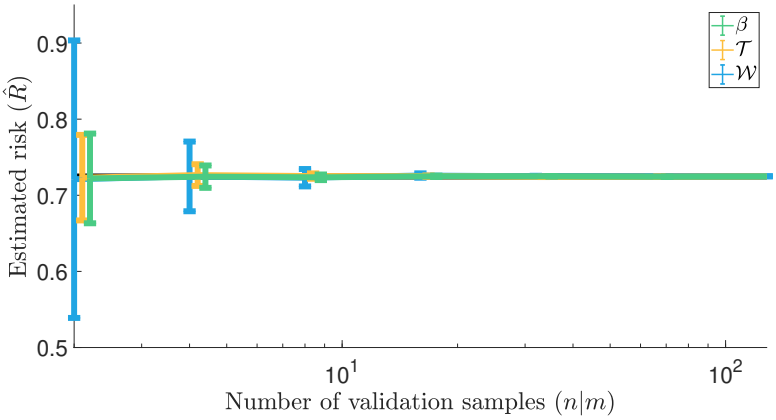
$$\sigma_\beta^2 = \sigma_W^2(1 - \rho^2)\,,$$

where ρ denotes the correlation between the weighted loss (the estimand), and the weights (the control variate). Essentially, the more the weights correlate - positively or negatively - with the weighted loss, the larger the reduction in variance.

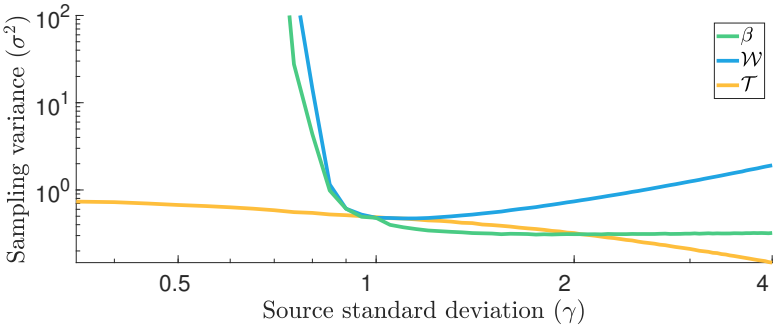
Computing β^* is not possible without knowledge of the probability distributions, but it can be estimated from data:

$$\hat{\beta} = \Big[\frac{1}{n}\sum_i^n \big(\ell(h(x_i \mid \theta), y_i)w(x_i) - \hat{R}_W^n\big)\big(w(x_i) - 1\big)\Big] \,/ \Big[\frac{1}{n}\sum_i^n \big(w(x_i) - 1\big)^2\Big].$$

Figure 3.4a provides an illustration similar to the one of Figure 3.3a), but adds the estimated risk of the controlled importance-weighted estimator. This is still the case of $\gamma = 2$, for which $\hat{R}_\beta$'s sampling variance is much smaller than that of $\hat{R}_W$. Similarly, Figure 3.4b is the equivalent of Figure 3.3b, which plots the sampling variance for the three estimators $\hat{R}_\beta$, $\hat{R}_W$, and $\hat{R}_\mathcal{T}$. For $\gamma > 1$, the sampling variance of the controlled estimator σ_β^2 reduces to roughly the same level as the original target risk estimator $\sigma_\mathcal{T}^2$. For $\gamma < 1$, σ_β also diverges at $1/\sqrt{2}$, but rises much more slowly.



(a) Estimated risk of the target (yellow), the importance-weighted (blue) and the controlled importance-weighted (green) estimator estimator as a function of the number of validation set samples, for the example setting with $\gamma = 2$.



(b) Sampling variance of the target (yellow), the importance-weighted (blue) and the controlled importance-weighted (green) risk estimators as a function of the source domain standard deviation γ.

Figure 3.4: The effect of the addition of the control variate.

3.5. Cross-validation

Accurate estimation of the target risk is important for cross-validation, which is, in turn, important for hyperparameter optimization. In this case, it is used to find an optimal regularization parameter. In order to account for the covariate shift, the validation data is importance-weighted [9]. However, as Section 3.3.1 has shown, weighting can increase the sampling variance, making the cross-validation estimator less accurate. Fortunately, the control variate can counteract this negative influence. The following subsections describe an experiment that compares the importance-weighted versus the controlled importance-weighted risk estimators in a cross-validation context.

3.5.1. Experimental setup

In the following experiments, the source set is split up into 10 folds, each of which is held out once for validation. Training samples are marked as $\{(x_t, y_t)\}_{t\in T}$ and validation samples are marked as $\{(x_v, y_v)\}_{v\in V}$, where the sets T and V together make up the total index set of the source samples $T \cup V = \{1, \ldots, n\}$. For the classifier, we employ a kernelized version of a regularized least-squares classifier [13]. So for risk evaluation, we evaluate the mean squared error (MSE): $\ell(h(x \mid \theta), y) = (h(x \mid \theta) - y)^2$. In particular, a quadratic polynomial kernel is taken: $\theta_\lambda = \sum_{t\in T}(\kappa(x_{t'}, x_t) + \lambda I)^{-1} y_t$, with $\kappa(x', x) = (x' x^\mathsf{T} + 1)^2$. Note that the classifier's parameters θ_λ are dependent on the regularization parameter λ. Predictions are made by applying the kernel to new samples and taking the inner product with the classifier parameters: $h(\cdot \mid \theta_\lambda) = \kappa(\cdot, x_t)\theta_\lambda$.

The true data marginal distributions are not known in practice. In most cases it is also not known to which family of distributions the data marginals belong to. As such, we opt for a nonparametric approach. Both the source and target distributions are estimated with a kernel density estimator [27]. A normal kernel was used, with its bandwidth set through Silverman's rule of thumb [28]. After estimation, the ratio of distributions is taken to compute the importance weights: $\hat{w}(\cdot) = \left(m^{-1}\sum_j^m \kappa(z_j, \cdot)\right) / \left(n^{-1}\sum_i^n \kappa(x_i, \cdot)\right)$.

We compare the following 4 risk estimators:

$$\hat{R}_S(\theta_\lambda) = \frac{1}{|V|}\sum_{v\in V}\left(\kappa(x_v, x_t)\,\theta_\lambda - y_v\right)^2$$

$$\hat{R}_W(\theta_\lambda) = \frac{1}{|V|}\sum_{v\in V}\left(\kappa(x_v, x_t)\,\theta_\lambda - y_v\right)^2 \hat{w}(x_v)$$

$$\hat{R}_\beta(\theta_\lambda) = \frac{1}{|V|}\sum_{v\in V}\left(\kappa(x_v, x_t)\,\theta_\lambda - y_v\right)^2 \hat{w}(x_v) - \hat{\beta}\,(\hat{w}(x_v) - 1)$$

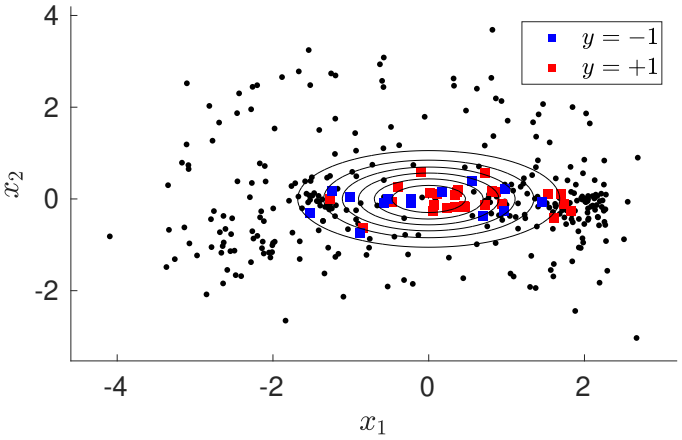
$$\hat{R}_T(\theta_\lambda) = \frac{1}{m}\sum_{j=1}^{m}\left(\kappa(z_j, x_t)\,\theta_\lambda - u_j\right)^2 .$$

$\hat{R}_S$ corresponds to validating on source data without compensating for the covariate shift, $\hat{R}_W$ compensates with the importance weights, $\hat{R}_\beta$ uses the control variate, and $\hat{R}_T$ corresponds to the oracle case, i.e., validating on labeled target samples. $|V|$ refers to the cardinality of the validation set. We start with a set of 100 regularization parameter values, ranging from 0 to n. The 4 risk estimators are used to select the $\hat{\lambda}$ for which the risk is minimal. This selected parameter is then used to train a classifier on all the source data and is evaluated using the target risk estimator.

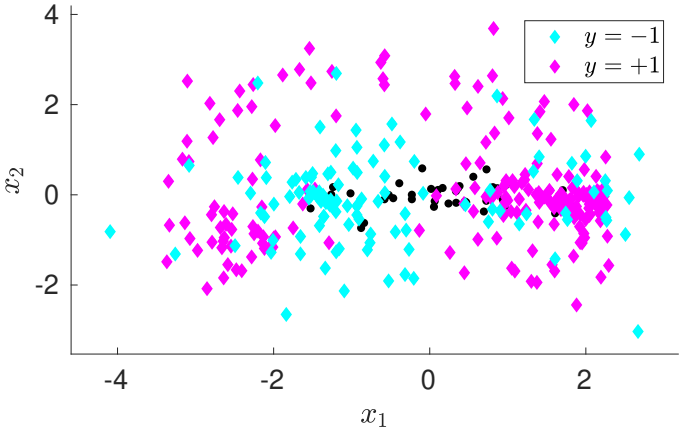
3.5.2. Data

The ionosphere dataset from the UCI machine learning repository was used. To allow for visualization, the dimensionality of the data was reduced to 2 using principal components

analysis. To simulate a covariate shift setting, we perform a biased sampling: a normal distribution is placed at the center with a standard deviation of γ times the covariance matrix of the whole set. Each sample from the ionosphere dataset is evaluated under this distribution and the resulting probabilities are used to draw - without replacement - a subset of 50 samples. γ is chosen from a logarithmic range between 2^{-3} and 2^4, which represents a very local, biased sampling to a nearly uniform, unbiased sampling. Figure 3.5 shows scatterplots of samples selected as part of the source domain (top) and the remainder as part of the target domain (bottom), for $\gamma = 0.5$.



(a) Source domain. Red markers denote the positive class, blue the negative one and black the unselected samples. The black ellipses denote the source domain sampling distribution, for $\gamma = 0.5$.

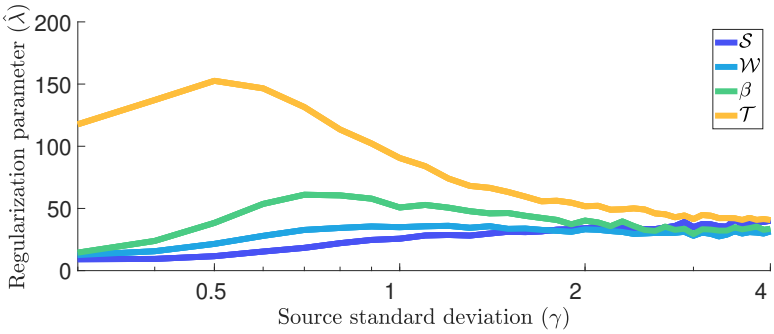


(b) Target domain. Magenta markers denote the positive class, cyan mark the negative one and black mark the previously selected source samples.

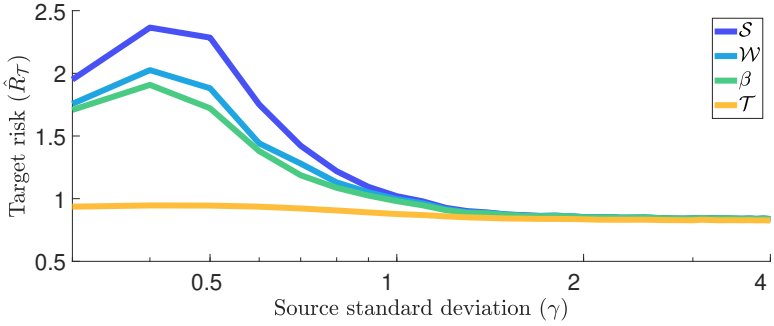
Figure 3.5: Example of the biased sampling for the ionosphere dataset.

3.5.3. Results

Figure 3.6a plots the estimated regularization parameters as a function of the scale of the source domain's sampling distribution. The value of the optimal regularization parameter tends to be quite large, but decreases from $\gamma > 0.5$ onwards. $\hat{R}_{\mathcal{W}}$ and $\hat{R}_{\beta}$ differ much more in the regime $0.5 < \gamma < 2$. From $\gamma > 2$ onwards, the source domain covers the dataset so well, that all samples evaluate to nearly the same probability under the source domain's sampling distribution. Hence, the selected data is an unbiased sample and there is no covariate shift. Figure 3.6b shows the risk of the estimated regularization parameter and indicates that the large differences between estimated regularization parameters cause large differences in the resulting risks. Conversely, no difference in $\hat{\lambda}$ causes no difference in risks, from $\gamma > 2$ onwards. The improvement from $\hat{R}_{\mathcal{W}}$ over $\hat{R}_{\mathcal{S}}$ is largest where the domains are the most different, as is the improvement of $\hat{R}_{\beta}$ over $\hat{R}_{\mathcal{W}}$. Overall, $\hat{R}_{\beta}$ always leads to superior or equal estimates compared to $\hat{R}_{\mathcal{W}}$.



(a) Regularization parameter λ estimated by $\hat{R}_{\mathcal{S}}$ (dark blue), $\hat{R}_{\mathcal{W}}$ (light blue), $\hat{R}_{\beta}$ (green) and $\hat{R}_{\mathcal{T}}$ (yellow), as a function of domain dissimilarity.



(b) Target risk resulting from training with the selected regularization parameter estimated by $\hat{R}_{\mathcal{S}}$ (dark blue), $\hat{R}_{\mathcal{W}}$ (light blue), $\hat{R}_{\beta}$ (green) and $\hat{R}_{\mathcal{T}}$ (yellow), as a function of domain dissimilarity.

Figure 3.6: Results for the experiment on the ionosphere dataset.

3.6. Conclusion

We presented a study of the sampling variance of the importance-weighted risk estimator as compared to the target risk estimator in the context of covariate shift. We showed that the sampling variance can increase substantially as a function of the scale of the source domain, leading to a far less accurate estimator for a given sample size. Furthermore, we introduced a control variate to reduce the sampling variance of the importance-weighted risk estimator. This reduction is beneficial for hyperparameter optimization in cases where the sampling variance becomes problematic. As it is never detrimental, the controlled importance-weighted risk estimator is the preferred choice.

In this work, only the additive control variate has been studied. Multiplicative control variates or more complex functions applied to the additive control variate have the potential to increase its correlation with the estimand, thus decreasing the sampling variance of the estimator even further. However, it is hard to predict whether a more complex control variate will be useful.

References

[1] J. J. Heckman, *Sample selection bias as a specification error (with an application to the estimation of labor supply functions)*, (1977).

[2] J. M. Wooldridge, *Econometric analysis of cross section and panel data* (MIT Press, 2010).

[3] J. G. Moreno-Torres, T. Raeder, R. Alaiz-RodríGuez, N. V. Chawla, and F. Herrera, *A unifying view on dataset shift in classification*, Pattern Recognition **45**, 521 (2012).

[4] B. Zadrozny, *Learning and evaluating classifiers under sample selection bias*, in *International Conference on Machine Learning* (ACM, 2004) p. 114.

[5] S. Bickel, M. Brückner, and T. Scheffer, *Discriminative learning for differing training and test distributions*, in *International Conference on Machine Learning* (2007) pp. 81–88.

[6] C. Cortes, M. Mohri, M. Riley, and A. Rostamizadeh, *Sample selection bias correction theory*, in *Algorithmic Learning Theory* (Springer, 2008) pp. 38–53.

[7] J. Quionero-Candela, M. Sugiyama, A. Schwaighofer, and N. D. Lawrence, *Dataset shift in machine learning* (MIT Press, 2009).

[8] S. Ben-David, J. Blitzer, K. Crammer, A. Kulesza, F. Pereira, and J. W. Vaughan, *A theory of learning from different domains*, Machine Learning **79**, 151 (2010).

[9] M. Sugiyama, M. Krauledat, and K.-R. Müller, *Covariate shift adaptation by importance weighted cross validation*, Journal of Machine Learning Research **8**, 985 (2007).

[10] J. C. Helton, J. D. Johnson, C. J. Sallaberry, and C. B. Storlie, *Survey of sampling-based methods for uncertainty and sensitivity analysis*, Reliability Engineering & System Safety **91**, 1175 (2006).

[11] W. G. Cochran, *Sampling techniques* (John Wiley & Sons, 2007).

[12] A. B. Owen, *Monte Carlo theory, methods and examples* (2013).

[13] B. Schölkopf and A. J. Smola, *Learning with kernels: support vector machines, regularization, optimization, and beyond* (MIT Press, 2002).

[14] B. Efron and R. J. Tibshirani, *An introduction to the bootstrap* (CRC Press, 1994).

[15] R. Kohavi, *A study of cross-validation and bootstrap for accuracy estimation and model selection*, in *International Joint Conference on Artificial Intelligence* (1995).

[16] M. Sugiyama and K.-R. Müller, *Model selection under covariate shift*, in *International Conference on Artificial Neural Networks* (Springer, 2005) pp. 235–240.

[17] W. M. Kouw and M. Loog, *On regularization parameter estimation under covariate shift*, in *International Conference on Pattern Recognition* (IEEE, 2016) pp. 426–431.

[18] G. C. Cawley and N. L. Talbot, *On over-fitting in model selection and subsequent selection bias in performance evaluation*, Journal of Machine Learning Research **11**, 2079 (2010).

[19] M. Markatou, H. Tian, S. Biswas, and G. M. Hripcsak, *Analysis of variance of cross-validation estimators of the generalization error*, Journal of Machine Learning Research **6**, 1127 (2005).

[20] H. Kahn and A. W. Marshall, *Methods of reducing sample size in monte carlo computations*, Journal of the Operations Research Society of America **1**, 263 (1953).

[21] J. M. Hammersley and J. Mauldon, *General principles of antithetic variates*, in *Mathematical Proceedings of the Cambridge Philosophical Society*, Vol. 52 (Cambridge University Press, 1956) pp. 476–481.

[22] B. L. Nelson, *On control variate estimators*, Computers & Operations Research **14**, 219 (1987).

[23] L. Wasserman, *All of statistics: a concise course in statistical inference* (Springer, 2013).

[24] H. Shimodaira, *Improving predictive inference under covariate shift by weighting the log-likelihood function*, Journal of Statistical Planning and Inference **90**, 227 (2000).

[25] J. Friedman, T. Hastie, and R. Tibshirani, *The elements of statistical learning*, Vol. 1 (Springer, 2001).

[26] A. Owen and Y. Zhou, *Safe and effective importance sampling*, Journal of the American Statistical Association **95**, 135 (2000).

[27] E. Parzen, *On estimation of a probability density function and mode*, Annals of Mathematical Statistics **33**, 1065 (1962).

[28] B. W. Silverman, *Density estimation for statistics and data analysis*, Vol. 26 (CRC Press, 1986).

4

Modeling feature-level transfer

Domain adaptation is the supervised learning setting in which the training and test data are sampled from different distributions: training data is sampled from a source domain, whilst test data is sampled from a target domain. We propose and study an approach, called feature-level domain adaptation (FLDA)*, that models the dependence between the two domains by means of a feature-level transfer model that is trained to describe the transfer from source to target domain. Subsequently, we train a domain-adapted classifier by minimizing the expected loss under the resulting transfer model. For linear classifiers and a large family of loss functions and transfer models, this expected loss can be computed or approximated analytically, and minimized efficiently. Our empirical evaluation of* FLDA *focuses on problems comprising binary and count data in which the transfer can be naturally modeled via a dropout distribution, which allows the classifier to adapt to differences in the marginal probability of features in the source and the target domain. Our experiments on several real-world problems show that* FLDA *performs on par with state-of-the-art domain-adaptation techniques.*

This chapter is based on the paper ”Feature-level domain adaptation”.

4.1. Introduction

Domain adaptation is an important research topic in machine learning and pattern recognition that has applications in, among others, speech recognition [1], medical image processing [2], computer vision [3], social signal processing [4], natural language processing [5], and bioinformatics [6]. Domain adaptation deals with supervised-learning settings in which the common assumption that the training and the test observations stem from the same distribution is dropped. This learning setting may arise, for instance, when the training data is collected with a different measurement device than the test data, or when a model that is trained on one data source is deployed on data that comes from another data source. This creates a learning setting in which the training set contains samples from one distribution (the so-called source domain), whilst the test set constitutes samples from another distribution (the target domain). In domain adaptation, one generally assumes a transductive learning setting: that is, it is assumed that the unlabeled test data are available to us at training time and that the main goal is to predict their labels as well as possible.

The goal of domain-adaptation approaches is to exploit information on the dissimilarity between the source and target domains that can be extracted from the available data in order to make more accurate predictions on samples from the target domain. To this end, many domain adaptation approaches construct a *sample-level transfer model* that assigns importance weights to observations from the source domain in order the make the source distribution more similar to the target distribution [7–11]. In contrast to such sample-level reweighing approaches, in this work, we develop a *feature-level transfer model* that describes the shift between the target and the source domain for each feature individually. Such a feature-level approach may have advantages in certain problems: for instance, when one trains a natural language processing model on news articles (the source domain) and applies it to Twitter data (the target domain), the marginal distribution of some of the words or n-grams (the features) is likely to vary between target and source domain. This shift in the marginal distribution of the features cannot be modeled well by sample-level transfer models, but it can be modeled very naturally by a feature-level transfer model.

Our feature-level transfer model takes the form of a conditional distribution that, conditioned on the training data, produces a probability density of the target data. In other words, our model of the target domain thus comprises a convolution of the empirical source distribution and the transfer model. The parameters of the transfer model are estimated by maximizing the likelihood of the target data under the model of the target domain. Subsequently, our classifier is trained as to minimize the expected value of the classification loss under the target-domain model. We show empirically that when the true domain shift can be modeled by the transfer model, under certain assumptions, our domain-adapted classifier converges to a classifier trained on the true target distribution. Our feature-level approach to domain adaptation is general in that it allows the user to choose a transfer model from a relatively large family of probability distributions. This allows practitioners to incorporate domain knowledge on the type of domain shift in their models. In the experimental section, we focus on a particular type of transfer distribution that is well-suited for problems in which the features are binary or count data (as often encountered in natural language processing), but the approach we describe is more generally

applicable. In addition to experiments on artificial data, we present experiments on several real-world domain adaptation problems, which show that our feature-level approach performs on par with the current state-of-the-art in domain adaptation.

The outline of the remainder of this chapter is as follows. In Section 2, we give an overview of related prior work on domain adaptation. Section 3 presents our feature-level domain adaptation (FLDA) approach. In Section 4, we present our empirical evaluation of feature-level domain adaptation and Section 5 concludes the chapter with a discussion of our results.

4.2. Related Work

Current approaches to domain adaptation can be divided into one of three main types. The first type constitutes *importance weighting* approaches that aim to reweigh samples from the source distribution in an attempt to match the target distribution as well as possible. The second type are *sample transformation* approaches that aim to transform samples from the source distribution in order to make them more similar to samples from the target distribution. The third type are *feature augmentation* approaches that aim to extract features that are shared across domains. Our feature-level domain adaptation (FLDA) approach is an example of a sample-transformation approach.

4.2.1. Importance-weighting

Importance-weighting approaches assign a weight to each source sample in such a way as to make the reweighed version of the source distribution as similar to the target distribution as possible [7–13]. If the class posteriors are identical in both domains (that is, the covariate-shift assumption holds) and the importance weights are unbiased estimates of the ratio of the target density to the source density, then the importance-weighted classifier converges to the classifier that would have been learned on the target data if labels for that data were available [7].

Despite their theoretic appeal, importance-weighting approaches generally do not to perform very well when the data set is small, or when there is little ”overlap” between the source and target domain. In such scenarios, only a very small set of samples from the source domain is assigned a large weight. As a result, the effective size of the training set on which the classifier is trained is very small, which leads to a poor classification model. In contrast to importance-weighting approaches, our approach performs a *feature-level* reweighing. Specifically, FLDA assigns a data-dependent weight to each of the features that represents how informative this feature is in the target domain. This approach effectively uses all the data in the source domain and therefore suffers less from the small sample size problem.

4.2.2. Sample transformation

Sample-transformation approaches learn functions that make the source distribution more similar to the target distribution [14–22]. Most sample-transformation approaches learn global (non)linear transformations that map source and target data points into the same,

shared feature space in such a way as to maximize the overlap between the transformed source data and the transformed target data [16–19, 21].

Approaches that learn a shared subspace in which both the source and the target data are embedded often minimize the maximum mean discrepancy (MMD) between the transformed source data and the transformed target data [16, 19]. If used in combination with a universal kernel, the MMD criterion is zero when all the moments of the (transformed) source and target distribution are identical. Most methods minimize the MMD subject to constraints that help to avoid trivial solutions (such as collapsing all data onto the same point) via some kind of spectral analysis. An alternative to the MMD is the subspace disagreement measure (SDM) of [18], which measures the discrepancy of the angles between the principal components of the transformed source data and the transformed target data.

Most current sample-transformation approaches work well for ”global” domain shifts such as translations or rotations in the feature space, but are less effective when the domain shift is ”local” in the sense that it strongly nonlinear. Similar limitations apply to the FLDA approach we explore, but it differs in that (1) our transfer model does not learn a subspace but operates in the original feature space and (2) the measure it minimizes to model the transfer is different, namely, the negative log-likelihood of the target data under the transferred source distribution.

4.2.3. Feature augmentation

Several domain-adaptation approaches extend the source data and the target data with additional features that are similar in both domains [14, 23]. Specifically, the approach by [14] tries to induce correspondences between the features in both domains by identifying so-called pivot features that appear frequently in both domains but that behave differently in each domain; singular value decomposition is applied on the resulting pivot features to obtain a low-dimensional, real-valued feature representation that is used to augment the original features. This approach works well for natural language processing problems due to the natural presence of correspondences between features, e.g. words that signal each other.

The approach of [14] is related to many of the instantiations of FLDA that we consider, but it is different in the sense that we only use information on differences in feature presence between the source and the target domain to reweigh those features (that is, we do not explicitly augment the feature representation). Moreover, the formulation of FLDA is more general, and can be extended through a relatively large family of transfer models.

4.3. Feature-level domain adaptation

Suppose we wish to train a sentiment classifier for reviews, and we have a data set with book reviews and associated sentiment labels (positive or negative review) available. After having trained a linear classifier on word-count representations of the book reviews, we wish to deploy it to predict the sentiment of kitchen appliance reviews. This leaves us with a domain-adaptation problem on which the classifier trained on book reviews will

likely not work very well: the classifier's parameters will be very large for, for instance, words such as "interesting" and "insightful", because these suggest positive book reviews. But these words hardly ever appear in reviews of kitchen appliances. As a result, a classifier trained naively on the book reviews may perform poorly on kitchen appliance reviews. Since the target domain data (the kitchen appliance reviews) are available at training time, a natural approach to resolving this problem may be to down-weigh features corresponding to words that do not appear in the target reviews, for instance, by applying a high level of dropout [24] to the corresponding features in the source data when training the classifier. The use of dropout mimics the target domain scenario in which the "interesting" and "insightful" features are hardly ever observed during the training of the classifier, and prevents that these features are assigned large parameter values. Feature-level domain adaptation FLDA aims to formalize this idea in a two-stage approach that (1) fits a probabilistic sample transformation model that aims to model the transfer between source and target domain and (2) trains a classifier by minimizing the risk of the source data under the transfer model.

In the first stage, FLDA models the transfer between the source and the target domain: the transfer model is a data-dependent distribution that models the likelihood of target data conditioned on observed source data. Examples of such transfer models may be a dropout distribution that assigns a likelihood of $1-\zeta$ to the observed feature value in the source data and a likelihood of ζ to a feature value of 0, or a Parzen density estimator in which the mean of each kernel is shifted by a particular value. The parameters of the transfer distribution are learned by maximizing the likelihood of target data under the transfer distribution (conditioned on the source data). In the second stage, we train a linear classifier to minimize the expected value of a classification loss under the transfer distribution. For quadratic and exponential loss functions, this expected value and its gradient can be analytically derived whenever the transfer distribution factorizes over features and is in the natural exponential family; for logistic and hinge losses, practical upper bounds and approximations can be derived [25–27].

In the experimental evaluation of FLDA, we focus on applying dropout transfer models to domain-adaptation problems involving binary and count features. These features frequently appear in, for instance, bag-of-words features in natural language processing [28] or bag-of-visual-words features in computer vision [29]. However, we note that FLDA can be used in combination with a larger family of transfer models; in particular, the expected loss that is minimized in the second stage of FLDA can be computed or approximated efficiently for any transfer model that factorizes over variables and that is in the natural exponential family.

4.3.1. Notation

Consider an input space $\mathcal{X}$, part of a D-dimensional vector space, and a set of classes $\mathcal{Y} = \{-1, +1\}$. A source domain is a joint distribution defined over these spaces, $(\mathcal{X}, \mathcal{Y}, p_{\mathcal{S},\mathcal{Y}})$, marked with the subscript $\mathcal{S}$ and a target domain is another $(\mathcal{X}, \mathcal{Y}, p_{\mathcal{T},\mathcal{Y}})$, marked with $\mathcal{T}$. Samples from the source domain are denoted as the pair (x, y), with n samples forming the source dataset $\mathcal{D}_{\mathcal{S}}^{n} = \{(x_i, y_i)\}_{i=1}^{n}$. Similarly, target samples are denoted as (z, u) with

m samples forming the target dataset $\mathcal{D}_{\mathcal{T}}^{m} = \{(z_j, u_j)\}_{j=1}^{m}$.

4.3.2. Target risk

We adopt the empirical risk minimization (ERM) framework for constructing our domain-adapted classifier. The ERM framework proposes a classification function $h : \mathbb{R}^D \to \mathbb{R}$ and assesses the quality of the hypothesis by comparing its predictions with the true labels on the empirical data using a loss function $\ell : \mathbb{R} \times \mathcal{Y} \to \mathbb{R}^+$. The empirical loss is an estimate of the *risk*, which is defined as the expected value of the loss function under the data distribution. Below, we show that if the target domain carries no additional information about the label distribution, the risk of a model on the target domain is equivalent to the risk on the source domain under a particular transfer distribution.

We first note that the joint source data, target data and label distribution can be decomposed into two conditional distributions and one marginal source distribution; $p_{\mathcal{Y},\mathcal{T},\mathcal{S}} = p_{\mathcal{Y}|\mathcal{T},\mathcal{S}}\ p_{\mathcal{T}|\mathcal{S}}\ p_{\mathcal{S}}$. The first conditional $p_{\mathcal{Y}|\mathcal{T},\mathcal{S}}$ describes the full class-posterior distribution given both source and target distribution. Next, we introduce our main assumption: the labels are conditionally independent of the target domain given the source domain ($\mathcal{Y} \perp\!\!\!\perp \mathcal{T} \mid \mathcal{S}$), which implies: $p_{\mathcal{Y}|\mathcal{T},\mathcal{S}} = p_{\mathcal{Y}|\mathcal{S}}$. In other words, we assume that we can construct an optimal target classifier if (1) we have access to infinitely many labeled source samples—we know $p_{\mathcal{Y}|\mathcal{S}}\ p_{\mathcal{S}}$—and (2) we know the true domain transfer distribution $p_{\mathcal{T}|\mathcal{S}}$. In this scenario, observing target labels does not provide new information.

To illustrate our assumption, imagine a sentiment classification problem. If people frequently use the word "nice" in positive reviews about electronics products (the source domain) and we know that electronics and kitchen products (the target domain) are very similar, then we assume that the word "nice" is not predictive of negative reviews of kitchen appliances. In other words, knowing that "nice" is predictive of a positive review and knowing that the domains are similar, it cannot be the case that "nice" is suddenly predictive of a negative review. Under this assumption, learning a good model for the target domain amounts to transferring the source domain to the target domain (that is, altering the marginal probability of observing the word "nice") and learning a good predictive model on the resulting transferred source domain.

Admittedly, there are scenarios in which our assumption is invalid: if people like "small" electronics but dislike "small" cars, the assumption is violated and our domain-adaptation approach will likely not work well. We do note, however, that our assumption is less stringent than the covariate-shift assumption, which assumes that the posterior distribution over classes is identical in the source and the target domain (i.e. that $p_{\mathcal{Y}|\mathcal{S}} = p_{\mathcal{Y}|\mathcal{T}}$). The covariate-shift assumption does not facilitate the use of a transfer distribution $p_{\mathcal{T}|\mathcal{S}}$.

We start by rewriting the risk of the target domain $R_{\mathcal{T}}$ as follows:

$$\begin{aligned} R_{\mathcal{T}}(h) &= \int_{\mathcal{X}} \sum_{y \in \mathcal{Y}} \ell(h(z), y)\, p_{y,\mathcal{T}}(y, z)\, \mathrm{d}z \\ &= \int_{\mathcal{X}} \sum_{y \in \mathcal{Y}} \int_{\mathcal{X}} \ell(h(z), y)\, p_{y,\mathcal{T},\mathcal{S}}(y, z, x)\, \mathrm{d}x\, \mathrm{d}z \\ &= \int_{\mathcal{X}} \sum_{y \in \mathcal{Y}} \int_{\mathcal{X}} \ell(h(z), y)\, p_{y \mid \mathcal{T},\mathcal{S}}(y \mid z, x)\, p_{\mathcal{T} \mid \mathcal{S}}(z \mid x)\, p_{\mathcal{S}}(x)\, \mathrm{d}x\, \mathrm{d}z\,. \end{aligned}$$

Using the assumption $p_{y \mid \mathcal{T},\mathcal{S}} = p_{y \mid \mathcal{S}}$ (or equivalently, $\mathcal{Y} \perp\!\!\!\perp \mathcal{T} \mid \mathcal{S}$) as introduced above, we can rewrite this expression as:

$$\begin{aligned} R_{\mathcal{T}}(h) &= \int_{\mathcal{X}} \sum_{y \in \mathcal{Y}} \int_{\mathcal{X}} \ell(h(z), y)\, p_{y \mid \mathcal{S}}(y \mid x)\, p_{\mathcal{T} \mid \mathcal{S}}(z \mid x)\, p_{\mathcal{S}}(x)\, \mathrm{d}x\, \mathrm{d}z \\ &= \int_{\mathcal{X}} \mathbb{E}_{y,\mathcal{S}}\big[\ell(h(z), y)\, p_{\mathcal{T} \mid \mathcal{S}}(z \mid x)\big]\, \mathrm{d}z\,. \end{aligned}$$

Next, we replace the target risk with its empirical estimate by plugging in source data $\mathcal{D}_{\mathcal{S}}$ for the source joint distribution $p_{y,\mathcal{S}}$:

$$\begin{aligned} \hat{R}_{\mathcal{T}}(h \mid \mathcal{D}_{\mathcal{S}}) &= \frac{1}{n} \int_{\mathcal{X}} \sum_{i=1}^{n} \ell(h(z), y_i)\, p_{\mathcal{T} \mid \mathcal{S}}(z \mid x = x_i)\, \mathrm{d}z \\ &= \frac{1}{n} \sum_{i=1}^{n} \mathbb{E}_{\mathcal{T} \mid \mathcal{S} = x_i}\left[\, \ell(h(z), y_i) \,\right]. \end{aligned} \tag{4.1}$$

Feature-level domain adaptation (FLDA) trains classifiers by constructing a parametric model of the transfer distribution $p_{\mathcal{T} \mid \mathcal{S}}$ and, subsequently, minimizing the expected loss in Equation 4.1 *on the source data* with respect to the parameters of the classifier. For linear classifiers, the expected loss in Equation 4.1 can be computed analytically for quadratic and exponential losses if the transfer distribution factorizes over dimensions and is in the natural exponential family; for the logistic and hinge losses, it can be upper-bounded or approximated efficiently under the same assumptions [25–27]. Note that no observed target samples z_j are involved Equation 4.1; the expectation is over the transfer model $p_{\mathcal{T} \mid \mathcal{S}}$, conditioned on a particular sample x_i. The target data is only used to estimate the parameters of the transfer model.

4.3.3. Transfer model

The transfer distribution $p_{\mathcal{T} \mid \mathcal{S}}$ describes the relation between the source and the target domain: given a particular source sample, it produces a distribution of which target samples are likely to be observed (with the same label). The transfer distribution is modeled by selecting a parametric distribution and learning the parameters of this distribution from the

source and target data (without looking at the source labels). Prior knowledge on the relation between source and target domain may be incorporated in the model via the choice for a particular family of distributions. For instance, if we know that the main variation between two domains consists of particular words that are frequently used in one domain (say, news articles) but infrequently in another domain (say, tweets), then we choose a distribution that alters the relative frequency of words.

Given a model of the transfer distribution $p_{\mathcal{T}|\mathcal{S}}$ and a model of the source distribution $p_{\mathcal{S}}$, we can work out the marginal distribution over the target domain as

$$q_{\mathcal{T}}(z \mid \zeta, \eta) = \int_{\mathcal{X}} p_{\mathcal{T}|\mathcal{S}}(z \mid x, \zeta)\, p_{\mathcal{S}}(x \mid \eta)\, \mathrm{d}x\,, \tag{4.2}$$

where ζ represents the parameters of the transfer model, and η the parameters of the source model. We learn these parameters separately: first, we learn η by maximizing the likelihood of the source data under the model $p_{\mathcal{S}}(x \mid \eta)$ and, subsequently, we learn ζ by maximizing the likelihood of the target data under the compound model $q_{\mathcal{T}}(z \mid \zeta, \eta)$. Hence, we first estimate the value of η by solving:

$$\hat{\eta} = \arg\max_{\eta} \sum_{i=1}^{n} \log p_{\mathcal{S}}(x_i \mid \eta)\,.$$

Subsequently, we estimate the value of ζ by solving:

$$\hat{\zeta} = \arg\max_{\zeta} \sum_{j=1}^{m} \log q_{\mathcal{T}}(z_j \mid \zeta, \hat{\eta})\,. \tag{4.3}$$

For the moment, we focus on domain-adaptation problems involving binary and count features. In such problems, we wish to encode changes in the marginal likelihood of observing non-zero values in the transfer model. To this end, we employ a dropout distribution as transfer model that can model domain-shifts in which a feature occurs less often in the target domain than in the source domain. Learning a FLDA model with a dropout transfer model has the effect of strongly regularizing classifier parameters for features that occur infrequently in the target domain.

Dropout transfer

To define our transfer model for binary or count features, we first set up a model that describes the likelihood of observing non-zero features in the source data. This model comprises a product of independent Bernoulli distributions:

$$p_{\mathcal{S}}(x_i \mid \eta) = \prod_{d=1}^{D} (1 - \eta_d)^{x_{id}=0}\, \eta_d^{x_{id}\neq 0}\,, \tag{4.4}$$

where d indicates the d-th feature. In this case, η_d corresponds to the probability of a non-zero value for the d-th feature; $x_d \neq 0$. For a Bernoulli distribution, the maximum likelihood estimate of η_d is the sample average: $\hat{\eta}_d = 1/n \sum_{i=1}^{n} [x_{id} \neq 0]$.

Next, we define a transfer model that describes how often a feature has a value of zero in the target domain when it has a non-zero value in the source domain. We assume an unbiased dropout distribution [26, 30] that sets an observed feature in the source domain to zero in the target domain with probability ζ_d:

$$p_{\mathcal{T}|\mathcal{S}}(z_d \mid x = x_{id}, \zeta_d) = \begin{cases} \zeta_d & \text{if } z_d = 0 \\ 1 - \zeta_d & \text{if } z_d = x_{id} \,/(1 - \zeta_d) \end{cases}, \tag{4.5}$$

where $\forall d : 0 \leq \zeta_d \leq 1$. The outcome of not *dropping out* is modeled as $x_{id}/(1 - \zeta_d)$ in order to ensure that the transfer model centers on the particular source sample:

$$\begin{aligned} \mathbb{E}_{\mathcal{T}|\mathcal{S}}[z_d] &= \zeta_d 0 + (1 - \zeta_d)\frac{x_i d}{1 - \zeta_d} \\ &= x_{id}\,. \end{aligned}$$

We assume that features are independent, which means that the joint transfer distribution consists of the product of univariate transfer distributions: $p_{\mathcal{T}|\mathcal{S}}(z \mid x_i, \zeta) = \prod_d^D p_{\mathcal{T}|\mathcal{S}}(z_d \mid x_{id}, \zeta_d)$. Equation 4.5 defines a transfer distribution for a single source sample. We apply this model to each source sample and share the parameters. That ensures that we can average over all source samples x_i to estimate ζ.

To compute the maximum likelihood estimate of ζ, the dropout transfer model from Equation 4.5 and the source model from Equation 4.4 are plugged into Equation 4.2 to obtain (see Appendix A for details):

$$\begin{aligned} q_{\mathcal{T}}(z \mid \zeta, \eta) &= \prod_{d=1}^{D} \int_{x_d} p_{\mathcal{T}|\mathcal{S}}(z_d \mid x_d, \zeta_d)\, p_{\mathcal{S}}(x_d \mid \eta_d)\, \mathrm{d}x_d \\ &= \prod_{d=1}^{D} \Big(1 - (1 - \zeta_d)\,\eta_d\Big)^{z_d = 0} \Big((1 - \zeta_d)\,\eta_d\Big)^{z_d \neq 0}\,. \end{aligned} \tag{4.6}$$

Plugging this expression into Equation 4.3 and maximizing with respect to ζ, we obtain:

$$\hat{\zeta}_d = \max\{\,0, 1 - \frac{1/m \sum_{j=1}^{m} [z_{jd} \neq 0]}{1/n \sum_{i=1}^{n} [x_{id} \neq 0]}\,\},$$

We note that our particular choice for the transfer model cannot represent rate changes, such as whether a word is used on average 10 times in a document versus on average only 3 times. The dropout distribution only captures whether a feature is present or not.

Because our dropout transfer model factorizes over features and is in the natural exponential family, the expectation in Equation 4.1 can be analytically computed. In particular, for a transfer distribution conditioned on source sample x_i, its mean and variance are:

$$\begin{aligned} \mathbb{E}_{\mathcal{T}|x_i}[z] &= x_i \\ \mathbb{V}_{\mathcal{T}|x_i}[z] &= x_i\, \mathrm{diag}\Big(\frac{\zeta}{1 - \zeta}\Big)\, x_i^{\top}\,. \end{aligned}$$

$\mathbb{V}_{\mathcal{T}|x_i}[z]$ is a diagonal matrix because we assumed that features are independently transferred, i.e., the joint transfer model consists of the product of D univariate transfer distributions.

4.3.4. Classification

In order to perform classification with the risk formulation in Equation 4.1, we first need to select a loss function ℓ. Popular choices for the loss function include the quadratic loss (used in least-squares classification), the exponential loss (used in boosting), the hinge loss (used in support vector machines) and the logistic loss (used in logistic regression). Equation 4.1 has been studied before in the context of *dropout training* for the quadratic, exponential, and logistic loss by [25, 26], and for hinge loss by [27]. For the moment, we use only the quadratic and logistic loss functions, but we note that FLDA can also be used in combination with exponential and hinge losses.

Secondly, we have to select an hypothesis class of classifier functions. We focus on linear classifiers in this paper, but nonlinear extensions using basis functions are possible as well. Linear classifiers project a sample onto a parameter vector, $\theta = (\theta_1, \ldots, \theta_D, \theta_0)$, and make decisions based on which side of its decision boundary it ends up. The classifier function is: $h(z) = \sum_{d=1}^{D} z_d\theta_d + \theta_0$. However, with some abuse of notation, it will be written as $h(z) = z\theta$, with the implicit requirement that z is augmented to $[z\ 1]$.

Quadratic loss

The quadratic loss function punishes the squared deviation between the classifier's prediction and the true label: $\ell(h(z), y) = (h(z) - y)^2$. Using this loss, the expectation in Equation 4.1 can be expressed as:

$$\begin{aligned}\hat{R}_{\mathcal{T}}(h \mid \mathcal{D}_S) =& \frac{1}{n}\sum_{i=1}^{n} \mathbb{E}_{\mathcal{T}|x_i}\left[(y_i - z_j\theta)^2\right] \\ =& \frac{1}{n}\sum_{i=1}^{n} y_i^2 - 2\, y_i \mathbb{E}_{\mathcal{T}|x_i}[z]\theta + \theta\left(\mathbb{E}_{\mathcal{T}|x_i}[z]^\top \mathbb{E}_{\mathcal{T}|x_i}[z] + \mathbb{V}_{\mathcal{T}|x_i}[z]\right)\theta\,,\end{aligned}$$

Minimizing the risk with respect to the classifier's parameters yields the optimal ones. For the quadratic loss, the result is called the *least-squares* classifier. Taking the gradient and setting it to zero yields the following closed-form solution:

$$\hat{\theta} = \Big(\sum_{i=1}^{n} \mathbb{E}_{\mathcal{T}|x_i}[z]^\top \mathbb{E}_{\mathcal{T}|x_i}[z] + \mathbb{V}_{\mathcal{T}|x_i}[z]\Big)^{-1}\Big(\sum_{i=1}^{n} \mathbb{E}_{\mathcal{T}|x_i}[z] y_i\Big)\,. \tag{4.7}$$

For multi-class problems ($\mathcal{Y} = \{1, \ldots, K\}$), multiple predictors can be built in an one-vs-all fashion or in an one-vs-one fashion.

The solution in Equation 4.7 is very similar to the solution of a standard ridge regression model. The main difference is that, in a standard ridge regressor, the regularization

is independent of the data. By contrast, the regularization of FLDA is determined by the variance of the transfer model: hence, it is different for each dimension and it depends on the transfer from source to target domain. Algorithm 1 summarizes the training of a binary quadratic-loss FLDA classifier with dropout transfer.

Algorithm 1 FLDA-q

for d=1 **to** D **do**

$$\zeta_d = \max\Big\{0, 1 - (1/m \textstyle\sum_j^m [z_{jd} \neq 0]) \,/\, (1/n \textstyle\sum_i^n [x_{id} \neq 0])\Big\}$$

end for

$$\hat{\theta} = \big(\textstyle\sum_i^n x_i^\top x_i + x_i^\top \mathrm{diag}(\zeta/(1-\zeta)) x_i\big)^{-1} \big(\textstyle\sum_i^n x_i y_i\big)$$

return $z_j \hat{\theta}$ for $j = 1, \ldots, m$

4

Logistic loss

The logistic loss function punishes incorrect predictions heavily but drops off as the prediction improves; $\ell(h(z), y) = -\log[\exp(yh(z))/(\sum_{y' \in \mathcal{Y}} \exp(y' h(z)))]$. The logistic version of FLDA can be expressed as:

$$\begin{aligned} \hat{R}_{\mathcal{T}}(h \mid \mathcal{D}_S) &= \frac{1}{n} \sum_{i=1}^{n} \mathbb{E}_{\mathcal{T}|x_i}\big[- y_i z\theta + \log \sum_{y' \in \mathcal{Y}} \exp(y' z\theta)\big] \\ &= \frac{1}{n} \sum_{i=1}^{n} -y_i \mathbb{E}_{\mathcal{T}|x_i}[z]\theta + \mathbb{E}_{\mathcal{T}|x_i}\big[\log \sum_{y' \in \mathcal{Y}} \exp(y' z\theta)\big]. \end{aligned} \tag{4.8}$$

The expectation is a linear operation and can therefore be applied to both terms of the sum separately. As the expectation does not depend on the classifier parameters, they can be pulled out: $\mathbb{E}_{\mathcal{T}|x_i}[z\theta] = \mathbb{E}_{\mathcal{T}|x_i}[z]\theta$.

Equation 4.8 is a convex function in θ because the original logistic loss function is convex and the expectation operation is convexity-preserving. Hence, the risk function has a global optimum. However, the expectation cannot be computed analytically. Following [26], we approximate the expectation of the log-partition function using a Taylor expansion. To avoid notational overload, we first introduce $A(\cdot) = \log \sum_{y' \in \mathcal{Y}} \exp(y' \cdot)$ as shorthand notation for the log-partition function. Now, $\mathbb{E}_{\mathcal{T}|x_i}\big[A(z\theta)\big]$ is approximated around the

point $x_i\theta$:

$$
\begin{aligned}
\mathbb{E}_{\mathcal{T}|x_i}\big[A(z\theta)\big] \approx\ & \mathbb{E}_{\mathcal{T}|x_i}\big[A(x_i\theta)\big] + \\
& \mathbb{E}_{\mathcal{T}|x_i}\big[\frac{\partial A(x_i\theta)}{\partial x_i\theta}(z\theta - x_i\theta)\big] + \\
& \mathbb{E}_{\mathcal{T}|x_i}\big[\frac{1}{2}\frac{\partial^2 A(x_i\theta)}{\partial (x_i\theta)^2}(z\theta - x_i\theta)^2\big] \\
=\ & A(x_i\theta) + \\
& \frac{\partial A(x_i\theta)}{\partial x_i\theta}(\mathbb{E}_{\mathcal{T}|x_i}[z\theta] - x_i\theta) + \\
& \frac{1}{2}\frac{\partial^2 A(x_i\theta)}{\partial (x_i\theta)^2}\mathbb{E}_{\mathcal{T}|x_i}[(z\theta - x_i\theta)^2] \qquad (4.9) \\
=\ & A(x_i\theta) + \\
& \frac{\partial A(x_i\theta)}{\partial x_i\theta}(\mathbb{E}_{\mathcal{T}|x_i}[z] - x_i)\theta + \\
& \frac{1}{2}\frac{\partial^2 A(x_i\theta)}{\partial (x_i\theta)^2}\theta^\top \mathbb{E}_{\mathcal{T}|x_i}[(z - x_i)^2]\theta \qquad (4.10) \\
=\ & \log \sum_{y' \in y} \exp(y' x_i\theta) + 1/2\big(1 - \beta_i^2/\alpha_i^2\big)\theta^\top \mathbb{V}_{\mathcal{T}|x_i}[z]\theta\,,
\end{aligned}
$$

where $\alpha_i = \sum_{y' \in y} \exp(y' x_i\theta)$ and $\beta_i = \sum_{y' \in y} y' \exp(y' x_i\theta)$. As there is no z in $A(x_i\theta)$, the expectation disappears in the first term on the right-hand side of Equation 4.9. For that same reason, the derivatives $\partial A(x_i\theta)/\partial x_i\theta$ and $\partial^2 A(x_i\theta)/\partial (x_i\theta)^2$ can be pulled out of the expectation. In Equation 4.10, θ is pulled out of the quadratic term: $(z\theta - x\theta)^2 = ((z - x)\theta)^2 = \theta^\top (z - x)^2\theta$. As $\mathbb{E}_{\mathcal{T}|x_i}[z] = x_i$, $\mathbb{E}_{\mathcal{T}|x_i}[(z - x_i)^2]$ corresponds to the variance of the transfer model $\mathbb{V}_{\mathcal{T}|x_i}[z]$. Furthermore, note that the whole first-order term of the approximation disappears if an unbiased transfer model is used, $\mathbb{E}_{\mathcal{T}|x_i}[z] = x_i$, which makes $(\mathbb{E}_{\mathcal{T}|x_i}[z] - x_i)$ equal to 0.

Unfortunately, there is no closed-form solution to the minimization of the risk approximation with respect to the classifiers parameters. In order to find them, we perform gradient descent. The derivation of the gradient can be found in Appendix B. Algorithm 2 presents pseudocode for FLDA with the logistic loss and a dropout transfer distribution.

4.4. Experiments

In our experiments, we first study the empirical behavior of FLDA on artificial data for which we know the true transfer distribution. Following that, we measure the performance of our method in a "missing data at test time" scenario, as well as on two image data sets and three text data sets with varying amounts of domain transfer.

Algorithm 2 FLDA-I

for d=1 **to** D **do**

$$\zeta_d = \max\Big\{0, 1 - (1/m \textstyle\sum_j^m [z_{jd} \neq 0]) \,/\, (1/n \textstyle\sum_i^n [x_{id} \neq 0])\Big\}$$

end for

$$\hat{\theta} = \arg\min_{\theta \in \Theta} \; 1/n \textstyle\sum_i^n \Big[- y_i x_i \theta + \log \sum_{y' \in \mathcal{Y}} \exp(y' x_i \theta) + 1/2\big(1 - \big[\sum_{y' \in \mathcal{Y}} \frac{y' \exp(y' x_i \theta)}{\sum_{y'' \in \mathcal{Y}} \exp(y'' x_i \theta)}\big]^2\big)\, \theta^\top \big(x_i \text{diag}\big(\frac{\zeta}{1-\zeta}\big) x_i^\top\big)\theta\Big]$$

return $z_j \hat{\theta}$ for $j = 1, \ldots, m$

4

4.4.1. Artificial data

We first investigate the behavior of FLDA on a problem in which the model assumptions are satisfied. We create such a problem setting by first sampling a source domain data set from known class-conditional distributions. Subsequently, we construct a target domain data set by sampling additional source data and transforming it using a pre-defined (dropout) transfer model.

Adaptation under correct model assumptions

We perform experiments in which the domain-adapted classifier estimates the transfer model and trains on the source data; we evaluate the quality of the resulting classifier by comparing it to an oracle classifier that was trained on the target data (that is, the classifier one would train if labels for the target data were available at training time).

In the first experiment, we generate binary features by drawing $100,000$ samples from two bivariate Bernoulli distributions. The marginal distributions are $[0.7,\ 0.7]$ for class one and $[0.3\ \ 0.3]$ for class two. The source data is transformed to the target data using a dropout transfer model with parameters $\zeta = [0.5,\ 0]$. This means that 50% of the values for feature 1 are set to 0 and the other values are scaled by $1/(1-0.5)$. For reference, two naive least-squares classifiers are trained, one on the labeled source data (s-ls) and one on the labeled target data (t-ls), and compared to FLDA-q. s-ls achieves a misclassification error of 0.40 while t-ls and FLDA-q achieve an error of 0.30. This experiment is repeated for the same classifiers but with logistic losses: a source logistic classifier (S-LR), a target logistic classifier (T-LR) and FLDA-l. In this experiment, S-LR again achieves an error of 0.40 and T-LR and FLDA-l an error of 0.30. Figure 4.1 shows the decision boundaries for the quadratic loss classifiers on the left and the logistic loss classifiers on the right. The figure shows that for both loss functions, FLDA has completely adapted to be equivalent to the target classifier in this artificial problem.

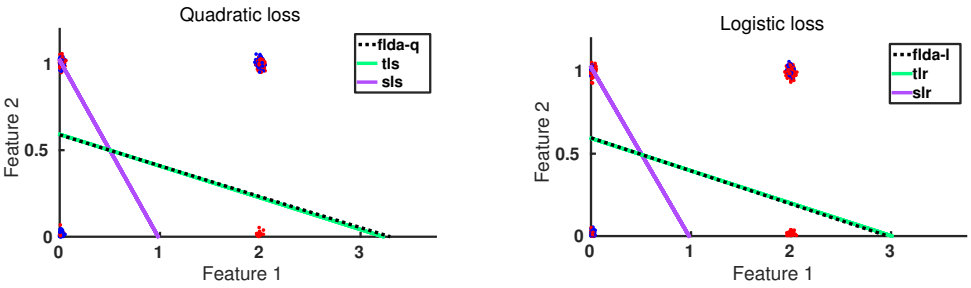


Figure 4.1: Scatter plots of the target domain. The data was generated by sampling from bivariate Bernoulli class-conditional distributions and transformed using a dropout transfer. Red and blue dots show different classes. The lines are the decision boundaries found by the source classifier (s-ls/S-LR), the target classifier (t-ls/T-LR) and the adapted classifier (left FLDA-q, right FLDA-l). Note that the decision boundary of FLDA lies on top of the decision boundary of t-ls/T-LR.

4

In the second experiment, we generate count features by sampling from bivariate Poisson distributions. Herein, we used rate parameters $\left[2,\ 2\right]$ for the first class and $\left[6,\ 6\right]$ for the second class. Again, we construct the target domain data by generating new samples and dropping out the values of feature 1 with a probability of 0.5. In this experiment s-ls achieves an error of 0.181 and t-ls / FLDA-q achieve an error of 0.099, while S-LR achieves an error of 0.170 and T-LR / FLDA-l achieve an error of 0.084. Figure 4.2 shows the decision boundaries of each of these classifiers and that FLDA has fully adapted to the domain shift.

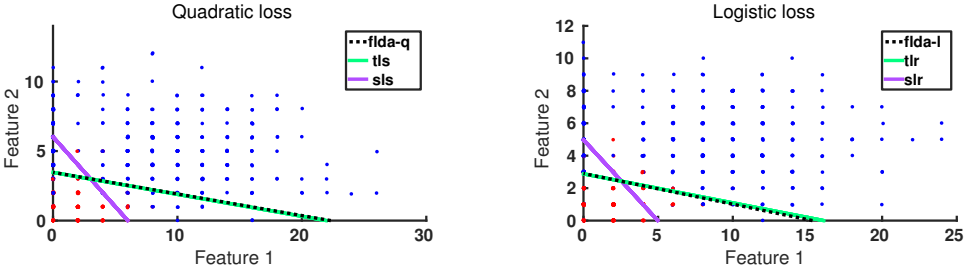


Figure 4.2: Scatter plots of the target domain with decision boundaries of classifiers. The data was generated by sampling from bivariate Poisson class-conditional distributions and the decision boundaries were constructed using the source classifier (s-ls/S-LR), the target classifier (t-ls/T-LR), and the adapted classifiers (left FLDA-q, right FLDA-l). Note that the decision boundary of FLDA lies on top of the decision boundary of t-ls / T-LR.

Learning curves

One question that arises from the previous experiments is how many samples FLDA needs to estimate the transfer parameters and adapt to be (nearly) identical to the target classifier. To answer it, we performed an experiment in which we computed the classification error rate as a function of the number of training samples. The source training and validation data was generated from the same bivariate Poisson distributions as in Figure 4.2. The target data was constructed by generating additional source data and dropping out the first feature with a probability of 0.5. Each of the four data sets contained $10,000$ samples. First, we trained a naive least-squares classifier on the source data (s-ls) and tested its per-

formance on both the source validation and the target sets as a function of the number of source training samples. Second, we trained a naive least-squares classifier on the labeled target data (t-ls) and tested it on the source validation and another target validation set as a function of the number target training samples. Third, we trained an adapted classifier (FLDA-q) on equal amounts of labeled source training data and unlabeled target training data and tested it on both the source validation and target validation sets. The experiment was repeated 50 times for every sample size to calculate the standard error of the mean.

The learning curves are plotted in Figure 4.3, which shows the classification error on the source validation set (top) and the classification error on the target validation (bottom). As expected, the source classifier (s-ls) outperforms the target (t-ls) and adapted (FLDA-q) classifiers on the source domain (dotted lines), while FLDA-q and t-ls outperform s-ls on the target domain (solid lines). In this problem, it appears that roughly 20 labeled source samples and 20 unlabeled target samples are sufficient for FLDA to adapt to the domain shift. Interestingly, FLDA-q is outperforming s-ls and t-ls for small sample sizes. This is most likely due to the fact that the application of the transfer model is acting as a kind of regularization. In particular, when the learning curves are computed with L^2-regularized classifiers, then the difference in performance disappears.

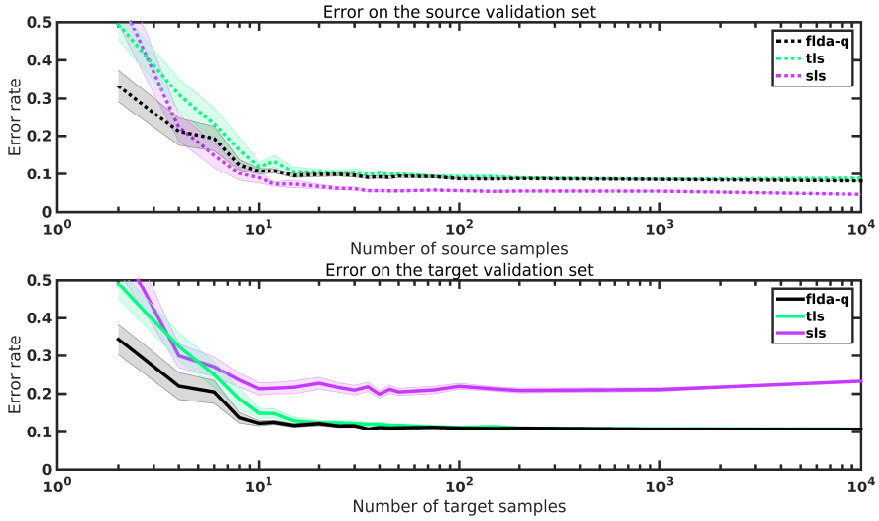


Figure 4.3: Learning curves of the source classifier (s-ls), the target classifier (t-ls), and adapted classifier (FLDA-q). The top figure shows the error on a validation set generated from two bivariate Poisson distributions. The bottom figure shows the error on a validation set generated from two bivariate Poisson distributions with the first feature dropped out with a probability of 0.5.

Robustness to transfer model parameter estimation errors

Another question that arises is how sensitive the approach is to estimation errors in the transfer parameters. To answer this question, we performed an experiment in which we artificially introduce an error in the transfer parameters by perturbing them. As before, we generate $100,000$ samples for both domains by sampling from bivariate Poisson dis-

tributions with rates $[2,\ 2]$ for class 1 and $[6,\ 6]$ for class 2. Again, the target domain is constructed by dropping out feature 1 with a probability of 0.5. We trained a naive classifier on the source data (s-ls), a naive classifier on the target data (t-ls), and an adapted classifier FLDA-q with four different sets of parameters: the maximum likelihood estimate of the first transfer parameter $\hat{\zeta}_1$ with an addition of 0, 0.1, 0.2, and 0.3. Table 4.1 shows the resulting classification errors, which reveal a relatively small effect of perturbing the estimated transfer parameters: the errors only increase by a few percent in this experiment.

Table 4.1: Classification errors for a naive source classifier, a naive target classifier, and the adapted classifier with a value of 0, 0.1, 0.2, and 0.3 added to the estimate of the first transfer parameter $\hat{\zeta}_1$.

	sl	tl	$\hat{\zeta}_1 + 0$	$\hat{\zeta}_1 + 0.1$	$\hat{\zeta}_1 + 0.2$	$\hat{\zeta}_1 + 0.3$
Quadratic	0.245	0.137	0.138	0.145	0.149	0.150
Logistic	0.264	0.139	0.139	0.140	0.142	0.146

To further illustrate the effect of the transfer parameters, Figure 4.4 shows the decision boundaries for the perturbed adapted classifiers. The figures show that the linear boundaries start to angle upwards when the error in the transfer parameter estimate increases. Overall, one could describe the effect of a dropout transfer model as steering the direction of the linear classifier. This experiment shows the importance of an accurate estimation of the transfer parameters to obtain high-quality adaptation. Nonetheless, our results do suggest that FLDA is robust to relatively small perturbations.

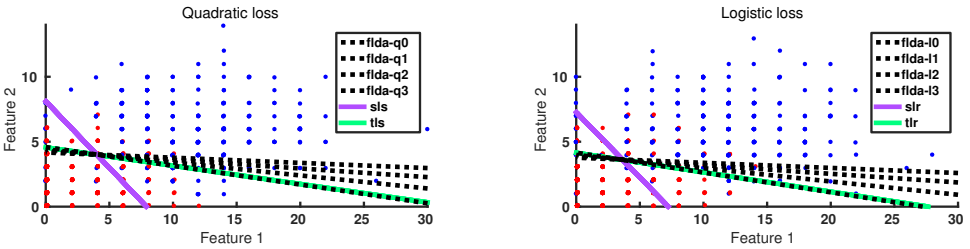


Figure 4.4: Scatter plots of the target data and decision boundaries of two naive and four adapted classifiers with transfer parameter estimate errors of 0, 0.1, 0.2, and 0.3. Results are show for both the quadratic loss classifier (FLDA-q; left) and the logistic loss classifier (FLDA-l; right).

4.4.2. Natural data

In a second set of experiments, we evaluate FLDA on a series of real-world data sets and compare it with several state-of-the-art methods. The evaluations are performed in the transductive learning setting: we measure the performance of the classifier on the already given, but unlabeled target samples.

As baselines, we consider eight alternative methods for domain adaptation. All of these employ a two-stage procedure. In the first step, importance weights, domain-invariant features, or a transformation of the feature space is estimated. In the second step, a classifier

is trained using the results of the first stage. In all experiments, we estimate the hyper-parameters, such as L^2-regularization parameters, via cross-validation on held-out source data. It should be noted that these values are optimal for generalizing to new source data but not necessarily for generalizing to the target domain [31]. Each of the eight baseline methods is described briefly below.

Naive support vector machine (S-SVM)

Our first baseline method is a support vector machine trained on only the source samples and applied on the target samples. We made use of the libsvm package by [32] with a radial basis function kernel and we performed cross-validation to estimate the kernel bandwidth and the L^2-regularization parameter. All multi-class classification is done through an one-vs-one scheme. This method can be readily compared to subspace alignment (SA) and transfer component analysis (TCA) to evaluate the effects of the respective adaptation approaches.

Naive logistic regression (S-LR)

Our second baseline method is an L^2-regularized logistic regressor trained on only the source samples. Its main difference with the support vector machine is that it uses a linear model, a logistic loss instead of a hinge loss, and that it has a natural extension to multi-class as opposed to one-vs-one. The value of the regularization parameter was set via cross-validation. This method can be readily compared to kernel mean matching (KMM), structural correspondence learning (SCL), as well as to the logistic loss version of feature-level domain adaptation (FLDA-l).

Kernel mean matching (KMM)

Kernel mean matching [8, 10] finds importance weights by minimizing the maximum mean discrepancy (MMD) between the reweighed source samples and the target samples. To evaluate the empirical MMD, we used the radial basis function kernel. The weights are then incorporated in an importance-weighted L^2-regularized logistic regressor.

Structural correspondence learning (SCL)

In order to build the domain-invariant subspace [14], the 20 features with the largest proportion of non-zero values in both domains are selected as the pivot features. Their values were dichotomized (1 if $x \neq 0$, 0 if $x = 0$) and predicted using a modified Huber loss [33]. The resulting classifier weight matrix was subjected to an eigenvalue decomposition and the eigenvectors with the 15 largest eigenvalues are retained. The source and target samples are both projected onto this basis and the resulting subspaces are added as features to the original source and target feature spaces, respectively. Consequently, classification is done by training an L^2-regularized logistic regressor on the augmented source samples and testing on the augmented target samples.

Transfer component analysis (TCA)

For transfer component analysis, the closed-form solution to the parametric kernel map described in [16] is computed using a radial basis function kernel. Its hyperparameters (kernel bandwidth, number of retained components and trade-off parameter μ) are estimated through cross-validation. After mapping the data onto the transfer components, we

trained a support vector machine with a radial basis function kernel, cross-validating over its kernel bandwidth and the regularization parameter.

Geodesic flow kernel (GFK)

The geodesic flow kernel is extracted based on the difference in angles between the principal components of the source and target samples [18]. The basis functions of this kernel implicitly map the data onto all possible subspaces on the geodesic path between domains. Classification is performed using a kernel 1-nearest neighbor classifier. We used the subspace disagreement measure (SDM) to select an optimal value for the subspace dimensionality.

Subspace alignment (SA)

For subspace alignment [21], all samples are normalized by their sum and all features are z-scored before extracting principal components. Subsequently, the Frobenius norm between the transformed source components and target components is minimized with respect to an affine transformation matrix. After projecting the source samples onto the transformed source components, a support vector machine with a radial basis function kernel is trained with cross-validated hyperparameters and tested on the target samples mapped onto the target components.

Target logistic regression (T-LR)

Finally, we trained a L^2-regularized logistic regressor using the normally unknown target labels as the oracle solution. This classifier is included to obtain an upper bound on the performance of our classifiers.

Missing data at test time

In this set of experiments, we study "missing data at test time" problems in which we argue that dropout transfer occurs naturally. Suppose that for the purposes of building a classifier, a data set is neatly collected with all features measured for all samples. At test time, however, some features could not be measured, due to for instance sensor failure, and the missing values are replaced by 0. This setting can be interpreted as two distributions over the same space with their transfer characterized by a relative increase in the number of 0 values, which our FLDA with dropout transfer is perfectly suited for. We have collected six data sets from the UCI machine learning repository [34] with missing data: Hepatitis (hepat.), Ozone (ozone; [35]), Heart Disease (heart; [36]), Mammographic masses (mam.; [37]), Automobile (auto), and Arrhythmia (arrhy.; [38]). Table 4.2 shows summary statistics for these sets. In the experiments, we construct the training set (source domain) by selecting all samples with no missing data, with the remainder as the test set (target domain). We note that instead of doing 0-imputation, we also could have used methods such as mean-imputation [39, 40]. It is worth noting that the FLDA framework can adapt to such a setting by simply defining a different transfer model (one that replaces a feature value by its mean instead of a 0).

Table 4.3 reports the classification error rate of all domain-adaptation methods on the before-mentioned data sets. The lowest error rates for a particular data set are bold-faced.

Table 4.2: Summary statistics of the UCI repository data sets with missing data.

	hepat.	ozone	heart	mam.	auto.	arrhy.
Features	19	72	13	4	24	279
Samples	155	2534	704	961	205	452
Classes	2	2	2	2	6	13
Missing	75	685	615	130	72	384

Table 4.3: Classification error rates on 6 UCI data sets with missing data. The data sets were partitioned into a training set (source domain), containing all samples with no missing features, and a test set (target domain), containing all samples with missing features.

	S-SVM	S-LR	KMM	SCL	SA	GFK	TCA	FLDA-q	FLDA-l	T-LR
hepat.	.213	.493	.347	.480	.253	.227	.213	.227	**.200**	.150
ozone	.060	.124	.126	.136	**.047**	.093	.140	**.047**	.079	.069
heart	.409	.338	.390	.319	.596	.362	.391	**.203**	**.203**	.177
mam.	.331	.462	.446	.462	**.323**	.423	**.323**	.462	.431	.194
auto.	.848	.935	.913	.935	.587	**.565**	.848	.848	.848	.371
arrhy.	.930	.854	.620	.818	**.414**	.651	.930	.456	.889	.353

From the results presented in the table, we observe that whilst there appears to be little difference between the domains in the Hepatitis and Ozone data sets, there is substantial domain shift in the other data sets: the naive classifiers even perform at chance level on the Arrhythmia and Automobile data sets. On almost all data sets, both FLDA-q and FLDA-l improve substantially over the S-LR, which suggests that they are successfully adapting to the missing data at test time. By contrast, most of the other domain-adaptation techniques do not consistently improve although, admittedly, sample transformation methods appear to work reasonable well on the Ozone, Mammography, and Arrhythmia data sets.

Handwritten digits

Handwritten digit data sets have been popular in machine learning due to the large sample size and the interpretability of the images. Generally, the data is acquired by assigning an integer value between 0 and 255 proportional to the amount of pressure that is applied at a particular spatial location on an electronic writing pad. Therefore, the probability of a non-zero value of a pixel informs us how often a pixel is part of a particular digit. For instance, the middle pixel in the digit 8 is a very important part of the digit because it nearly always corresponds to a high-pressure location, but the upper-left corner pixel is not used that often and is less important. Domain shift may be present between digit data sets due to differences in recording conditions. As a result, we may observe pixels that are discriminative in one data set (the source domain) that are hardly ever observed in another data set (the target domain). These pixels cannot be used to classify digits in the target domain, and we would like to inform the classifier that it should not assign a large weight to such pixels.

We created a domain adaptation setting by considering two handwritten digit sets, namely MNIST [41] and USPS [42]. In order to create a common feature space, images from both data sets are resized to 16 by 16 pixels. To reduce the discrepancy between the size of MNIST data set (which contains $70,000$ examples) and the USPS data set (which contains $9,298$ examples), we only use $14,000$ samples from the MNIST data set. The classes are balanced in both data sets.

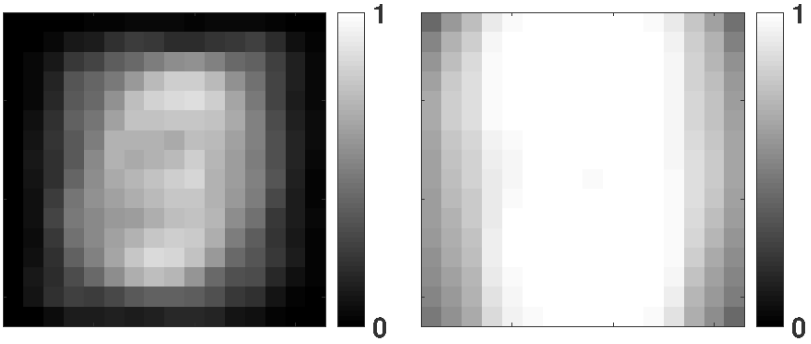


Figure 4.5: Visualization of the probability of non-zero values for each pixel on the MNIST data set (left) and the USPS data set (right).

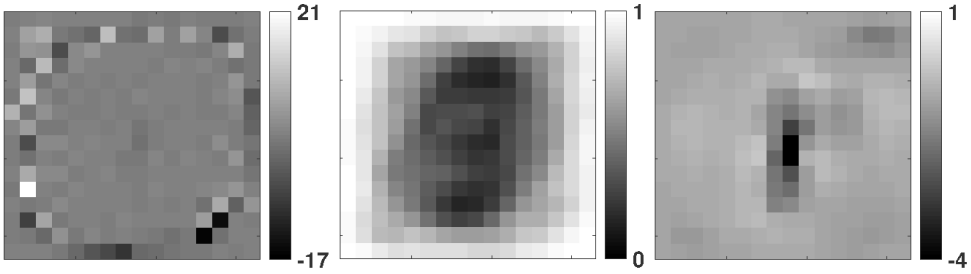


Figure 4.6: Classifier parameter values assigned by the naive source classifier to the 0-digit predictor (left), the transfer model parameters of the dropout transfer model (middle), and the classifier parameter values assigned by the adapted classifier to the 0-digit predictor for training on USPS images and testing on MNIST (right; $U \rightarrow M$).

Figure 4.5 shows a visualization of the probability that each pixel is non-zero for both data sets. The visualization shows that while the digits in the MNIST data set occupy mostly the center region, the USPS digits tend to occupy a substantially larger part of the image, specifically a center column. Figure 4.6 (left) visualizes the classifier parameter values of the naive linear classifier (S-LR), (middle) the dropout probabilities ζ, and (right) the adapted classifier's parameter values (FLDA-I). The middle image shows that dropout probabilities are large in regions where USPS pixels are frequent (the white pixels in Figure 4.5 right) but MNIST pixels are infrequent (the black pixels in Figure 4.5, left). The parameter values of the naive classifier appear to be shaped in a somewhat noisy circular pattern in the periphery, with the center containing negative values (if these center pixels have a low intensity in a new sample, then the image is more likely to be a 0 digit). By contrast, the

parameter values of the FLDA classifier are smoothed in the periphery, which indicates that the classifier is placing more value on the center pixels and is ignoring the peripheral ones.

Table 4.4 shows the classification error rates where the rows correspond to the combinations of treating one data set as the source domain and the other as the target. The results show that there is a large difference between the domain-specific classifiers (S-LR and T-LR), which indicates that the domains are highly dissimilar. We note that the error rates of the target classifier on the MNIST data set are higher than usual for this data set (T-LR has an error rate of 0.234), which is because of the down-sampling of the images to 16×16 pixels and the smaller sample size. The results presented in the table highlight an interesting property of FLDA with dropout transfer: while FLDA performs well in settings in which the domain transfer can be appropriately modeled by the transfer distribution (U→M setting where pixels that appear in the USPS do not appear in MNIST), it does not perform well the other way around. The dropout transfer model does not capture pixels appearing *more often* instead of less often in the target domain. To work well in that setting, it is presumably necessary to use a richer transfer model.

Table 4.4: Classification error rates obtained on both combinations of treating one domain as the source and the other as the target. M='MNIST' and U='USPS'.

	S-SVM	S-LR	KMM	SCL	SA	GFK	TCA	FLDA-q	FLDA-l	T-LR
M→U	.522	.747	.748	.747	.890	**.497**	.808	.811	.678	.055
U→M	.766	.770	769	.808	.757	.660	.857	**.640**	.684	.234

Office-Caltech

The Office-Caltech data set [43] consists of images of objects gathered using four different methods: one from images found through a web image search (referred to as 'C'), one from images of products on Amazon (A), one taken with a digital SLR camera (D) and one taken with a webcam (W). Overall, the set contains 10 classes, with 1123 samples from Caltech, 958 samples from Amazon, 157 samples from the DSLR camera, and 295 samples from the webcam. Our first experiment with the Office-Caltech data set is based on features extracted through SURF features [44]. These descriptors determine a set of interest points by finding local maxima in the determinant of the image Hessian. Weighted sums of Haar features are computed in multiple sub-windows at various scales around each of the interest points. The resulting descriptors are vector-quantized to produce a bag-of-visual-words histogram of the image that is both scale and rotation-invariant. We perform domain-adaptation experiments by training on one domain and testing on another.

Table 4.5 shows the results of the classification experiments, where compared to competing methods, SA is performing well for a number of domain pairs, which may indicate that the SURF descriptor representation leads to domain dissimilarities that can be accurately captured by subspace transformations. This result is further supported by the fact that the transformations found by GFK and TCA are also outperforming S-SVM. FLDA-q and FLDA-l are among the best performers on certain domain pairs. In general, FLDA does appear to perform at least as good or better than a naive S-LR classifier.The results on the

Table 4.5: Classification error rates obtained by ten (domain-adapted) classifiers for all pairwise combinations of domains on the Office-Caltech data set with SURF features (A='Amazon', D='DSLR', W='Webcam', and C='Caltech').

	S-SVM	S-LR	KMM	SCL	SA	GFK	TCA	FLDA-q	FLDA-l	T-LR
A→D	**.599**	.618	.616	.621	.627	.624	.624	**.599**	.624	.303
A→W	.688	.675	.668	.686	**.606**	.631	.712	.648	.678	.181
A→C	.557	.553	.563	.555	.594	.614	.579	.565	**.550**	.427
D→W	.312	.312	.346	.317	.167	**.153**	.295	.322	.312	.181
D→C	.744	.712	.734	.712	**.655**	.706	.680	.712	.710	.427
W→C	.721	.698	.709	.705	.677	.697	.688	**.675**	.701	.427
D→A	.876	.719	.727	.724	**.616**	.680	.650	.700	.722	.258
W→A	.676	.695	.706	.707	**.631**	.665	.668	.671	.691	.258
C→A	.493	.523	.515	.496	.538	.592	.504	.490	**.475**	.258
W→D	.198	.191	.178	.198	.214	**.121**	.166	.191	.185	.303
C→D	.612	.616	.631	.583	.575	.599	.612	**.510**	.599	.303
C→W	.712	.725	.729	.724	**.600**	.603	.695	.654	.702	.181

Office-Caltech data set depend on the type of information the SURF descriptors are extracting from the images. We also studied the performance of domain-adaptation methods on a richer visual representation, produced by a pre-trained convolutional neural network. Specifically, we used a data set provided by [45], who extracted 1000-dimensional feature-layer activations (so-called DeCAF$_8$ features) in the upper layers of the a convolutional network that was pre-trained on the Imagenet data set. Donahue *et al.* [45] used a larger superset of the Office-Caltech data set that contains 31 classes with 2817 images from Amazon, 498 from the DSLR camera, and 795 from the webcam. The results of our experiments with the DeCAF$_8$ features are presented in Table 4.6. The results show substantially lower error rates overall, but they also show that domain transfer in the the DeCAF$_8$ feature representation is not amenable to effective modeling by subspace transformations. KMM and SCL obtain performances that are similar to the of the naive S-LR classifier but in one experiment, the naive classifier is actually the best-performing model. Whilst achieving the best performance on 2 out of 6 domain pairs, the FLDA-q and FLDA-l models are not as effective as on other data sets, presumably, because dropout is not a good model for the transfer in a continuous feature space such as the DeCAF$_8$ feature space.

Table 4.6: Classification error rates obtained by ten (domain-adapted) classifiers for all pairwise combinations of domains on the Office data set with DeCAF$_8$ features (A='Amazon', D='DSLR', and W='Webcam').

	S-SVM	S-LR	KMM	SCL	SA	GFK	TCA	FLDA-q	FLDA-l	T-LR
A→D	.406	.388	.402	.422	.460	.424	**.351**	.428	.388	.104
A→W	.434	.468	.455	.474	.499	.477	**.426**	.491	.468	.064
D→W	.086	.079	.083	.074	.103	**.073**	.087	.088	.079	.064
D→A	.516	.496	.502	.497	.520	.569	.489	.589	**.487**	.216
W→A	.520	**.496**	.514	.506	.541	.584	.510	.645	.501	.216
W→D	.034	.030	.032	.034	.062	.052	.042	**.024**	.044	.104

IMDb

The Internet Movie Database (IMDb) [46] contains written reviews of movies labeled with a 1-10 star rating, which we dichotomize by setting values > 5 as $+1$ and values ≤ 5 as -1. Using this dichotomy, both classes are roughly balanced. From the original bag-of-words representation, we selected only the features with more than 100 non-zero values in the entire data set, resulting in 4180 features. To obtain the domains, we split the data set by genre and obtained 3402 reviews of action movies, 1249 reviews of family movies, and 3697 reviews of war movies. We assume that people tend to use different words to review different genres of movies, and we are interested in predicting viewer sentiment after adapting to changes in the word frequencies. To visualize whether this assumption is valid, we plot the proportion of non-zero values of 10 randomly chosen words per domain in Figure 4.7. The figure suggests that action movie and war movie reviews are quite similar, but the word use in family movie reviews does appear to be different.

Table 4.7 reports the results of the classification experiments on the IMDb database. The first thing to note is that the performances of S-LR and T-LR are quite similar, which suggests that the frequencies of discriminative words do not vary too much between genres. The results also show that GFK and TCA are not as effective on this data set as they were on the handwritten digits and Office-Caltech data sets, which suggests that finding a joint subspace that is still discriminative is hard, presumably, because only a small number of the 4180 words actually carry discriminative information. FLDA-q and FLDA-l are better suited for such a scenario, which is reflected by their competitive performance on all domain pairs.

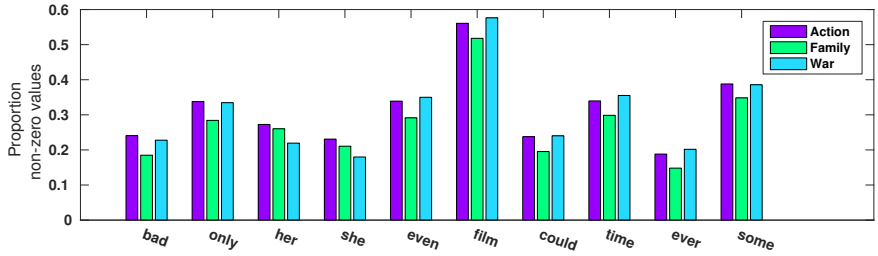


Figure 4.7: Proportion of non-zero values for a subset of words per domain on the IMDb data set.

Spam

Domain adaptation settings also arise in spam detection systems. For this experiment, we concatenated two data sets from the UCI machine learning repository: one containing 4205 emails from the Enron spam database [47] and one containing 5338 text messages from the SMS-spam data set [48]. Both were represented using bag-of-words vectors over 4272 words that occurred in both data sets. Figure 4.8 shows the proportions of non-zero values for some example words, and shows that there exist large differences in word frequencies between the two domains. In particular, much of the domain differences are due to text messages using shortened words, whereas email messages tend to be more formal.

Table 4.7: Classification error rates obtained by ten (domain-adapted) classifiers for all pairwise combinations of domains on the IMDb data set. (A='Action', F='Family', and W='War').

	S-SVM	S-LR	KMM	SCL	SA	GFK	TCA	FLDA-q	FLDA-l	T-LR
A→F	.145	.136	**.133**	**.133**	.184	.276	.230	.135	.136	.196
A→W	.158	.155	.155	.165	.163	.249	.266	.158	**.154**	.163
F→W	.256	.206	.208	.206	.182	.289	.355	.205	**.202**	.163
F→A	.201	.195	**.193**	.198	**.193**	.296	.363	.194	.194	.169
W→A	.168	.160	.159	.159	.167	.238	.222	**.155**	.157	.169
W→F	.340	.167	.163	.163	.232	.292	.203	.172	**.159**	.196

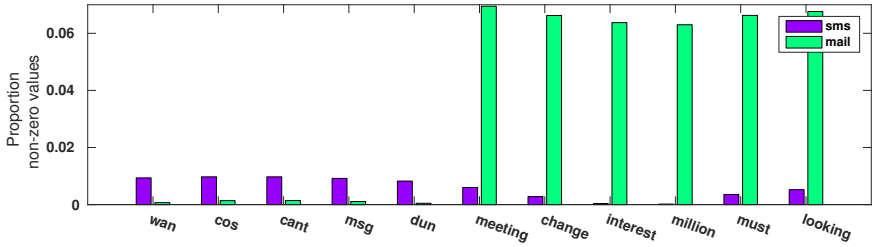


Figure 4.8: Proportion of non-zero values for a subset of words per domain on the spam data set.

Table 4.8 shows results from our classification experiments on the spam data set. As can be seen from the results of T-LR, fairly good accuracies can be obtained on the spam detection task. However, the domains are so different that the naive classifiers S-SVM and S-LR are performing according to chance or worse. Most of the domain-adaptation models do not appear to improve much over the naive models. For KMM this makes sense, as the importance weight estimator will assign equal values to each sample when the empirical supports of the two domains are disjoint. There might be some features that are shared between domains, *i.e.*, words that are spam in both emails and text messages, but considering the performance of SCL these might not be corresponding well with the other features. FLDA-q and FLDA-l are showing slight improvements over the naive classifiers, but the transfer model we used is too poor as the domains contain a large amount of increased word frequencies.

Table 4.8: Classification error rates obtained by ten (domain-adapted) classifiers for both domain pairs on the spam data set. (S='SMS' and M='E-Mail').

	S-SVM	S-LR	KMM	SCL	SA	GFK	TCA	FLDA-q	FLDA-l	T-LR
S→M	.460	.522	.521	.524	**.445**	.491	.508	.511	.521	.073
M→S	.830	.804	.799	.804	**.408**	.696	.863	.636	.727	.133

Amazon

We performed a similar experiment on the Amazon sentiment analysis data set of product reviews [49]. The data consists of a $30,000$ dimensional bag-of-words representations of

27,677 reviews with the labels derived from the dichotomized 5-star rating (ratings > 3 are +1 and ratings ≤ 3 as -1). Each review describes a product from one of four categories: books (6465 reviews), DVDs (5586 reviews), electronics (7681 reviews) and kitchen appliances (7945 reviews). Figure 4.9 shows the probability of a non-zero value for some example words in each category. Some words, such as 'portrayed' or 'barbaric', are very specific to one or two domains, but the frequencies of many other words do not vary much between domains. We performed experiments on the Amazon data set using the same experimental setup as before.

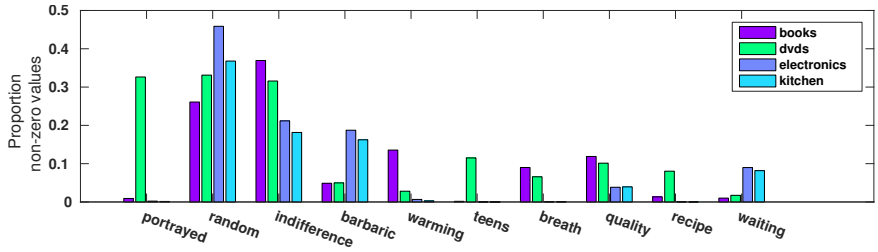


Figure 4.9: Proportion of non-zero values for a subset of words per domain on the Amazon data set.

In Table 4.9, we report the classification error rates on all pairwise combinations of domains. The difference in classification errors between S-LR and T-LR is up to 10%, which suggests there is potential for success with domain adaptation. However, the domain transfer is not captured well by SA, GFK, TCA: on average, these methods are performing worse than the naive classifiers. We presume this happens because only a small number of words are actually discriminative, and these words carry little weight in the sample transformation measures used. Furthermore, there are significantly less samples than features in each domain which means models with large amounts of parameters are likely to experience estimation errors. By contrast, FLDA-l performs strongly on the Amazon data set, achieving the best performance on many of the domain pairs. FLDA-q performs substantially worse than FLDA-l, presumably, because of the singular covariance matrix and the fact that least-squares classifiers are very sensitive to outliers.

4.5. Discussion and conclusions

We have presented an approach to domain adaptation, called FLDA, that fits a probabilistic model to capture the transfer between the source and the target data and, subsequently, trains a classifier by minimizing the expected loss on the source data under this transfer model. Whilst the FLDA approach is very general, in this chapter, we have focused on one particular transfer model, namely, a dropout model. Our extensive experimental evaluation with this transfer model shows that FLDA performs on par with the current state-of-the-art methods for domain adaptation.

An interesting interpretation of our formulation is that the expected loss under the transfer model performs a kind of data-dependent regularization [26]. For instance, if a quadratic loss function is employed in combination with a Gaussian transfer model, FLDA

Table 4.9: Classification error rates obtained by ten (domain-adapted) classifiers for all pairwise combinations of domains on the Amazon data set. (B='Books', D='DVD', E='Electronics', and K='Kitchen').

	S-SVM	S-LR	KMM	SCL	SA	GFK	TCA	FLDA-q	FLDA-l	T-LR
B→D	.180	.168	**.166**	.167	.414	.392	.413	.303	**.166**	.153
B→E	.217	.221	.222	.220	.372	.429	.369	.343	**.210**	.116
B→K	.188	.188	.189	**.184**	.371	.443	.338	.384	.185	.095
D→E	.201	.202	.205	.207	.403	.480	.385	.369	**.196**	.116
D→K	**.182**	**.182**	.185	.190	.330	.494	.360	.379	.185	.095
E→K	.108	.110	.106	.112	.311	.416	.261	.308	**.104**	.095
D→B	.192	.190	.191	.202	.351	.388	.420	.368	**.186**	.145
E→B	.257	.262	**.253**	.260	.372	.445	.481	.406	.261	.145
K→B	**.261**	.277	.268	.273	.414	.418	.426	.399	.271	.145
E→D	.245	.240	**.238**	.242	.398	.441	.427	.384	**.238**	.153
K→D	.230	.230	.230	.231	.383	.410	.400	.370	**.228**	.153
K→E	.123	.131	.126	.126	.290	.353	.296	.292	**.119**	.116

reduces to a transfer-dependent variant of ridge regression [50]. This transfer-dependent regularizer increases the amount of regularization on features when it is undesired for the classifier to assign a large weight to that feature. In other words, the regularizer forces the classifier to ignore features that are frequently present in the source domain but very infrequently present in the target domain.

In some of our experiments, the adaptation strategies are producing classifiers that perform worse than a naive classifier trained on the source data. A potential reason for this is that many domain-adaptation models make strong assumptions on the data that are invalid in many real-world scenarios. In particular, it is unclear to what extent the relation between source data and classes truly is informative about the target labels. This issue arises in every domain-adaptation problem: without target labels, there is no way of knowing whether matching the source distribution $p_{\mathcal{S}}$ to the target distribution $p_{\mathcal{T}}$ will improve the match between $p_{y|\mathcal{S}}$ and $p_{y|\mathcal{T}}$.

4.6. Appendix A

For some combinations of source and target models, the source domain can be integrated out. For others, we would have to resort to Markov Chain Monte Carlo sampling and subsequently averaging the samples drawn from the *transferred* source distribution $q_{\mathcal{T}}$. For the Bernoulli and dropout model defined in Equations 4.4 and 4.5 respectively, the integration as in Equation 4.6 can be performed by plugging in the specified probabilities and performing the summation:

$$
\begin{aligned}
q_{\mathcal{T}}(z \mid \eta, \zeta) &= \prod_{d=1}^{D} \int_{X_d} p_{\mathcal{T}|\mathcal{S}}(z_d \mid x_d, \zeta_d)\, p_{\mathcal{S}}(x_d \mid \eta_d)\, \mathrm{d}x_d \\
&= \prod_{d=1}^{D} p_{\mathcal{T}|\mathcal{S}}(z_d \mid [x_d = 0], \zeta_d)\, p_{\mathcal{S}}([x_d = 0]; \eta_d) + \\
&\qquad p_{\mathcal{T}|\mathcal{S}}(z_d \mid [x_d \neq 0], \zeta_d)\, p_{\mathcal{S}}([x_d \neq 0]; \eta_d) \\
&= \prod_{d=1}^{D} \begin{cases} p_{\mathcal{T}|\mathcal{S}}(z_d = 0 \mid [x_d = 0], \zeta_d)(1 - \eta_d) + p_{\mathcal{T}|\mathcal{S}}(z_d = 0 \mid [x_d \neq 0], \zeta_d)\eta_d \\ p_{\mathcal{T}|\mathcal{S}}(z_d \neq 0 \mid [x_d = 0], \zeta_d)(1 - \eta_d) + p_{\mathcal{T}|\mathcal{S}}(z_d \neq 0 \mid [x_d \neq 0], \zeta_d)\eta_d \end{cases} \\
&= \prod_{d=1}^{D} \begin{cases} 1(1 - \eta_d) + \zeta_d \eta_d & \text{if } z_d = 0 \\ 0(1 - \eta_d) + (1 - \zeta_d)\eta_d & \text{if } z_d \neq 0 \end{cases} \\
&= \prod_{d=1}^{D} \left(1 - (1 - \zeta_d)\, \eta_d\right)^{z_d = 0} \left((1 - \zeta_d)\, \eta_d\right)^{z_d \neq 0}.
\end{aligned}
$$

Note that we chose our transfer model such that the probability is 0 for a non-zero target sample value given a zero source sample value; $p_{\mathcal{T}|\mathcal{S}}(z_d \neq 0 \mid [x_d = 0], \zeta_d) = 0$. In other words, if a word is not used in the source domain, then we expect that it is also not used in the target domain. By setting different values for these probabilities, a different type of transfer is modeled.

4.7. Appendix B

This appendix lists the second-order Taylor approximation of the logistic loss version of FLDA with a general transfer model and presents its gradient with respect to the classifier parameters. To keep the derivations manageable, we use the following shorthands:

$$\alpha_i = \sum_{y' \in \mathcal{Y}} \exp(y' x_i \theta)$$
$$\beta_i = \sum_{y' \in \mathcal{Y}} y' \exp(y' x_i \theta)$$
$$\gamma_i = \sum_{y' \in \mathcal{Y}} \exp(y' x_i \theta) x_i^\top$$
$$\delta_i = \sum_{y' \in \mathcal{Y}} y' \exp(y' x_i \theta) x_i^\top .$$

The risk function of FLDA with a logistic loss and a general transfer model is:

$$\begin{aligned}\hat{R}_{\mathcal{T}}(h \mid \mathcal{D}_S) = \frac{1}{n}\sum_{i=1}^{n} & -y_i \mathbb{E}_{\mathcal{T}|x_i}[z]\theta + A(x_i\theta) + \\ & \frac{\partial A(x_i\theta)}{\partial x_i \theta}(\mathbb{E}_{\mathcal{T}|x_i}[z] - x_i)\theta + \frac{1}{2}\frac{\partial^2 A(x_i\theta)}{\partial (x_i\theta)^2}\,\theta^\top \mathbb{V}_{\mathcal{T}|x_i}[z]\theta \\ = \frac{1}{n}\sum_{i=1}^{n} & -y_i \mathbb{E}_{\mathcal{T}|x_i}[z]\theta + \log(\alpha_i) + \\ & \frac{\beta_i}{\alpha_i}(\mathbb{E}_{\mathcal{T}|x_i}[z] - x_i)\theta + \frac{1}{2}(1 - \frac{\beta_i^2}{\alpha_i^2})\,\theta^\top \mathbb{V}_{\mathcal{T}|x_i}[z]\theta .\end{aligned}$$

The gradient of the risk is:

$$\begin{aligned}\frac{\partial}{\partial\theta}\hat{R}_{\mathcal{T}}(h \mid \mathcal{D}_S) = \frac{1}{n}\sum_{i=1}^{n} & -y_i \mathbb{E}_{\mathcal{T}|x_i}[z]^\top + \frac{\beta_i}{\alpha_i}x_i^\top + \\ & (\frac{\gamma_i}{\alpha_i} - \frac{\beta_i\delta_i}{\alpha_i^2})(\mathbb{E}_{\mathcal{T}|x_i}[z] - x_i)\theta + \frac{\beta_i}{\alpha_i}(\mathbb{E}_{\mathcal{T}|x_i}[z] - x_i)^\top \\ & \frac{1}{2}(1 - \frac{\beta_i^2}{\alpha_i^2})(\mathbb{V}_{\mathcal{T}|x_i}[z] + \mathbb{V}_{\mathcal{T}|x_i}[z]^\top)\theta - (\frac{\beta_i^2\delta_i}{\alpha_i^3} - \frac{\beta_i\gamma_i}{\alpha_i^2})\theta^\top \mathbb{V}_{\mathcal{T}|x_i}[z]\theta .\end{aligned}$$

References

[1] C. Leggetter and P. Woodland, *Maximum likelihood linear regression for speaker adaptation of continuous density hidden markov models*, Computer Speech & Language **9**, 171 (1995).

[2] A. Van Opbroek, M. Ikram, M. Vernooij, and M. De Bruijne, *A transfer-learning approach to image segmentation across scanners by maximizing distribution similarity*, in *International Workshop on Machine Learning in Medical Imaging* (2013) pp. 49–56.

[3] K. Saenko, B. Kulis, M. Fritz, and T. Darrell, *Adapting visual category models to new domains*, European Conference on Computer Vision , 213 (2010).

[4] G. Zen, E. Sangineto, E. Ricci, and N. Sebe, *Unsupervised domain adaptation for personalized facial emotion recognition*, in *International Conference on Multimodal Interaction* (2014) pp. 128–135.

[5] V. Peddinti and P. Chintalapoodi, *Domain adaptation in sentiment analysis of twitter*, in *AAAI Conference on Artificial Intelligence* (2011).

[6] K. M. Borgwardt, A. Gretton, M. J. Rasch, H.-P. Kriegel, B. Schölkopf, and A. J. Smola, *Integrating structured biological data by kernel maximum mean discrepancy*, Bioinformatics **22**, e49 (2006).

[7] H. Shimodaira, *Improving predictive inference under covariate shift by weighting the log-likelihood function*, Journal of Statistical Planning and Inference **90**, 227 (2000).

[8] J. Huang, A. Gretton, K. M. Borgwardt, B. Schölkopf, and A. J. Smola, *Correcting sample selection bias by unlabeled data*, in *Advances in Neural Information Processing Systems* (2007) pp. 601–608.

[9] C. Cortes, M. Mohri, M. Riley, and A. Rostamizadeh, *Sample selection bias correction theory*, in *Algorithmic Learning Theory* (Springer, 2008) pp. 38–53.

[10] A. Gretton, A. Smola, J. Huang, M. Schmittfull, K. Borgwardt, and B. Schölkopf, *Covariate shift by kernel mean matching*, in *Dataset Shift in Machine Learning*, edited by Q. Candela, M. Sugiyama, A. Schwaighofer, and N. Lawrence (MIT Press, 2009) pp. 131–160.

[11] C. Cortes and M. Mohri, *Domain adaptation in regression*, in *Algorithmic Learning Theory* (2011) pp. 308–323.

[12] B. Gong, K. Grauman, and F. Sha, *Connecting the dots with landmarks: discriminatively learning domain-invariant features for unsupervised domain adaptation*, in *International Conference on Machine Learning* (2013) pp. 222–230.

[13] M. Baktashmotlagh, M. Harandi, B. Lovell, and M. Salzmann, *Domain adaptation on the statistical manifold*, in *Conference on Computer Vision and Pattern Recognition* (2014) pp. 2481–2488.

[14] J. Blitzer, R. McDonald, and F. Pereira, *Domain adaptation with structural correspondence learning*, in *Empirical Methods in Natural Language Processing* (2006) pp. 120–128.

[15] J. Blitzer, S. Kakade, and D. Foster, *Domain adaptation with coupled subspaces*, in *International Conference on Artificial Intelligence and Statistics* (2011) pp. 173–181.

[16] S. Pan, I. Tsang, J. Kwok, and Q. Yang, *Domain adaptation via transfer component analysis*, Neural Networks **22**, 199 (2011).

[17] R. Gopalan, R. Li, and R. Chellappa, *Domain adaptation for object recognition: an unsupervised approach*, in *International Conference on Computer Vision* (IEEE, 2011) pp. 999–1006.

[18] B. Gong, Y. Shi, F. Sha, and K. Grauman, *Geodesic flow kernel for unsupervised domain adaptation*, in *Conference on Computer Vision and Pattern Recognition* (IEEE, 2012) pp. 2066–2073.

[19] M. Baktashmotlagh, M. Harandi, B. Lovell, and M. Salzmann, *Unsupervised domain adaptation by domain invariant projection*, in *International Conference on Computer Vision* (2013) pp. 769–776.

[20] C. V. Dinh, R. P. Duin, I. Piqueras-Salazar, and M. Loog, *Fidos: A generalized fisher based feature extraction method for domain shift*, Pattern Recognition **46**, 2510 (2013).

[21] B. Fernando, A. Habrard, M. Sebban, and T. Tuytelaars, *Unsupervised visual domain adaptation using subspace alignment*, in *International Conference on Computer Vision* (2013) pp. 2960–2967.

[22] M. Shao, D. Kit, and Y. Fu, *Generalized transfer subspace learning through low-rank constraint*, International Journal of Computer Vision **109**, 74 (2014).

[23] W. Li, L. Duan, D. Xu, and I. Tsang, *Learning with augmented features for supervised and semi-supervised heterogeneous domain adaptation*, IEEE Transactions on Pattern Analysis and Machine Intelligence **36**, 1134 (2014).

[24] G. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, *Improving neural networks by preventing co-adaptation of feature detectors*, arXiv preprint arXiv:1207.0580 (2012).

[25] L. Van der Maaten, M. Chen, S. Tyree, and K. Weinberger, *Learning with marginalized corrupted features*, in *International Conference on Machine Learning* (2013) pp. 410–418.

[26] S. Wager, S. Wang, and P. Liang, *Dropout training as adaptive regularization*, in *Advances in Neural Information Processing Systems* (2013) pp. 351–359.

[27] N. Chen, J. Zhu, J. Chen, and B. Zhang, *Dropout training for support vector machines*, in *AAAI Conference on Artificial Intelligence* (2014).

[28] D. Blei, A. Ng, and M. Jordan, *Latent dirichlet allocation*, Journal of Machine Learning Research **3**, 993 (2003).

[29] H. Jégou, F. Perronnin, M. Douze, J. Sanchez, P. Perez, and C. Schmid, *Aggregating local image descriptors into compact codes*, IEEE Transactions on Pattern Analysis and Machine Intelligence **34**, 1704 (2012).

[30] A. Rostamizadeh, A. Agarwal, and P. Bartlett, *Learning with missing features*, in *Conference on Uncertainty in Artificial Intelligence* (2011) pp. 635–642.

[31] M. Sugiyama, M. Krauledat, and K.-R. Müller, *Covariate shift adaptation by importance weighted cross validation*, Journal of Machine Learning Research **8**, 985 (2007).

[32] C. Chang and C. Lin, *LIBSVM: A library for support vector machines*, ACM Transactions on Intelligent Systems and Technology **2**, 27:1 (2011), software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.

[33] R. K. Ando and T. Zhang, *A framework for learning predictive structures from multiple tasks and unlabeled data*, Journal of Machine Learning Research **6**, 1817 (2005).

[34] M. Lichman, *UCI machine learning repository*, (2013).

[35] K. Zhang and W. Fan, *Forecasting skewed biased stochastic ozone days: analyses, solutions and beyond*, Knowledge and Information Systems **14**, 299 (2008).

[36] R. Detrano, A. Janosi, W. Steinbrunn, M. Pfisterer, J. Schmid, S. Sandhu, K. Guppy, S. Lee, and V. Froelicher, *International application of a new probability algorithm for the diagnosis of coronary artery disease*, American Journal of Cardiology **64**, 304 (1989).

[37] M. Elter, R. Schulz-Wendtland, and T. Wittenberg, *The prediction of breast cancer biopsy outcomes using two CAD approaches that both emphasize an intelligible decision process*, Medical Physics **34**, 4164 (2007).

[38] H. Guvenir, B. Acar, G. Demiroz, and A. Cekin, *Supervised machine learning algorithm for arrhythmia analysis*, Computers in Cardiology , 433 (1997).

[39] D. Rubin, *Inference and missing data*, Biometrika **63**, 581 (1976).

[40] R. J. Little and D. B. Rubin, *Statistical analysis with missing data* (John Wiley & Sons, 2014).

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

[42] J. Hull, *A database for handwritten text recognition research*, IEEE Transactions on Pattern Analysis and Machine Intelligence **16**, 550 (1994).

[43] J. Hoffman, E. Rodner, J. Donahue, T. Darrell, and K. Saenko, *Efficient learning of domain-invariant image representations*, International Conference on Learning Representations (2013).

[44] H. Bay, T. Tuytelaars, and L. Van Gool, *SURF: Speeded Up Robust Features*, in *European Conference on Computer Vision* (2006) pp. 404–417.

[45] J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, and T. Darrell, *DeCAF: A deep convolutional activation feature for generic visual recognition*, in *International Conference on Machine Learning* (2014) pp. 647–655.

[46] B. Pang and L. Lee, *A sentimental education: sentiment analysis using subjectivity summarization based on minimum cuts*, in *Association for Computational Linguistics* (2004) p. 271.

[47] B. Klimt and Y. Yang, *Introducing the enron corpus*, (2004).

[48] T. Almeida, J. Hidalgo, and A. Yamakami, *Contributions to the study of sms spam filtering: new collection and results*, in *ACM Symposium on Document Engineering* (2011) pp. 259–262.

[49] J. Blitzer, M. Dredze, F. Pereira, *et al.*, *Biographies, bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification*, in *Association for Computational Linguistics*, Vol. 7 (2007) pp. 440–447.

[50] C. Bishop, *Training with noise is equivalent to tikhonov regularization*, Neural Computation **7**, 108 (1995).

5

Acquisition-invariant representations

*Voxelwise classification approaches are popular and effective methods for tissue quantification in brain magnetic resonance imaging (MRI) scans. However, generalization of these approaches is hampered by large differences between sets of MRI scans such as differences in field strength, vendor or acquisition protocols. Due to this acquisition related variation, classifiers trained on data from a specific scanner fail or under-perform when applied to data that was acquired differently. In order to address this lack of generalization, we propose a Siamese neural network (*MRAI-NET*) to learn a representation that minimizes the between-scanner variation, while maintaining the contrast between brain tissues necessary for brain tissue quantification. The proposed* MRAI-NET *was evaluated on both simulated and real MRI data. After learning the MR acquisition invariant representation, any supervised classification model that uses feature vectors can be applied. In this chapter, we provide a proof of principle, which shows that a linear classifier applied on the* mrai *representation is able to outperform supervised convolutional neural network classifiers for tissue classification when little target training data is available.*

This chapter is based on the paper ”MR acquisition-invariant representation learning”.

5.1. Introduction

Very few of the many medical image analysis algorithms that were proposed in the literature are applicable in clinical practice. One of the reasons for this is the complexity of the medical image data, i.e. the vast amount of variation that is present in this data. A more specific example of this, is brain tissue segmentation in MRI scans. Many automatic methods have been proposed [1–8], but due to a lack of generalization, large scale use in clinical practice remains a challenge [9]. In order to test the capacity of algorithms to generalize to new data, a representative sample (dataset) is required. This entails identifying all factors of variation in the data that would influence algorithm performance with respect to the medical image analysis task at hand. For brain tissue segmentation in MRI scans, we identify for example subject related variation (i.e. pathology, age, ethnicity, gender) and acquisition related variation (i.e. MR field strength, protocol settings, scanner vendor, artefacts).

Supervised voxel classification approaches have been shown to perform well on small data sets [10–12]. However, in order to ensure generalization, these algorithms should be trained and tested on a sufficiently large representative dataset that covers all possible types of variation. This is practically infeasible since training and testing require not only the MRI scans, but also manual labels as ground truth. The manual segmentation process is labor intensive and time consuming, and adds another layer of variation due to non-standardized manual segmentation protocols and inter- and intra-observer variability. To address this problem, we propose an alternative approach, by learning a representation of the data [13] that is invariant to disturbing types of variation, while preserving the variation relevant for the selected classification task, i.e. clinically relevant variation. By reducing undesired variation, this method has the potential to decrease the number of fully labeled samples required for generalization and enable broader use of voxel classification approaches.

Overcoming acquisition-variation is a relatively new challenge in medical imaging. One particularly interesting approach focuses on weighting classifiers based on how well their training data matches the test data [10, 14, 15]. Examples of transfer classifiers include weighted SVM's [10] and weighted ensembles [15]. But these methods are very dataset-dependent: the classifiers need to be retrained for every new test dataset. Ideally, we would like to have a method that removes acquisition-variation or extracts acquisition-invariant features. Domain adaptation researchers have proposed representation learning methods that explicitly maximize "domain confusion": if a classifier cannot distinguish between domains then the representation is domain-invariant [16–18]. For MRI scans, different scanners or acquisition protocols constitute different data domains. These representation learning methods are variants of deep neural networks, called domain-adversarial networks. They have two layers in which a loss function is computed: one layer for the task-dependent loss, such as tissue or lesion classification, and one that maximizes domain confusion. The networks learn representations in which the data from each domain overlaps while the different classes become separable [16]. They are adversarial because the loss layers operate with different objectives, which can make them very difficult to train [17–19]. A recent paper has applied domain-adversarial networks to segmenting brain le-

sions [20]. They achieved excellent performance and provided an in-depth analysis of the adversarial training procedure. However, their networks are still very task-dependent: the learned representation works well for brain lesion segmentation but cannot be used for tumor detection for example. It is not a method for learning a general acquisition-invariant representation.

We propose to learn a general representation by marking certain factors of variation as desirable and others as undesired [21]. Learning a representation by explicitly minimizing undesirable factors of variation while maintaining desirable factors will produce a task-independent representation, which can be used for a variety of tasks later on. In order to minimize certain factors of variation while maintaining others, we exploit a particular type of neural network, referred to as a Siamese network [22]. Our work was inspired by the work of Hadsell [23], who used Siamese neural networks to learn a lighting-invariant representation for airplane images in the NORB [24] dataset. In this chapter, we aim to provide a proof of principle for learning an MR-acquisition invariant (MRAI-NET) representation for MR brain tissue segmentation.

To test MRAI-NET we simulated MRI data (SIMRI [25–27]) from a 1.5T scanner and 3T scanner based on acquisition protocols used to acquire real data (Section 5.4.3) and real tissue segmentations from healthy adults (Brainweb). In addition we used real patient data (3T) as provided by MRBrainS [28]. We acknowledge that the simulated data is idealistic as compared to real patient data. However, experiments in a controlled environment provide a proof of principle to ensure that the method is behaving appropriately. Translation to real patient data is provided by including the MRBrainS data. For the experiments with the simulated data (Section 5.4.4 and 5.4.5), the same subject acquired with different acquisition protocols is used. This is however not a prerequisite to train MRAI-NET. For the experiments that use real patient data, different subjects are used. MRAI-NET is not trained by using tissue labels, but with patches labeled as similar or dissimilar. Factors of variation that should be preserved should be labeled as dissimilar, MRAI-NET will aim to reduce all other factors of variation.

5.2. MR acquisition-invariant network

Neural networks transform data based on minimizing a loss function. In supervised neural networks, labels are used to determine the loss (error between prediction and label). Many labels are required to learn a task. We aim to use as little labels as possible to learn a representation in which the variation over different methods of acquisition is minimal, without destroying the variation relevant to distinguish between brain tissues.

The proposed network works as follows. Suppose that we have scans that are acquired in two different ways (A and B). Possible differences can be in field strength, scanner vendor, acquisition protocol, and so on. A tissue patch, for example gray matter, is selected from both scans A and B. The aim is to teach the network that both these patches are gray matter regardless of their acquisition variation. Therefore, we use a loss function that expresses that in MRAI-NET's representation, pairs of samples from the same tissue but from

different scanners should be as similar as possible. However, that expression alone would cause all samples to be mapped to a single point and would destroy variation between tissues. To balance out the action of pulling pairs marked as similar together, it is necessary to push other pairs apart [23]. Since we want to maintain the relevant variation between tissues, we additionally express that in MRAI-NET's representation, pairs from different tissues should retain their dissimilarity. The loss function is described in section 5.2.1. Section 5.2.2 describes how pairs of samples are labeled as similar or dissimilar. The Siamese neural network that is used to learn the acquisition-invariant representation is described in section 5.2.3. The network consists of two pipelines with shared weights and a Siamese loss layer that acts on the output layer of the two pipelines.

5.2.1. Siamese loss

Neural networks transform data in each layer. We summarize the total transformation from input to output with the symbol f, i.e. patch a will be mapped to the new representation with $f(a)$ and patch b will be mapped with $f(b)$. To find an optimal transformation, we employ a loss function based on distances between pairs of patches in the output representation, i.e. $\|f(a) - f(b)\|$. Pairwise distances are computed through an L^1-norm, denoted by $\|\cdot\|_1$. We used an L^1-norm as opposed to for instance an L^2-norm, because larger values of p in L^p-norms either result in problems in high-dimensional spaces or result in problems with the gradient during optimization (see Appendix 5.8).

The loss function for the similar pairs consists of the squared distance, $\ell_{\text{sim}}(f \mid a, b) = (\|f(a) - f(b)\|_1)^2$. We chose this formulation in order to express that large distances are less desirable. The loss function for the dissimilar pairs consists of a hinge loss, where the distance is subtracted from a margin parameter m and the negative values are set to 0: $\ell_{\text{dis}}(f \mid a, b) = \max(0, m - \|f(a) - f(b)\|_1)$. Pairs that lie close together will suffer a loss, while pairs that are pushed sufficiently apart, i.e. past the margin, will not suffer a loss. We discuss the effect of the margin parameter in Section 5.4.7.

Each pair of patches is marked with a similarity variable; $y = 1$ for similar and $y = 0$ for dissimilar. Using the similarity label we can combine the similar and dissimilar loss functions into a single loss function:

$$\begin{aligned}\ell(f \mid \mathcal{D}) &= \sum_i y_i\, \ell_{\text{sim}}(f \mid a_i, b_i) + (1 - y_i)\, \ell_{\text{dis}}(f \mid a_i, b_i)) \\ &= \sum_i y_i \|f(a_i) - f(b_i)\|_1^2 + (1 - y_i) \max(0, m - \|f(a_i) - f(b_i)\|_1)\ .\end{aligned}$$

where i iterates over pairs and $\mathcal{D}$ refers to the whole dataset of pairs.

This type of loss function is known as a *Siamese* loss [22, 23]. Note that it is asymmetric: it penalizes samples from one class differently than samples from another class.

5.2.2. Labeling pairs as similar or dissimilar

As described above, suppose we have two medical images from two different scanners; A and B. Assume that we have sufficient manual segmentations (labeled voxels) on scans

from scanner A, to train a supervised classifier, but a very limited amount of labels from scanner B, for example 1 labeled voxel per tissue for 1 subject. The data from scanner A will be referred to as the source set, and the data from scanner B as the target set. Let K be the set of tissue labels. The set of patches extracted from Scanner A is denoted $\{(a_{tn}\}_{n=1}^{N}$, and the set from scanner B is denoted $\{b_{tm}\}_{m=1}^{M}$, with t specifying the sample's tissue. Given these two sets of patches, we form sets of similar and dissimilar pairs, with a similarity label y. The following pairs are labeled as similar ($y = 1$) and therefore will be pulled closer together:

- Source patches from the same tissue $k \in K$: $\{(a_{t=k}, a_{t=k})\}$,
- Source and target patches from the same tissue $k \in K$: $\{(a_{t=k}, b_{t=k})\}$,
- Target patches from the same tissue $k \in K$: $\{(b_{t=k}, b_{t=k})\}$.

The subscript $t = k$ selects all patches that belong to tissue k. The following pairs are labeled as dissimilar ($y = 0$) and therefore will be pushed apart:

- Source patches from different tissues $k, l \in K$: $\{(a_{t=k}, a_{t=l})\}$,
- Source and target patches from different tissues $k, l \in K$: $\{(a_{t=k}, b_{t=l})\}$,
- Target patches from different tissues $k, l \in K$: $\{(b_{t=k}, b_{t=l})\}$.

Figure 5.1 illustrates the process of selecting pairs of patches from different scanners. Consider a medical image from scanner A and scanner B, with 2 GM patches (green), 1 WM patch (yellow) and 1 CSF patch (blue) for each image. Using these patches we can generate the following pairs: a GM patch from A with another GM patch from A $(a_{t=k}, a_{t=k})$, a GM patch from A with a GM patch from B $(a_{t=k}, b_{t=k})$, a GM patch from B with another GM patch from B $(b_{t=k}, b_{t=k})$, a GM patch from A with a CSF patch from A $(a_{t=k}, a_{t=l})$, a GM patch from B with a WM patch from B $(a_{t=k}, b_{t=l})$, and a GM patch from B with a CSF patch from B $(b_{t=k}, b_{t=l})$. The bottom of the image shows examples of these 6 pairs of patches.

The pairs are concatenated into a dataset $\mathcal{D} = \{(a_i, b_i, y_i)\}_{i=1}^{C}$, where i iterates over the pairs. In total, the number of combinations is $C = \sum_{k \in K}(N_k + M_k)^2 + \sum_{(k,l) \in \binom{K}{2}}(N_k N_l + N_k M_l + M_k M_l)$, where N_k refers to the number of source patches from the k-th tissue and, likewise, M_k refers to the number of target patches from the k-th tissue. The number of pairs that can be generated is very large, even when only a small number of patches is available. For example, taking 10 patches of 3 tissues from 4 source scans and 1 patch of 3 tissues from 1 target scan, results in 2784 pairs of patches that can be used for training the deep neural network.

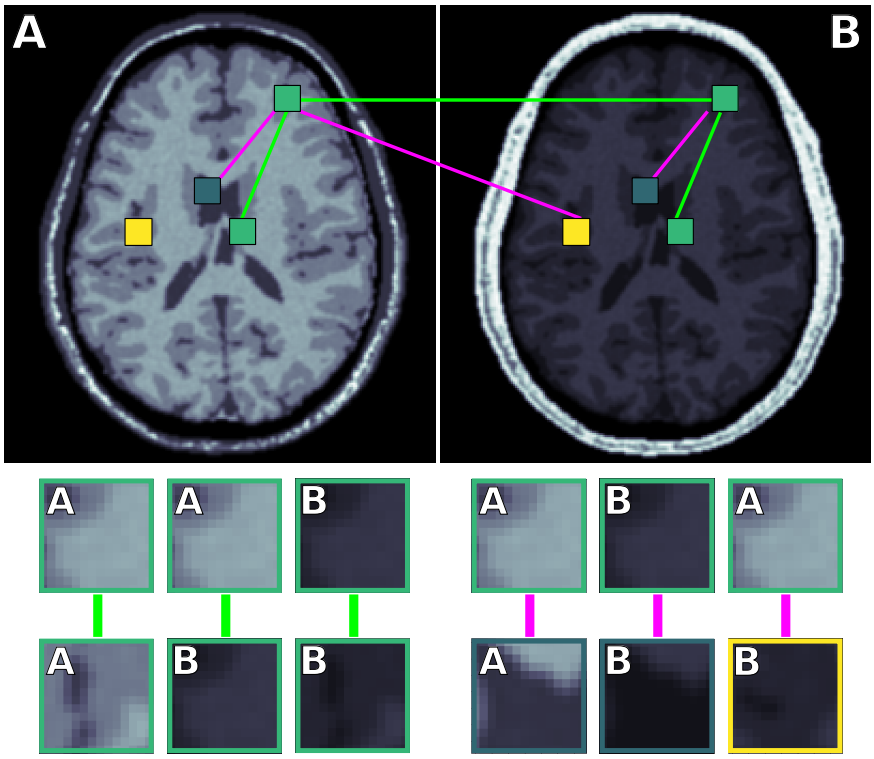


Figure 5.1: Illustration of extracting pairs of patches from images from scanner A and B. (Top) Each image shows 4 patches: 2 gray matter ones (green), 1 cerebrospinal fluid (blue) and 1 white matter (yellow). The lines mark the 6 types of combinations from Section 5.2.2. Green lines indicate similar pairs and purple lines indicate dissimilar pairs. (Bottom) Enlarged patches belonging to the 6 pairs marked in the top images.

5.2.3. Network architecture

Figure 5.2 shows a diagram of the network architecture. The network consists of two pipelines and a Siamese loss layer that acts on the output layers (red nodes). Pairs of patches enter the input layer (black squares) where they are convolved (blue squares) and mapped to feature vectors (blue nodes). The final layer is a low-dimensional feature space (red nodes). The Siamese loss layer (section 5.2.1) calculates the distance between each pair in their new representation and computes the loss based on whether the pair is marked as similar or dissimilar. The two pipelines share their weights, which means they are constrained to perform the same transformation. During training, the loss is propagated back through the network, adjusting the network weights.

Width and depth of the network may vary. We made the following choices: input patches are size $[15{\times}15]$ and scanner identification is set to a single variable. The convolution block consists of 8 kernels of size $[3 \times 3]$ with a rectifying linear unit (ReLU) activation function and a max-pooling layer of size $[2 \times 2]$. The output of these operations is flattened and the scanner ID (0 for source, 1 for target) is appended. The scanner ID ensures that regions

of different tissues in different scanners do not overlap in the input space. The flattened and pooled convolutional layer output, plus the scanner ID is then densely mapped to a 16-dimensional representation. A dropout noise of size 0.2 is set for each edge. This 16-dimensional representation is then densely mapped, again with a dropout of 0.2, to an 8-dimensional representation, which is finally mapped to a 2-dimensional representation. We chose a final representation of 2 dimensions because this allows for scatter plot visualizations.

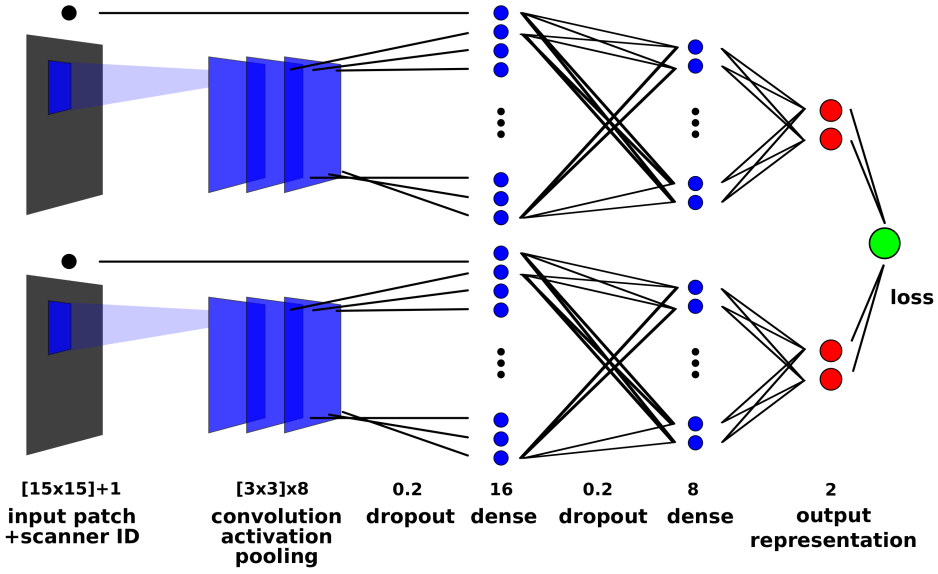


Figure 5.2: Schematic of MRAI-NET's architecture. Pairs of patches are fed into two pipelines that share parameters (i.e. produce the same mapping). The red nodes depict the representation in the final layer, while the green node depicts the loss function.

Our method is implemented in a combination of Tensorflow[1] and Keras[2] [29, 30]. This proof of principle uses a 4-layer hybrid convolutional-dense network for the pipeline. However, the network architecture can be changed. Variations involve, for example, more layers, wider layers, larger convolution kernels, and heavier max-pooling. See Section 5.4.6 for an experiment that varies the layer widths in the network.

Regularization

During training, we apply an l_2-regularization of 0.001 to every layer with weights. Regularization punishes the size of the weights, which prevents model over-complexity. In our experiments, the regularization parameter could be increased or decreased by two orders of magnitude with little effect on the networks performance. It is however always neces-

[1] https://www.tensorflow.org/
[2] https://keras.io/

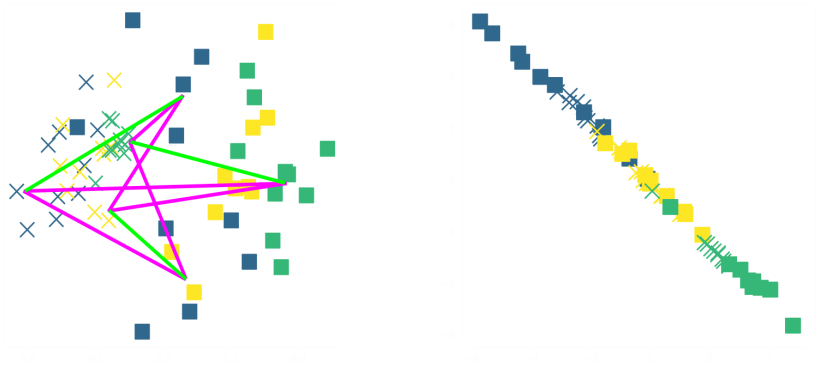
sary to include *some* regularization as there is not only the danger of overfitting to training data but also the danger of overfitting to the specific target subject used for training.

Optimization

All experiments in this chapter are performed with the default backpropagation algorithm "RMSprop", which normalizes the gradient update with a running average of itself [31]. Its default parameters are: a learning rate of 0.001, a ρ of 0.9, an ϵ of 1e-08, and a weight decay factor of 0.0 (see [31] for more details on optimizer parameters). RMSprop is based on stochastic gradient descent, which splits the dataset into batches and updates the networks parameters after processing each batch. An epoch is the number of times the optimization procedure splits the training set into batches. The number of epochs cannot be too large, otherwise the network starts to overfit to the specific target subject from which the target patches originated.

During experimentation we found that it is important that the batches are well-mixed with respect to the 6 types of pairs outlined in Section 5.2.2. If this is not the case, such as when one batch mostly consists of similar gray-matter patches and another batch consists mostly of dissimilar gray-matter / white-matter patches, then the network tends to push and pull in the same direction. These actions cancel each other out. The overall effect of having too many uniform batches is that the optimization procedure is slowed down.

5.3. Tissue segmentation



(a) Representation before training. (b) Representation after training.

Figure 5.3: Conceptual visualization of MRAI-NET's training procedure: the network pulls the similar pairs (green lines) closer together and pushes dissimilar pairs (purple lines) apart until it learns a representation in which the variation between scanners is minimal while the variation between tissues is maintained.

Figure 5.3 illustrates the training procedure of MRAI-NET. Once it is trained and an MR acquisition-invariant representation is learned, it can be used as a preprocessing step for tissue segmentation (Figure 5.4). Because of the shared weights, either one of the pipelines

can be used to transform the input patches into MRAI-NET's representation. Input patches from both the source and target scanner can be fed into the network, and any supervised classification model that uses feature vectors can subsequently be trained to distinguish tissues in the acquisition-invariant representation. Once the supervised classifier is trained, both the trained MRAI-NET and the trained supervised classifier are used to segment a new image. This is done by feeding a new patch through MRAI-NET and letting the tissue classifier predict the label in the MR acquisition invariant space. In this way, MRAI-NET acts as a preprocessing step to ensure that acquisition-based variation does not affect the tissue classifier.

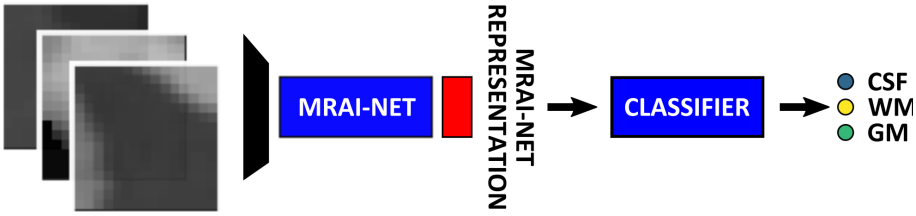


Figure 5.4: A dataset of tissue-labeled single patches is fed through MRAI-NET and represented in the acquisition-invariant space. Subsequently, a classifier is trained to distinguish tissues. A new image is decomposed into patches and fed through the network as well. The trained tissue classifier then makes a prediction for each patch. The predictions are then reshaped back into an image, resulting in the tissue segmentation.

5.4. Evaluating representations

Since the aim of MRAI-NET is to preserve variation between tissues while reducing the MR acquisition related variation, two different measures of performance are used to evaluate MRAI-NET. MR acquisition invariance is measured with the proxy $\mathcal{A}$-distance that measures the distance between the source and target scanner patches, as described in section 5.4.1. The preservation of tissue variation is measured using the tissue classification performance, and compared to supervised classification with CNN (Section 5.4.2). Section 5.4.3 describes the simulated (Brainweb 1.5T, Brainweb3.0T) and real data (MRBrainS) used for the experiments. For each experiment a source and target domain was specified. Four source subjects (100 random patches per tissue) and 1 target subject (1-1000 patches per tissue depending on experiment) were used for training. Four independent target subjects (100 random patches per tissue) were used for testing.

In total, we set up four experiments: 1) Only 1 patch per tissue from the target domain subject is used for training both the supervised CNNs (source, target) as well as MRAI-NET followed by a linear classifier on the simulated data (Brainweb1.5T, Brainweb3.0T), 2) Multiple target training samples per tissue (randomly selected with 50 repeats) are used for training the source, target, and MRAI-NET classifiers for both simulated (Brainweb3.0T) and real patient data (MRBrainS). The first experiment (Section 5.4.4) was set-up to test if only 1 target patch per tissue would be sufficient in order to learn an MR-acquisition invariant representation. If so, then calibrating a supervised segmentation algorithm for a new scanner using MRAI-NET would require only three clicks in one scan acquired with a new scanner. The second experiment (Section 5.4.5) illustrates the performance of MRAI-

NET compared to the target, and MRAI-NET classifiers when adding more target training samples (Figure 5.7). Results of using 1 patch per tissue and 100 patches per tissue from the target subject for training are shown in Figures 5.8-5.10. The third experiment (Section 5.4.6) looks at the performance of the network if we vary the number of convolution kernels and the number of nodes in the dense layers. For the setting where Brainweb1.5T is the source scanner and Brainweb3.0T is the target scanner, the network will keep gaining in performance at the cost of adding tens of thousands more parameters. Finally, the fourth experiment (Section 5.4.7) shows the influence of the margin parameter on the Siamese loss function. If the margin parameter is set too low, tissue variation will not be preserved. On the other hand, if the margin parameter is set too high the acquisition variation will not be reduced. The next two sections describe how these two types of variation are measured.

5.4.1. MR acquisition invariance measure

The $\mathcal{H}$-divergence can be used as a measure of the discrepancy between the source and target scanner data sets [32–34]. This divergence relies on the ability of a classifier to distinguish between domains. If a classifier is not able to distinguish source from target, i.e. has a test error of $1/2$, then invariance is achieved. Unfortunately, the original $\mathcal{H}$-divergence is a measure between distributions and not samples. Since we only have samples, we use its proxy instead: the $\mathcal{A}$-distance [33, 34], as used in [18]. The proxy $\mathcal{A}$-distance, denoted by $d_{\mathcal{A}}$, is defined as follows:

$$d_{\mathcal{A}}(x, z) = 2(1 - 2e(x, z)) , \tag{5.1}$$

where e represents the test error of a classifier trained to discriminate source samples x from target samples z. If the source and target data lie far apart, the error will be close to 0, i.e. perfect separability, and the proxy $\mathcal{A}$-distance will be close to 2. If the source and target data overlap, the error will be around 0.5, i.e. no separability (invariance), and the proxy $\mathcal{A}$-distance will approach 0. We use a linear support vector machine (SVM) as domain classifier.

5.4.2. Measure of preserving tissue variation

The tissue classification error is used as a measure of tissue variation preservation. The aim is to learn a linearly separable representation with MRAI-NET, to aid the number of methods that can be used for classification. Therefore, we evaluate the tissue classification error of the samples in the acquisition-invariant representation with a logistic regressor. The classifier is ℓ_2-regularized and cross-validated for optimal regularization parameters. This classifier MRAI-NET, based on MRAI-NET, is compared to two other supervised classifiers: 1) source classifier: a convolutional-dense neural network (CNN) trained on samples from the source (4 subjects) and target data (1 subject), and 2) target classifier: a CNN trained on samples from the target data (1 subject). In order to ensure that differences in performance between source, MRAI-NET and target are not due to differences between classifiers, the network architecture from MRAI-NET (Figure 5.2) was used for the source and target classifiers as well.

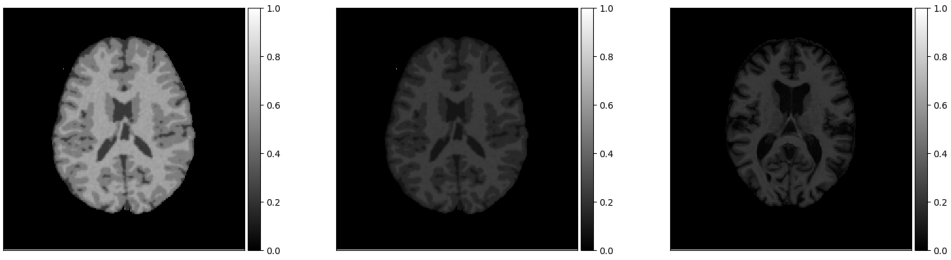
5.4.3. MRI-scan data sets

To be able to provide a proof of principle, we simulated different MR acquisitions from various anatomical models of the human brain [27, 35], using an MRI simulator (SIMRI [25–27]). The anatomical models consist of transverse slices of 20 normal brains and are publicly available through Brainweb[3]. These models were used as input for the MRI simulator. For the experiments, we simulated two acquisition types: 1) Brainweb1.5T, a standard gradient-echo acquisition protocol for a 1.5 Tesla scanner (c.f. [36]), and 2) Brainweb3.0T, a standard gradient-echo protocol for a 3.0 Tesla scanner (c.f. [28]). Table 5.1 describes the parameters used for the simulation: magnetic field strength (B0), flip angle (θ), repetition time (TR), echo time (TE). Magnetic field inhomogeneities and voxel inhomogeneity (partial volume effects) were not included in the simulation.

Table 5.1: SIMRI Acquisition parameters for the simulation of the Brainweb1.5T and Brainweb3.0T data sets.

	B0	θ	TR	TE
Brainweb1.5T	1.5 Tesla	20°	13.8 ms	2.8 ms
Brainweb3.0T	3.0 Tesla	90°	7.9 ms	4.5 ms

Appendix 5.7 describes the nuclear magnetic resonance (NMR) relaxation times for the tissues in the Brainweb anatomical models, for 1.5 and 3.0 Tesla field strengths. The tissues in the anatomical models are grouped into "background" (BKG), "cerebrospinal fluid" (CSF), "gray matter" (GM), and "white matter" (WM) to compose the ground truth segmentation labels for the simulated scans. The simulations result in images of 256 by 256 pixels, with a 1.0x1.0mm resolution. Figures 5.5a and 5.5b show examples of the Brainweb1.5T and Brainweb3.0T scan of the same subject. For all scans, we used a brain mask to strip the skull.



(a) Brainweb1.5T (b) Brainweb3.0T (c) MRBrains

Figure 5.5: Example of an MRI scan of a Brainweb anatomical model simulated with SIMRI with a 1.5T protocol (a) and a 3.0T protocol (b), and a real patient scan (MRBrainS) acquired with a 3.0T protocol (c).

In order to test the proposed method on real data, we use the publicly available training data (5 subjects) from the MRBrainS challenge[4]. The acquisition parameters used for simulating the Brainweb3.0T are based on the MRBrainS acquisition protocol (3.0T scanner,

[3] http://www.bic.mni.mcgill.ca/brainweb/
[4] http://mrbrains13.isi.uu.nl/Figure

gradient-echo, B0 = 3.0T, $\theta = 90^\circ$ flip angle, TE = 4.5ms, and TR = 7.9ms). Figure 5.5c shows an example of an MRBrainS scan. Again, a brain mask is used to strip the skull.

5.4.4. One target label per tissue

The first experiment with the simulated data tests the scenario described at the beginning of this section: suppose a supervised classification algorithm trained on one scanner needs to be calibrated for a new scanner, would this be possible with three clicks (1 for each tissue type) using MRAI-NET? To study this, we manually selected 1 patch for each tissue in the target scan (1 subject) and used this data to train MRAI-NET. Once MRAI-NET has been trained and an acquisition-invariant representation has been learned, we compute the proxy $\mathcal{A}$-distance and perform a tissue classification experiment.

For computing the proxy $\mathcal{A}$-distance, we used scans from 10 source subjects and 10 target subjects that had been held back (i.e. we did not draw samples from them to either train MRAI-NET or train any of the tissue classifiers). We randomly drew 50 patches per tissue from each subject, resulting in two sets of 1500 patches. These patches were fed into MRAI-NET which mapped them to the new acquisition-invariant representation. The datasets were labeled 0 and 1 for source and target. Next, we trained a linear classifier with 5-fold cross-validation to obtain a test error on data set discrimination. Finally, using this test error and Equation 5.1, we computed the proxy $\mathcal{A}$-distance.

For evaluating the tissue classification performance, we used scans from 10 target subjects that had been held back. From these 10 scans, we drew 50 patches per tissue at random, for a total of 1500 patches. We computed the error rate by computing the proportion of wrong predictions on this test set. We trained the following three classifiers (described in Section 5.4.2): firstly, the source classifier (CNN) was trained on images from the source dataset, and applied to the test set to make predictions. Secondly, we trained a linear classifier on the source data mapped to MRAI-NET's representation. We mapped the test data to MRAI-NET's representation as well and applied the trained linear classifier to make predictions. Its performance on the test set is indicated with MRAI-NET in Table 5.2. The target classifier (CNN) was applied to the available target patches. In this experiment, there were 3 target patches in total, which is far too little data to train such a large convolutional network. We included its performance to indicate that using the target classifier in this kind of situation is not a sensible option.

For comparison, we performed the same experiment but with randomly selected target patches. Table 5.2 lists the tissue classification errors of the three classifiers and the proxy $\mathcal{A}$-distance between the source and target patches before (raw) and after (rep) applying MRAI-NET. The whole experiment was repeated 10 times and the average error rate is reported with the standard error of the mean between brackets.

Figure 5.6 displays the manually selected patches and their position within the image. For both the source and target classifier, one target patch per tissue is insufficient to achieve good tissue classification performance (5.2 (top row): 0.631 and 0.613). However, the

Table 5.2: Manually versus randomly selecting 1 target patch per tissue from 1 subject. (Left) Tissue classification error is reported for MRAI-NET (linear classifier after MRAI-NET), source (supervised CNN trained on source patches and 1 target patch per tissue), and target (supervised CNN trained on 1 target patch per tissue) tested on the target test data. (Right) Proxy $\mathcal{A}$-distance between the original source and target patches (raw) and the source and target patches after applying MRAI-NET (rep).

	source	MRAI-NET	target	raw	rep
manual	0.631 (.02)	0.223 (.01)	0.613 (.01)	1.88 (.01)	0.26 (.05)
random	0.667 (.02)	0.250 (.02)	0.610 (.06)	1.91 (.01)	0.41 (.06)

MRAI-NET classifier shows considerably better performance (0.223), using only one target patch per tissue. The proxy $\mathcal{A}$-distance also drops from near perfect separability (1.88) to near invariance (0.26). Randomly selecting (10 repeats) 1 target patch per tissue (Table 5.2 (bottom row)), shows worse performance of the MRAI-NET classifier, for both the classification error (0.250) as well as the $\mathcal{A}$-distance (0.41). Suggesting that purposive (information rich) sampling beats random sampling in this case.

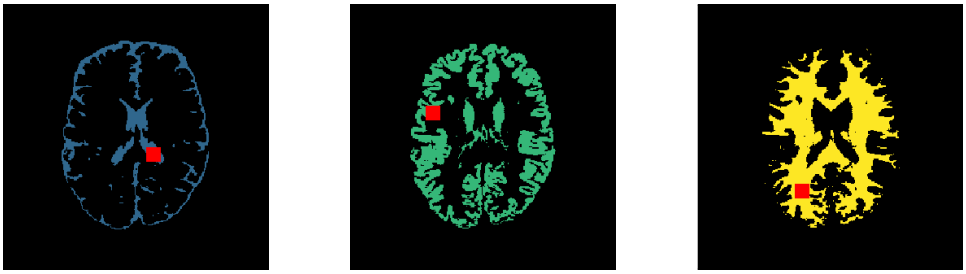


Figure 5.6: Locations of the manually selected target patches (red squares): Blue = cerebrospinal fluid, green = gray matter, yellow = white matter.

5.4.5. Multiple target labels per tissue

The second experiment tests the performance when adding more target training samples, for both simulated (Brainweb3.0T) and real patient data (MRBrainS). We set-up the following sub-experiments:

2.1) Experiment on simulated data with two different acquisition protocols (Source: Brainweb1.5T, Target: Brainweb3.0T).

2.2) Experiment on 1.5T simulated data and 3.0T real data (Source: Brainweb1.5T, Target: MRBrainS).

2.3) Experiment on 3.0T simulated data and 3.0T real data (Source: Brainweb3.0T, Target: MRBrainS).

Each of these experiments is repeated 50 times. Figure 5.7 shows the performance (both tissue classification error as well as proxy $\mathcal{A}$-distance) as a function of the number of used target training samples. The average error (solid line) and the standard error of the mean (line thickness) is shown, ranging from using 1 target patch up to more than 1000 target

patches per tissue for training both the supervised CNNs (source, target) as well as the MRAI-NET followed by a linear classifier (MRAI-NET).

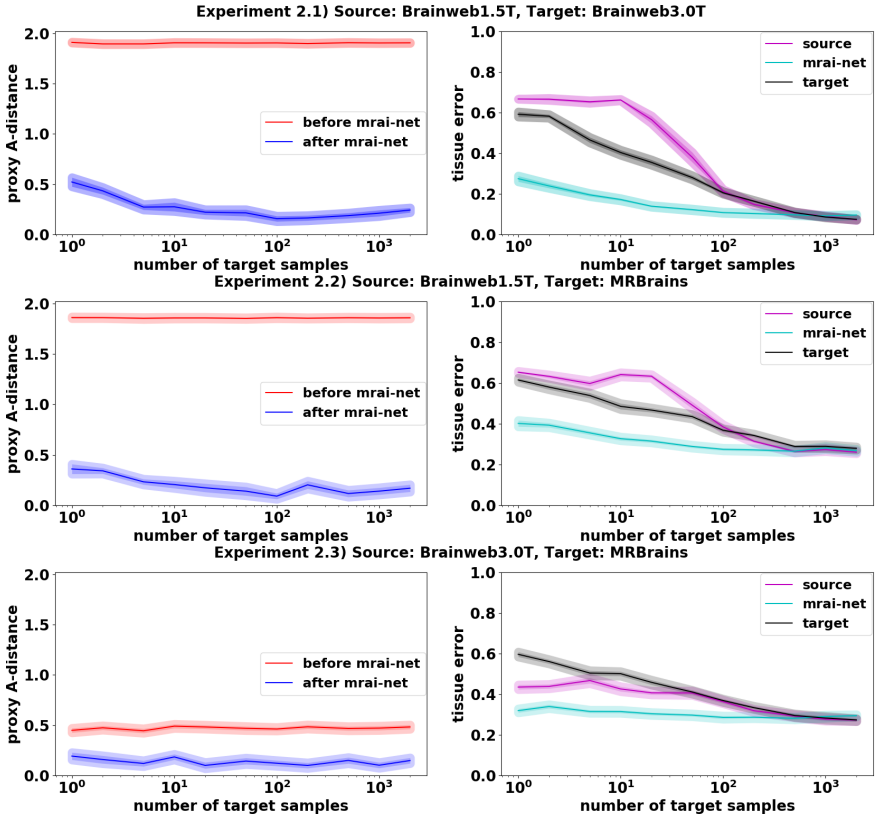
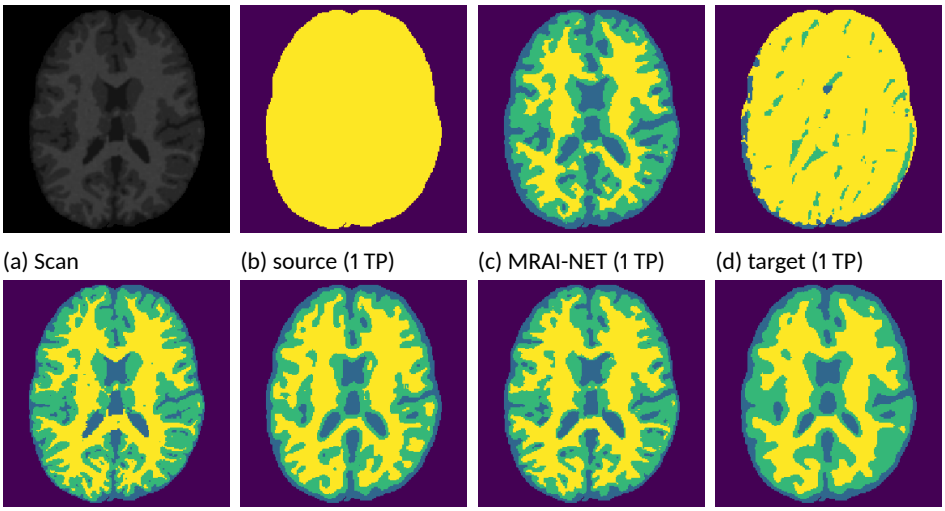


Figure 5.7: Graphs showing the effect of adding labeled samples from the target scanner for training the networks. (Left) Proxy A-distance between source and target scanner patches before (red) and after (blue) learning the mrai representation (smaller distance is more acquisition-invariance). (Right) Tissue classification error for the three classifiers source (supervised CNN trained on patches from source and target), MRAI-NET (supervised SVM trained on the source and target data mapped to MRAI-NET's representation) and target (supervised CNN trained on target patches). Note that when the proxy $\mathcal{A}$-distance between the source and target data before MRAI-NET is small (red line exp 2.3), the source data is representative of the target data (both 3T data), and the source tissue classifier (purple) shows better performance than using the target tissue classifier (cyan) with a small amount of target samples. However, if the proxy $\mathcal{A}$-distance is large (exp 2.1 and 2.2) before MRAI-NET (red line), the source tissue classifier (purple) shows worse performance than the target tissue classifier (cyan) with a small amount of target samples, since the source data (1.5T) is not representative of the target data (3T).

Figure 5.7 (left) shows the proxy $\mathcal{A}$-distance between the source and target samples for all three experiments. The proxy $\mathcal{A}$-distance for experiments 2.1 and 2.2 shows that in the original representation (raw; red line), the source and target distributions lie far apart (proxy $\mathcal{A}$-distance approaches 2). This illustrates the difference in acquisition protocol (1.5T versus 3.0T). After applying MRAI-NET (rep; blue line) the proxy $\mathcal{A}$-distance drops drastically (approaches 0) showing that the network managed to learn an MR-acquisition

invariant representation. Adding more target training samples improves the invariance up to about 100 samples, but the proxy $\mathcal{A}$-distance is already quite low after only using 1 target sample per tissue type for training. In experiment 2.3 the proxy $\mathcal{A}$-distance before applying MRAI-NET (raw) is already much lower than in the previous two experiments (around 0.5), this illustrates that the acquisition protocols are more similar to begin with (both 3.0T). The main difference between the distributions presumably results from simulated versus real data, since not all factors of acquisition variation are included in the simulations, most notably partial volume (0.96x0.96x3mm voxels in MRBrainS versus no partial volume in Brainweb). However, after applying MRAI-NET the proxy $\mathcal{A}$-distance is reduced further (approaches 0), again showing that MRAI-NET is able to learn an MR-acquisition invariant representation (rep) on this data, even for simulated and real data. Note that the MRBrainS data adds other modes of variation in terms of pathology and age in comparison to the Brainweb healthy adults, which could influence the tissue classification performance.



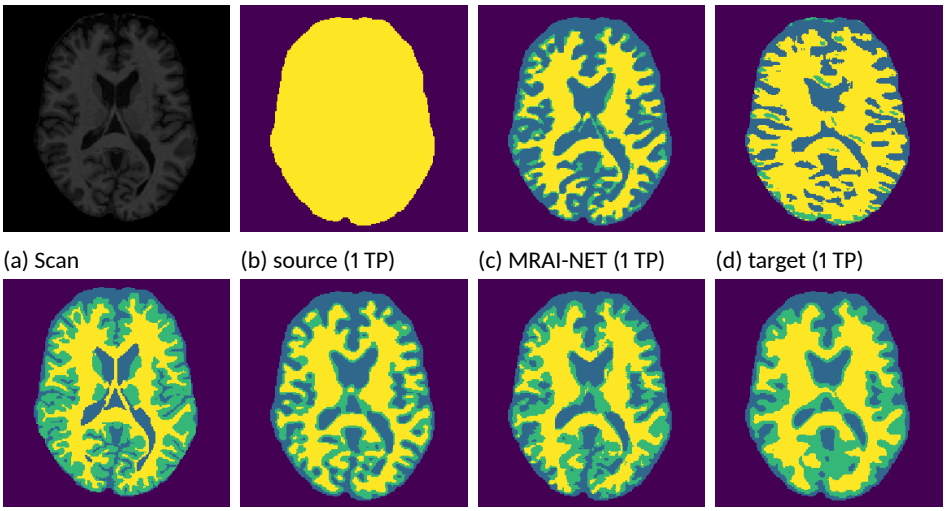
(a) Scan (b) source (1 TP) (c) MRAI-NET (1 TP) (d) target (1 TP)

(e) Ground truth (f) source (100 TPs) (g) MRAI-NET (100 TPs) (h) target (100 TPs)

Figure 5.8: Example brain tissue segmentations into white matter (yellow), gray matter (green) and cerebrospinal fluid (blue) for experiment 2.1 (Source: Brainweb1.5T, Target: Brainweb3.0T). A simulated MRI scan of a test subject from Brainweb3.0T (a) is shown, with corresponding ground truth segmentation (e), and the results of applying the source (b,f), target (d,h) and proposed MRAI-NET (c,g) classifiers, with either 1 or 100 target patches per tissue type used for training the classifiers (Figure 5.7).

Figure 5.7 (right) shows the tissue classification error for all three experiments. If the proxy $\mathcal{A}$-distance between the source and target distribution is high (experiment 2.1 and 2.2), and when using only one target sample per tissue, the source classifier that uses both the source data and target data for training shows worse performance than the one that uses only the target data (target); an error of 0.667 versus 0.591, respectively. Even when adding more target samples for training, the results show that it is more beneficial to train a supervised classifier on the target data alone, instead of on both the source and target data; using 10 target samples for training, source achieves an error of 0.662 versus an error

of 0.403 for target. The source classifier is focused on its source samples, which in this case are not informative of the target data. Given enough target samples, however, source starts to shift focus towards its target data and starts to match the performance of target: for 100 target samples, errors of 0.213 versus 0.205 respectively. If the proxy $\mathcal{A}$-distance between the source and target distributions is low (distributions are more similar; experiment 2.3), using the source data for training is beneficial; for 1 target sample per tissue source achieves an error of 0.435 and target an error of 0.596. In this case, the source samples are more representative of the target data and are aiding the classifier. In general, the MRAI-NET classifier outperforms both the source and target classifiers: an error of 0.269 for 1 sample, 0.175 for 10 samples and 0.111 for 100 samples. MRAI-NET's representation ensures that the source and target samples are more similar and that the source samples can be effectively used for training.



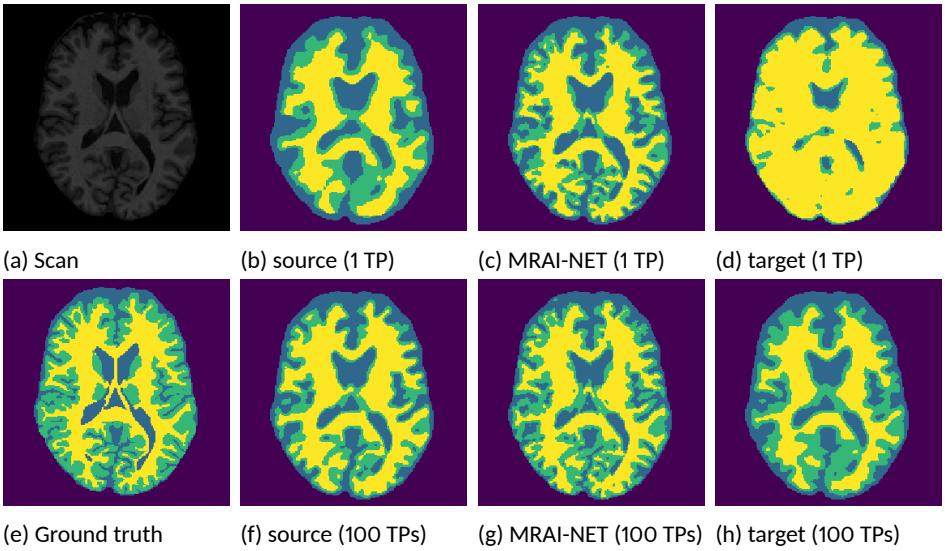
(a) Scan (b) source (1 TP) (c) MRAI-NET (1 TP) (d) target (1 TP)

(e) Ground truth (f) source (100 TPs) (g) MRAI-NET (100 TPs) (h) target (100 TPs)

Figure 5.9: Example brain tissue segmentations into white matter (yellow), gray matter (green) and cerebrospinal fluid (blue) for experiment 2.2 (Source: Brainweb1.5T, Target: MRBrainS). A simulated MRI scan of a test subject from MRBrainS (a) is shown, with corresponding ground truth segmentation (e), and the results of applying the source (b,f), target (d,h) and proposed MRAI-NET (c,g) classifiers, with either 1 or 100 target patches per tissue type used for training the classifiers (Figure 5.7).

Examples of the segmentation results on one of the target test images are shown in Figure 5.8 for experiment 2.1, Figure 5.9 for experiment 2.2, and Figure 5.10 for experiment 2.3. Examples are shown after using 1 target patch per tissue for training, and after using 100 target patches per tissue for training. The results show that only the MRAI-NET classifier is able to predict a segmentation that approaches the ground truth with only 1 target patch per tissue for training (error for experiment 2.1 = 0.269, experiment 2.2 = 0.403, experiment 2.3 = 0.320), while the source and target classifiers cannot (source error for experiment 2.1 = 0.667, experiment 2.2 = 0.653, experiment 2.3 = 0.435; target error for experiment 2.1: 0.591, experiment 2.2: 0.614, experiment 2.3 = 0.596). After using 100 patches the source

and target classifiers can predict a gross segmentation of WM, GM and CSF (source error for experiment 2.1 = 0.213, experiment 2.2 = 0.384, experiment 2.3 = 0.363; target error for experiment 2.1: 0.205, experiment 2.2: 0.368, experiment 2.3 = 0.368), but the MRAI-NET classifier prediction shows more details and a lower tissue classification error (error for experiment 2.1 = 0.111, experiment 2.2 = 0.276, experiment 2.3 = 0.284).



(a) Scan (b) source (1 TP) (c) MRAI-NET (1 TP) (d) target (1 TP)

(e) Ground truth (f) source (100 TPs) (g) MRAI-NET (100 TPs) (h) target (100 TPs)

Figure 5.10: Example brain tissue segmentations into white matter (yellow), gray matter (green) and cerebrospinal fluid (blue) for experiment 2.3 (Source: Brainweb3.0T, Target: MRBrainS). A simulated MRI scan of a test subject from MRBrainS (a) is shown, with corresponding ground truth segmentation (e), and the results of applying the source (b,f), target (d,h) and proposed MRAI-NET (c,g) classifiers, with either 1 or 100 target patches per tissue type used for training the classifiers (Figure 5.7.

5.4.6. Number of network parameters

Setting neural network hyperparameters, such as the number of convolution kernels to use, is always a tricky issue. The optimal parameter is different for each dataset, which means there are no easy defaults. In order to get some insight into the behavior of the network for different choices of hyperparameters, we performed an additional experiment. We used experiment 2.1's setting: Brainweb1.5T as source and Brainweb3.0T as target.

MRAI-NET has three layers with parameters: a convolution layer and two dense layers. We varied the number of kernels in the convolution layer and the number of nodes in the dense layers. We use the following sets of hyperparameters: [2 kernels, 4 nodes, 4 nodes], [4 kernels, 8 nodes, 4 nodes], [8 kernels, 16 nodes, 8 nodes], [16 kernels, 32 nodes, 16 nodes], [32 kernels, 64 nodes, 32 nodes] and [64 kernels, 128 nodes, 64 nodes] (i.e. the layer widths double each time). The total number of parameters are 322, 1254, 4874, 19218, 76322, and 304194, respectively. We used 10 labeled target patches per classes, from which we generated 18000 pairs of patches. The network was trained for 320 epochs and the experiment was repeated 20 times to obtain standard errors of the means. Figure

5.11 shows the results: the left figure looks at the proxy $\mathcal{A}$-distance as a function of the number of parameters and the right figure looks at the tissue classification error of a linear classifier trained on the resulting representation. For the proxy $\mathcal{A}$-distance, the graphs show a steady decrease in distance and then roughly levels off after [8, 16, 8]. This result indicates that an extremely wide MRAI-NET (i.e. [64, 128, 64]) will still be able to reduce acquisition variation. As for the tissue classification error, the thin network (i.e. [2, 4, 2]) starts out with a average error rate of 0.28 (underfitting) and drops immediately to 0.18 for [4, 8, 4]. Afterwards, it slowly increases to 0.19. This indicates that the network is not overfitting too drastically yet, which is probably due to the regularization (see Section 5.2.3). However, the graph does indicate that its error rate will go up if the number of parameters is increased further.

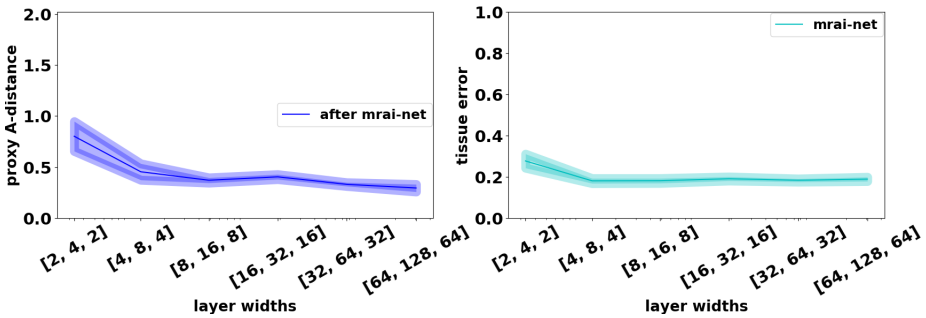


Figure 5.11: MRAI-NET's performance as a function of layer widths. (Left) The proxy $\mathcal{A}$-distance. (Right) The tissue classification error obtained through a linear classifier trained on data in MRAI-NET's representation. Both graphs show a slow gain in performance as the number of parameters grows.

5.4.7. Effect of the margin parameter

The margin parameter m in the dissimilar loss function, $\ell_{\text{dis}}(f|a,b) = \max(0, m - \|f(a) - f(b)\|_p)$, is important as it balances the actions of pushing and pulling between pairs. For small values, ℓ_{dis} will be much smaller than ℓ_{sim} and the network will focus on pulling pairs together. For large values, ℓ_{dis} will always be much larger than ℓ_{sim} and network will focus on pushing pairs apart. Figure 5.12 plots a synthetic data setting with the outcome of using three different values for the margin parameter. The left figure shows two synthetic 2-dimensional data sets, one with red versus blue crosses and the other with red versus blue squares. The right figures show validation samples fed through three networks with different values for the margin parameter. Firstly, the right top figure displays the result of using a margin parameter of 0: the network does not suffer *any* loss by making pairs of samples of different tissues too similar and consequently maps everything to a single point. Secondly, the right middle figure shows an appropriate choice for the margin, where the two data sets overlap and where red and blue points are separated. Lastly, the right bottom figure shows what happens when a large margin parameter is used: it focuses almost entirely on separating red versus blue and is not making the data sets more similar.

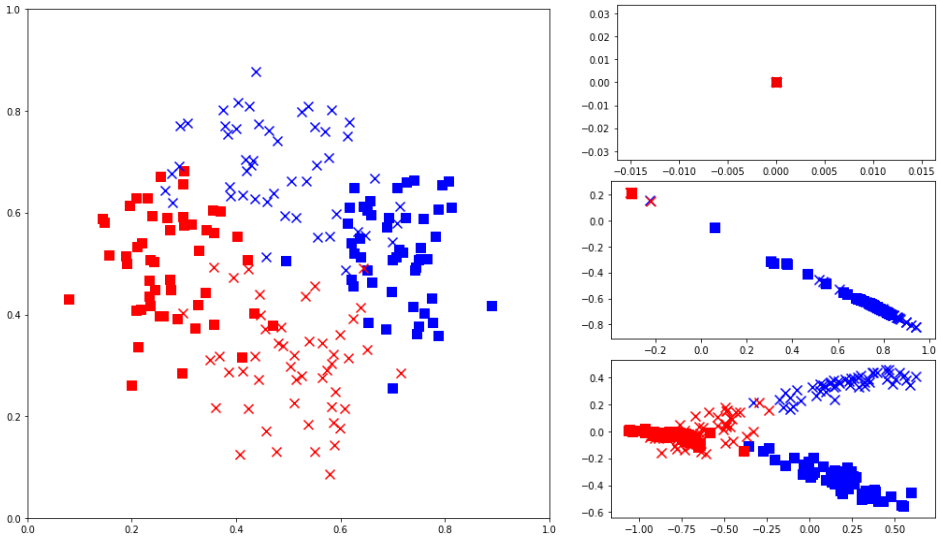


Figure 5.12: Effect of the margin hyperparameter. (Left) Two synthetic binary data sets, with markers indicating scanners and colors tissues. (Right) Representation found by a network with a margin of 0 (top), a margin of 1 (middle) and a margin of 10 (bottom).

Additionally, the optimal value for the margin parameter is affected by the number of similar versus dissimilar pairs. If there are twice as many similar pairs, then their loss will be twice as large as well and the network will focus more on pulling pairs together. Overall, the more similar pairs there are, the larger the margin parameter will need to be.

5.5. Discussion

We proposed a method to learn an MRI scanner acquisition-invariant representation that preserves the variation between brain tissues for segmentation. Once the representation is learned using MRAI-NET, any supervised classification model that uses feature vectors can be used to classify the brain tissues. The proposed method addresses the problem that the difference between scans acquired with two different MRI scanners or protocols can be so large that scans from one scanner are not representative of scans from another scanner. This difference does not affect assessment by human vision (e.g. radiologists can perform diagnostic work-up on both), but it does affect computer vision. To get insight into the difference between scans and to assess the performance of MRAI-NET to reduce this difference (achieve invariance), the proxy $\mathcal{A}$-distance measure between source and target patches was used. The experiments (Figure 5.7) show that this is a good measure to determine the difference between source and target acquisition, and might be used to predict classifier performance of a source classifier. Note that this measure does not require any tissue labels, and can thus be used as a general measure of distance between scanners. It merely requires source patches to be labeled as source, and target patches to be labeled as target. When the proxy $\mathcal{A}$-distance is low (Figure 5.7 bottom row) the source (source) classifier outperforms the target (target) classifier when a small number of target training

patches are used. When the proxy $\mathcal{A}$-distance is large the target classifier outperforms the source classifier, even when one target training patch per tissue is used. This suggests that if the proxy $\mathcal{A}$-distance is large (source data is not representative of target data), a source classifier trained on the source data should not be applied to the target data. Ground truth labels on the source data that are labor-intensive to acquire can in this case not be used for the target data. However, since MRAI-NET learns a representation that reduces the acquisition difference between source and target scanner the proxy $\mathcal{A}$-distance is drastically reduced. Therefore the MRAI-NET classifier outperforms both the source and target classifiers, when a small number of target training samples is available, and leverages the source ground truth labels.

Due to the complexity of the problem addressed, simulated data was used to provide a proof of principle. Ideal real data would require the same subject to be scanned on different scanners with different protocols, after which the scans should be manually segmented to obtain the ground truth for both scans. However, inter-observer variability would add an extra layer of variation. To test MRAI-NET on real data, the MRBrainS challenge data was used. Although, additional layers of variation include resolution, population and manual segmentation protocol, the experiments (Figure 5.7) show that MRAI-NET's performance on real data follows the same pattern as its performance on simulated data, be it with a higher classification error due to additional factors of variation.

A limitation of the proposed method is that learning an acquisition-invariant representation with MRAI-NET, will not necessarily work well on data sets with poor contrast between tissues. In that case, the network will both push and pull points in the overlap. Since these actions will mostly cancel each other out, the network will not be able to reduce acquisition-variation without sacrificing tissue variation, and vice versa.

Another limitation is that the proposed MRAI-NET requires at least 1 sample per tissue from the target scanner. This is not an unreasonable request, as it is not hard to find at least 1 patch per tissue (Section 5.4.4) in only one subject scanned with the target scanner. However, it may be possible to perform the similar/dissimilar labeling based on assumptions instead. For instance, if one assumes that the registration between two scans is accurate and that the subject-variation is not too large, then one could assume that target patches at certain locations are the same tissue as the source patches at these locations. Hence, those voxels could be used for the similarity-labeling process.
The proposed representation learning method could be used to reduce any type of variation, by adjusting the way that the similar and dissimilar pairs are defined. For example, registration, which can be viewed as variation in position, might be approached in a similar manner [37]. Key is to identify the forms of variation, determine which variation should be preserved and which should be reduced, and to find a way to label them as similar or dissimilar accordingly.

5.6. Conclusion

We addressed one of the major challenges of supervised voxel classification, i.e. generalization to data that is not representative of the training data. We provided a proof of principle for learning an MR acquisition invariant representation that reduces the variation between MRI scans acquired with different scanners or acquisition protocols, while preserving the variation between brain tissues. We showed that the proposed MRAI-NET is able to learn an MR acquisition invariant representation (low proxy $\mathcal{A}$-distance), and outperform supervised convolution neural networks trained on patches from the source or target scanners for tissue classification, when little target training patches are available. By reducing the acquisition related variation using MRAI-NET, the ground truth labels from the source data can be reused for the target data, since the source and target data are mapped to the same representation achieving generalization.

5.7. Appendix A

SIMRI requires NMR relaxation times for tissues based on particular magnetic static field strengths [25]. We performed a literature study for the T1 and T2 relaxation times, the results of which are listed in Table 5.3. The proton density values ρ stem from [38]. The 3.0T CSF parameters were interpolated using an exponential function fit ([39] justifies an exponential function based on physical properties). We equate connective tissue to glial matter (90% of the brain's connective tissue system is glial matter[5]).

Table 5.3: NMR relaxation times for brain tissue (IT'IS database).
[a] Glial matter values are unknown and are imputed with gray matter values.
[b] T2 values for cortical bone are actually T2* values (UTE seq).
[c] Equated to glial matter (see text).
[d] 3.0T T2 relaxation time is from dermis, other values are from hypodermis.

Tissue	ρ	T1(1.5T)	T2(1.5T)	T1(3.0T)	T2(3.0T)	Ref
CSF	100 (0)	4326 (0)	791 (127)	4313 (0)	503 (64)	[39–42]
GM	86 (.4)	1124 (24)	95 (8)	1820 (114)	99 (7)	[43]
WM	77 (3)	884 (50)	72 (4)	1084 (45)	69 (3)	[43]
Fat	100 (0)	343 (37)	58 (4)	382 (13)	68 (4)	[44]
Muscle	100 (0)	629 (50)	44 (6)	832 (62)	50 (4)	[43, 45]
Skin[d]	100 (0)	230 (8)	35 (4)	306 (18)	22 (0)	[45–47]
Skull[b]	0 (0)	200 (0)	.46 (0)	223 (11)	.39 (.02)	[48, 49]
Glial[a]	86 (0)	1124 (24)	95 (8)	1820 (114)	99 (7)	[40, 43]
Conn. [c]	77 (0)	1124 (24)	95 (8)	1820 (114)	99 (7)	[43]

[5] http://www.neuroplastix.com/styled-2/page139/styled-42/brainsconnectivetissue.html

5.8. Appendix B

In Section 5.2.1 we specified the Siamese loss as the networks objective function. The input of this loss consists of a pairwise distance, for which we chose an L^1-norm. There are 2 reasons for this: the first is that L^p-norms with larger values for p concentrate densely in high-dimensional spaces [50]. Concentration means that the differences between pairwise distances of a set of points become smaller as the number of dimensions increases. This is a problem because the actions of pulling and pushing will not sufficiently decrease the distance between similar pairs or sufficiently increase the distance between dissimilar pairs. The second reason is that the gradient of the L^1-norm is constant, while the gradient of an L^p-norms with $p > 1$ are functions of the distance [51]. Gradients of norms with large p's become smaller as the distance between pairs becomes smaller, which means the incentive for the network to pull pairs closer decreases. A constant gradient ensures that there will also be a constant incentive to pull similar pairs closer together. Considering that we want our representation to be truly invariant, we want the network to continue to pull similar pairs together until they are as close as possible.

References

[1] J. C. Bezdek, L. Hall, and L. Clarke, *Review of MR image segmentation techniques using pattern recognition*, Medical Physics **20**, 1033 (1993).

[2] A. P. Zijdenbos and B. M. Dawant, *Brain segmentation and white matter lesion detection in MR images*, Critical Reviews in Biomedical Engineering **22**, 401 (1994).

[3] L. Clarke, R. Velthuizen, M. Camacho, J. Heine, M. Vaidyanathan, L. Hall, R. Thatcher, and M. Silbiger, *MRI segmentation: methods and applications*, Magnetic Resonance Imaging **13**, 343 (1995).

[4] N. Saeed, *Magnetic resonance image segmentation using pattern recognition, and applied to image registration and quantitation*, NMR in Biomedicine **11**, 157 (1998).

[5] D. L. Pham, C. Xu, and J. L. Prince, *Current methods in medical image segmentation*, Annual Review of Biomedical Engineering **2**, 315 (2000).

[6] J. S. Suri, S. Singh, and L. Reden, *Computer vision and pattern recognition techniques for 2-D and 3-D MR cerebral cortical segmentation (part i): a state-of-the-art review*, Pattern Analysis & Applications **5**, 46 (2002).

[7] J. S. Duncan, X. Papademetris, J. Yang, M. Jackowski, X. Zeng, and L. H. Staib, *Geometric strategies for neuroanatomic analysis from MRI*, NeuroImage **23**, S34 (2004).

[8] M. A. Balafar, A. R. Ramli, M. I. Saripan, and S. Mashohor, *Review of brain MRI image segmentation methods*, Artificial Intelligence Review **33**, 261 (2010).

[9] A. Giorgio and N. De Stefano, *Clinical use of brain volumetry*, Journal of Magnetic Resonance Imaging **37**, 1 (2013).

[10] A. van Opbroek, M. Ikram, M. Vernooij, and M. De Bruijne, *Transfer learning improves supervised image segmentation across imaging protocols*, IEEE Transactions on Medical Imaging **34**, 1018 (2015).

[11] P. Moeskops, M. A. Viergever, A. M. Mendrik, L. S. de Vries, M. J. Benders, and I. Išgum, *Automatic segmentation of MR brain images with a convolutional neural network*, IEEE Transactions on Medical Imaging **35**, 1252 (2016).

[12] H. Chen, Q. Dou, L. Yu, J. Qin, and P.-A. Heng, *Voxresnet: deep voxelwise residual networks for brain segmentation from 3D MR images*, NeuroImage **2017** (2017).

[13] Y. Bengio, A. Courville, and P. Vincent, *Representation learning: a review and new perspectives*, IEEE Transactions on Pattern Analysis and Machine Intelligence **35**, 1798 (2013).

[14] A. van Opbroek, M. A. Ikram, M. W. Vernooij, and M. de Bruijne, *Supervised image segmentation across scanner protocols: a transfer learning approach*, in *Machine Learning in Medical Imaging* (Springer, 2012) pp. 160–167.

[15] V. Cheplygina, A. van Opbroek, M. A. Ikram, M. W. Vernooij, and M. de Bruijne, *Transfer learning by asymmetric image weighting for segmentation across scanners*, arXiv preprint arXiv:1703.04981 (2017).

[16] E. Tzeng, J. Hoffman, N. Zhang, K. Saenko, and T. Darrell, *Deep domain confusion: Maximizing for domain invariance*, arXiv preprint arXiv:1412.3474 (2014).

[17] Y. Ganin and V. Lempitsky, *Unsupervised domain adaptation by backpropagation*, in *International Conference on Machine Learning* (2015) pp. 1180–1189.

[18] Y. Ganin, E. Ustinova, H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, M. Marchand, and V. Lempitsky, *Domain-adversarial training of neural networks*, Journal of Machine Learning Research **17**, 1 (2016).

[19] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, *Generative adversarial nets*, in *Advances in Neural Information Processing Systems* (2014) pp. 2672–2680.

[20] K. Kamnitsas, C. Baumgartner, C. Ledig, V. Newcombe, J. Simpson, A. Kane, D. Menon, A. Nori, A. Criminisi, D. Rueckert, *et al.*, *Unsupervised domain adaptation in brain lesion segmentation with adversarial networks*, in *International Conference on Information Processing in Medical Imaging* (Springer, 2017) pp. 597–609.

[21] Y. Bengio *et al.*, *Learning deep architectures for AI*, Foundations and Trends ® in Machine Learning **2**, 1 (2009).

[22] J. Bromley, J. W. Bentz, L. Bottou, I. Guyon, Y. LeCun, C. Moore, E. Säckinger, and R. Shah, *Signature verification using a "Siamese" time delay neural network*, International Journal of Pattern Recognition and Artificial Intelligence **7**, 669 (1993).

[23] R. Hadsell, S. Chopra, and Y. LeCun, *Dimensionality reduction by learning an invariant mapping*, in *Conference on Computer Vision and Pattern Recognition*, Vol. 2 (IEEE, 2006) pp. 1735–1742.

[24] Y. LeCun, F. J. Huang, and L. Bottou, *Learning methods for generic object recognition with invariance to pose and lighting*, in *Conference on Computer Vision and Pattern Recognition*, Vol. 2 (2004) pp. II–97–104 Vol.2.

[25] H. Benoit-Cattin, G. Collewet, B. Belaroussi, H. Saint-Jalmes, and C. Odet, *The SIMRI project: a versatile and interactive MRI simulator*, Journal of Magnetic Resonance **173**, 97 (2005).

[26] B. Aubert-Broche, A. C. Evans, and L. Collins, *A new improved version of the realistic digital brain phantom*, NeuroImage **32**, 138 (2006).

[27] B. Aubert-Broche, M. Griffin, G. B. Pike, A. C. Evans, and D. L. Collins, *Twenty new digital brain phantoms for creation of validation image data bases*, IEEE Transactions on Medical Imaging **25**, 1410 (2006).

[28] A. M. Mendrik, K. L. Vincken, H. J. Kuijf, M. Breeuwer, W. H. Bouvy, J. De Bresser, A. Alansary, M. De Bruijne, A. Carass, A. El-Baz, *et al.*, *MRBrainS challenge: online evaluation framework for brain image segmentation in 3T MRI scans*, Computational Intelligence and Neuroscience (2015).

[29] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, M. Kudlur, J. Levenberg, R. Monga, S. Moore, D. G. Murray, B. Steiner, P. Tucker, V. Vasudevan, P. Warden, M. Wicke, Y. Yu, and X. Zheng, *Tensorflow: A system for large-scale machine learning*, in *USENIX Symposium on Operating Systems Design and Implementation* (2016) pp. 265–283.

[30] F. Chollet *et al.*, *Keras*, (2015).

[31] L. Bottou, *Stochastic gradient descent tricks*, in *Neural networks: tricks of the trade* (Springer, 2012) pp. 421–436.

[32] D. Kifer, S. Ben-David, and J. Gehrke, *Detecting change in data streams*, in *International Conference on Very Large Data Bases* (VLDB Endowment, 2004) pp. 180–191.

[33] S. Ben-David, J. Blitzer, K. Crammer, and F. Pereira, *Analysis of representations for domain adaptation*, in *Advances in Neural Information Processing Systems* (2007) pp. 137–144.

[34] S. Ben-David, J. Blitzer, K. Crammer, A. Kulesza, F. Pereira, and J. W. Vaughan, *A theory of learning from different domains*, Machine Learning **79**, 151 (2010).

[35] D. L. Collins, A. P. Zijdenbos, V. Kollokian, J. G. Sled, N. J. Kabani, C. J. Holmes, and A. C. Evans, *Design and construction of a realistic digital brain phantom*, IEEE Transactions on Medical Imaging **17**, 463 (1998).

[36] M. A. Ikram, A. van der Lugt, W. J. Niessen, P. J. Koudstaal, G. P. Krestin, A. Hofman, D. Bos, and M. W. Vernooij, *The rotterdam scan study: design update 2016 and main findings*, European Journal of Epidemiology **30**, 1299 (2015).

[37] M. Simonovsky, B. Gutiérrez-Becker, D. Mateus, N. Navab, and N. Komodakis, *A deep metric for multimodal registration*, in *International Conference on Medical Image Computing and Computer-Assisted Intervention* (Springer, 2016) pp. 10–18.

[38] D. Yoder, E. Changchien, C. B. Paschal, and J. M. Fitzpatrick, *MRI simulator with static field inhomogeneity*, in *Medical Imaging* (International Society for Optics and Photonics, 2002) pp. 592–603.

[39] R. M. Kroeker and R. M. Henkelman, *Analysis of biological NMR relaxation data with continuous distributions of relaxation times*, Journal of Magnetic Resonance **69**, 218 (1986).

[40] K. H. Cheng, *In vivo tissue characterization of human brain by chi-squares parameter maps: multiparameter proton T2-relaxation analysis*, Magnetic Resonance Imaging **12**, 1099 (1994).

[41] W. D. Rooney, G. Johnson, X. Li, E. R. Cohen, S.-G. Kim, K. Ugurbil, and C. S. Springer, *Magnetic field and tissue dependencies of human brain longitudinal 1H2O relaxation in vivo*, Magnetic Resonance in Medicine **57**, 308 (2007).

[42] S. Piechnik, J. Evans, L. Bary, R. G. Wise, and P. Jezzard, *Functional changes in CSF volume estimated using measurement of water T2 relaxation*, Magnetic Resonance in Medicine **61**, 579 (2009).

[43] G. J. Stanisz, E. E. Odrobina, J. Pun, M. Escaravage, S. J. Graham, M. J. Bronskill, and R. M. Henkelman, *T1, T2 relaxation and magnetization transfer in tissue at 3T*, Magnetic Resonance in Medicine **54**, 507 (2005).

[44] C. M. De Bazelaire, G. D. Duhamel, N. M. Rofsky, and D. C. Alsop, *Mr imaging relaxation times of abdominal and pelvic tissues measured in vivo at 3.0T: preliminary results*, Radiology **230**, 652 (2004).

[45] J. K. Barral, N. Stikov, E. Gudmundson, P. Stoica, and D. G. Nishimura, *Skin T1 mapping at 1.5T, 3T, and 7T*, in *Annual Meeting of ISMRM* (2009) p. 4451.

[46] S. Richard, B. Querleux, J. Bittoun, I. Idy-Peretti, O. Jolivet, E. Cermakova, and J.-L. Lévêque, *In vivo proton relaxation times analysis of the skin layers by magnetic resonance imaging*, Journal of Investigative Dermatology **97**, 120 (1991).

[47] H. K. Song, F. W. Wehrli, and J. Ma, *In vivo MR microscopy of the human skin*, Magnetic Resonance in Medicine **37**, 185 (1997).

[48] I. L. Reichert, M. D. Robson, P. D. Gatehouse, T. He, K. E. Chappell, J. Holmes, S. Girgis, and G. M. Bydder, *Magnetic resonance imaging of cortical bone with ultrashort TE pulse sequences*, Magnetic Resonance Imaging **23**, 611 (2005).

[49] J. Du, M. Carl, M. Bydder, A. Takahashi, C. B. Chung, and G. M. Bydder, *Qualitative and quantitative ultrashort echo time (UTE) imaging of cortical bone*, Journal of Magnetic Resonance **207**, 304 (2010).

[50] A. Flexer and D. Schnitzer, *Choosing ℓ^p norms in high-dimensional spaces based on hub analysis*, Neurocomputing **169**, 281 (2015).

[51] S. Boyd and L. Vandenberghe, *Convex optimization* (Cambridge University Press, 2004).

6

Robust adaptation

In domain adaptation, classifiers with information from a source domain adapt to generalize to a target domain. However, an adaptive classifier can perform worse than a non-adaptive classifier due to invalid assumptions, increased sensitivity to estimation errors or model misspecification. Our goal is to develop a domain-adaptive classifier that is robust in the sense that it does not rely on restrictive assumptions on how the source and target domains relate to each other and that it does not perform worse than the non-adaptive classifier. We formulate a conservative parameter estimator that only deviates from the source classifier when a lower risk is guaranteed for all possible labellings of the given target samples. We derive the classical least-squares and discriminant analysis cases and show that these perform on par with state-of-the-art domain adaptive classifiers in sample selection bias settings, while outperforming them in more general domain adaptation settings.

This chapter is based on the paper ”Target contrastive pessimistic risk for robust domain adaptation”.

6.1. Introduction

Generalization in supervised learning relies on the fact that future samples should originate from the same underlying distribution as the ones used for training. However, this is not the case in settings where data is collected from different locations, different measurement instruments are used or there is only access to biased data. In these situations the labeled data does not represent the distribution of interest. This problem setting is referred to as a *domain adaptation* setting, where the distribution of the labeled data is called the *source domain* and the distribution that one is actually interested in is called the *target domain*. Most often, data in the target domain is not labeled and adapting a source domain classifier, i.e. changing its predictions to be more suited to the target domain, is the only means by which one can make predictions for the target domain. Unfortunately, depending on the domain dissimilarity, adaptive classifiers can perform *worse* than non-adaptive ones. In this work, we formulate a conservative adaptive classifier that always performs at least as well as the non-adaptive one.

Biased samplings tend to occur when one samples locally from a much larger population [1, 2]. For instance, in computer-assisted diagnosis, biometrics collected from two different hospitals will be different due to differences between the patient populations: ones diet might not be the same as the others. Nonetheless, both patient populations are subsamples of the human population as a whole. Adaptation in this example corresponds to accounting for the differences between patient populations, training a classifier on the corrected labeled data from one hospital, and applying the adapted classifier to the other hospital. Additionally, different measurement instruments cause different biased samplings: photos of the same object taken with different cameras lead to different distributions over images [3]. Lastly, biases arise when one only has access to particular subsets, such as data from individual humans in a activity recognition task [4].

In the general setting, domains can be arbitrarily different and contain almost no mutual information, which means generalization will be extremely difficult. However, there are cases where the problem setting is more structured: in the *covariate shift* setting, the marginal data distributions differ but the class-posterior distributions are equal [5–7]. This means that the underlying true classification function is the same in both domains, implying that a correctly specified adaptive classifier converges to the same solution as the target classifier. Adaptation occurs by weighing each source sample by how important it is under the target distribution and training on the importance-weighed labeled source data. A model that relies on equal class-posterior distributions can perform very well when its assumption is true, but it can deviate in detrimental ways when its assumption is false.

Considering their potential, a number of papers have looked at conditions and assumptions that allow for successful adaptation. A particular robust one specifies the existence of a common latent embedding, represented by a set of *transfer components* [8]. After mapping data onto these components, one can train and test standard classifiers again. Other possible assumptions include low-data-divergence [9–11], low-error joint prediction [10, 11], the existence of a domain manifold [12–14], restrictions to subspace transformations [15],

conditional independence of class and target given source data [16] and unconfoundedness [17]. The more restrictive an assumption is, the worse the classifier tends to perform when it is invalid. One of the strengths of the estimator that we develop here is that it does not require making any assumptions on the relationship between the domains.

The domain adaptation and covariate shift settings are very similar to the sample selection bias setting in the statistics and econometrics communities [7, 18, 19]. There, the bias is explicitly modeled as a variable that denotes how likely it is for a particular sample to be selected for the training set. One hopes to generalize to an unbiased sample, i.e., the case where each sample is equally likely to be selected. As such, this setting can also be viewed as a case of domain adaptation, with the biased sample set as the source domain and the unbiased sample set as the target domain. In this case, there is even additional information: the support of the source domain will be contained in the support of the target domain. This information can be exploited, as some methods rely on a non-zero target probability for every source sample [6, 20]. Lastly, the causal inference community has also considered causes for differing training and testing distributions, including how to estimate and control for these differences [2, 21, 22].

Although not often discussed, a variety of papers have reported adaptive classifiers that, at times, perform worse than the non-adaptive source classifier [6, 13, 16, 23–25]. On closer inspection, this tends to happen when a classifier with a particular assumption is deployed in a problem setting for which this assumption is not valid. For example, if the assumption of a common latent representation does not hold or when the domains are too dissimilar to recover the transfer components, then mapping both source and target data onto the found transfer components will result in mixing of the class-conditional distributions [8]. Additionally, one of the most popular covariate shift approaches, kernel mean matching (KMM), assumes that the support of the target distribution is contained in the support of the source distribution [20, 26]. When this is not the case, the resulting estimated weights can become very bimodal: a few samples are given very large weights and all other samples are given near-zero weights. This greatly reduces the effective sample size for the subsequent classifier [27].

Since the validity of the aforementioned assumptions are difficult, if not impossible, to check, it is of interest to design an adaptive classifier that is at least guaranteed to perform as well as the non-adaptive one. Such a property is often framed as a minimax optimization problem in statistics, econometrics and game theory [28]. Wen et al. constructed a minimax estimator for the covariate shift setting: Robust Covariate Shift Adjustment (RCSA) [29] accounts for estimation errors in the importance weights by considering their worst-case configuration. However, this can sometimes be too conservative, as the worst-case weights can be very disruptive to the subsequent classifier optimization. Another minimax strategy, dubbed the Robust Bias-Aware (RBA) classifier [25], plays a game between a risk minimizing target classifier and a risk maximizing target class-posterior distribution, where the adversary is constrained to pick posteriors that match the moments of the source distribution statistics. This constraint is important, as the adversary would otherwise be able

to design posterior probabilities that result in degenerate classifiers (e.g. assign all class-posterior probabilities to 1 for one class and 0 for the other). However, it also means that their approach loses predictive power in areas of feature space where the source distribution has limited support, and thus is not suited very well for problems where the domains are very different.

The main contribution of this chapter is that we provide an empirical risk minimization framework to train a classifier that will always perform at least as well as the naive source classifier. Furthermore, we show that a discriminant analysis model derived from our framework will *always be likelier* than the naive source model. To the best of our knowledge, strict improvements have not been shown before.

The chapter continues as follows: section 6.2 presents the motivation and general formulation of our method, with the specific case of a least-squares classifier in section 6.3 and the specific case of a discriminant analysis classifier in section 6.4. Sections 6.5.2 and 6.5.3 show experiments on sample selection bias problems and general domain adaptation problems, respectively, and we conclude with discussing some limitations and implications in section 6.6.

6.2. Target contrastive pessimistic risk

This section starts with the problem definition, followed by our risk formulation.

6.2.1. Problem definition

Given a D-dimensional input space $\mathcal{X} \subseteq \mathbb{R}^D$ and a class space $\mathcal{Y} = \{1, \dots, K\}$ with K as the number of classes, a *domain* refers to a particular joint probability distribution over this pair of spaces. One is called the *source* domain, for which labels are available, and the other is called the *target* domain, for which no labels are available. Let $\mathcal{S}$ mark the source domain, with n samples drawn from the source domain's joint distribution, $p_{\mathcal{S}}(x, y)$, referred to as $\{(x_i, y_i)\}_{i=1}^n$. Similarly, let $\mathcal{T}$ mark the target domain, with m samples drawn from the target domain's joint distribution, $p_{\mathcal{T}}(x, y)$, referred to as $\{(z_j, u_j)\}_{j=1}^m$. Note that both domains are defined over the same input space, which implies that x and z are represented in the same D-dimensional feature space. The target labels u are unknown at training time and the goal is to predict them, using only the given unlabeled target samples $\{z_j\}_j^m$ and the given labeled source samples $\{(x_i, y_i)\}_i^n$.

6.2.2. Target risk

The risk minimization framework formalizes *risk*, or the expected loss ℓ incurred by classification function h, mapping input to classes $h : \mathcal{X} \to \mathcal{Y}$, with respect to a particular joint labeled data distribution; $R(h) = \mathbb{E}[\ell(h \mid x, y)]$. By minimizing empirical risk, i.e. the approximation using the sample average, with respect to classifiers from a space of hypothetical classification functions $\mathcal{H}$, one hopes to find the function that generalizes most to novel samples. Additionally, a regularization term that punishes classifier complexity is often incorporated to avoid finding classifiers that are too specific to the given labeled

data. For a given joint distribution, the choice of loss function, the hypothesis space and amount of regularization largely determine the behavior of the resulting classifier.

The empirical risk in the source domain can be computed as follows:

$$\hat{R}\left(h \mid x, y\right) = \frac{1}{n} \sum_{i=1}^{n} \ell\left(h \mid x_i, y_i\right) ,$$

with the *source classifier* being the classifier that is found by minimizing this risk:

$$\hat{h}^{\mathcal{S}} = \arg\min_{h \in \mathcal{H}} \hat{R}\left(h \mid x, y\right) . \tag{6.1}$$

Since the source classifier does not incorporate any part of the target domain, it is essentially entirely naive of it. But, if we assume that the domains are related in some way, then it makes sense to apply the source classifier on the target data. To evaluate $\hat{h}^{\mathcal{S}}$ in the target domain, the empirical *target risk*, i.e. the risk of the classifier with respect to target samples, is computed:

$$\hat{R}\big(\hat{h}^{\mathcal{S}} \mid z, u\big) = \frac{1}{m} \sum_{j=1}^{m} \ell\big(\hat{h}^{\mathcal{S}} \mid z_j, u_j\big) . \tag{6.2}$$

Training on the source domain and testing on the target domain is our baseline, non-adaptive approach.

Although the source classifier does not incorporate information from the target domain nor any knowledge on the relation between the domains, it is often *not the worst* classifier (especially in cases where the domains are very similar). In cases where approaches rely heavily on assumptions, the adaptive classifiers can deviate from the source classifier in ways that lead to even larger target risks. Considering that these assumptions cannot be checked for validity, there are no guarantees that these adaptive classifiers outperform the source classifier. Essentially, they are not *safe* to use.

6.2.3. Contrast

We are interested in finding a classifier that is never worse than the source classifier. We formalize this desire by subtracting the source classifiers target risk in (6.2) from the target risk of a different classifier h:

$$\hat{R}\big(h \mid z, u\big) \; - \; \hat{R}\big(\hat{h}^{\mathcal{S}} \mid z, u\big) \tag{6.3}$$

If such a contrast is used as a risk minimization objective, i.e. $\min_{h \in \mathcal{H}} \hat{R}(h|z,u) - \hat{R}(\hat{h}^{\mathcal{S}}|z,u)$, then the risk of the resulting classifier is bounded above by the risk of the source classifier: the maximal value of the contrast is 0, which occurs when the same classifier is found, $h = \hat{h}^{\mathcal{S}}$. Classifiers that lead to larger target risks are not valid solutions to the minimization problem, which implies that certain parts of the hypothesis space $\mathcal{H}$ will never be reached. As such, the contrast implicitly constrains $\mathcal{H}$ in a similar way as projection estimators [30].

6.2.4. Pessimism

However, (6.3) still incorporates the target labels u, which are unknown. Taking a conservative approach, we use a worst-case labeling instead, achieved by *maximizing* risk with respect to a hypothetical labeling q. For any classifier h, the risk with respect to this worst-case labeling will always be larger than the risk with respect to the true target labeling:

$$\hat{R}\left(h \mid z, u\right) \leq \max_{q} \hat{R}\left(h \mid z, q\right) . \tag{6.4}$$

Unfortunately, maximizing over a set of discrete labels is a combinatorial problem and is computationally very expensive. To avoid this expense, we represent the hypothetical labeling probabilistically, $q_{jk} := p(u_j = k \mid z_j)$, sometimes also referred to as a *soft* label [31]. This means that q_j is means that q_j is a vector of K elements that sum to 1, represented as an element of a $K-1$ simplex, Δ_{K-1}. For m samples, an m-dimensional $K-1$ simplex Δ_{K-1}^{m} is taken. Note that known labels can also be represented probabilistically, for example $y_i = 1 \Leftrightarrow p(y_i = 1 \mid x_i) = 1,\ p(y_i \neq 1 \mid x_i) = 0$. Hence, in practice, both y_i and u_j are represented as 1 by K vectors with the k-th element marking the probability that sample i or j belongs to class k.

6.2.5. Contrastive pessimistic risk

Joining the contrastive target risk from (6.3) with the pessimistic labeling q from (6.4) forms the following risk function:

$$\hat{R}^{\text{TCP}}\big(h \mid \hat{h}^{\mathcal{S}}, z, q\big) = \frac{1}{m} \sum_{j=1}^{m} \ell\big(h \mid z_j, q_j\big) - \ell\big(\hat{h}^{\mathcal{S}} \mid z_j, q_j\big) . \tag{6.5}$$

We refer to the risk in equation 6.5 as the Target Contrastive Pessimistic risk (TCP). Minimizing it with respect to a classifier h and maximizing it with respect to the hypothetical labeling q, leads to the new TCP target classifier:

$$\hat{h}^{\mathcal{T}} = \operatorname*{arg\,min}_{h \in \mathcal{H}} \max_{q \in \Delta_{K-1}^{m}} \hat{R}^{\text{TCP}}\big(h \mid \hat{h}^{\mathcal{S}}, z, q\big) . \tag{6.6}$$

Note that the TCP risk only considers the performance on the target domain. It is different from the risk formulations in [25] and [29], because those incorporate the classifiers performance on the source domain as well. Our formulation contains no evaluation on the source domain, and focuses solely on the performance gain we can achieve in the target domain with respect to the source classifier.

6.2.6. Optimization

If the loss function ℓ is restricted to be globally convex and the hypothesis space $\mathcal{H}$ is a convex set, then the TCP risk with respect to h will be globally convex and there will be a unique optimum with respect to h. The TCP risk with respect to q is linear and bounded due to the simplex, which means that it is possible that the optimum is not unique. However, the combined minimax objective function is globally convex-linear. This is important, because it guarantees the existence of a saddle point, i.e. an optimum with respect to both h and q [32].

Finding the saddle point can be done through first performing a gradient descent step according to the partial derivative with respect to h, followed by a gradient ascent step according to the partial derivative with respect to q. However, this last step causes the updated q to leave the simplex. In order to enact the constraint, it is projected back onto the simplex after performing the gradient step. This projection, $\mathcal{P}$, maps the point outside the simplex, a, to the point, b, that is the closes point on the simplex in terms of Euclidean distance: $\mathcal{P}(a) = \arg\min_{b \in \Delta} \|a - b\|_2$ [33, 34]. Unfortunately, the projection step complicates the computation of the step size, which we replace by a learning rate α^t, decreasing over iterations t. This results in the overall update: $q^{t+1} \leftarrow \mathcal{P}(q^t + \alpha^t \nabla q^t)$. Note that the projection step is linear, which means the overall update for q remains linear.

A gradient descent - gradient ascent procedure for globally convex-linear objectives is guaranteed to converge to the saddle point (c.f. proposition 4.4 and corollary 4.5 of [32]).

6.3. Least-squares

Discriminative classification models make no assumptions on the data distributions and directly optimize predictions. We incorporate a discriminative model in the TCP risk through the least-squares classifier, which is defined by a quadratic loss function $\ell_{\text{LS}}(h \mid x_i, y_i) = (h(x_i) - y_i)^2$ [35]. For multi-class classification, we employ a one-vs-all scheme [36].

Furthermore, we chose a linear hypothesis space, $h(z) = \arg\max_{k \in \mathcal{Y}} \sum_d^D z_d \theta_{dk} + \theta_{0k}$, which we will denote as the inner product $z\theta_k$ between the data row vector, implicitly augmented with a constant 1, and the classifier parameter vector. θ is an element of a $(D+1) \times K$-dimensional parameter space Θ and in the following, we will refer to the classifier optimization step, i.e. minimization over $h \in \mathcal{H}$, as a parameter estimation step, i.e. a minimization over $\theta \in \Theta$. In summary, the least-squares loss of a sample is:

$$\ell_{\text{LS}}(\theta \mid z_j, q_j) = \sum_{k=1}^{K} \left(z_j \theta_k - q_{jk}\right)^2 . \tag{6.7}$$

Plugging (6.7) into (6.5), the TCP-LS risk is defined as:

$$\begin{aligned} \hat{R}_{\text{LS}}^{\text{TCP}}\left(\theta \mid \hat{\theta}^{\mathcal{S}}, z, q\right) &= \frac{1}{m} \sum_{j=1}^{m} \ell_{\text{LS}}\left(\theta \mid z_j, q_j\right) - \ell_{\text{LS}}\left(\hat{\theta}^{\mathcal{S}} \mid z_j, q_j\right) \\ &= \frac{1}{m} \sum_{j=1}^{m} \sum_{k=1}^{K} \left(z_j \theta_k - q_{jk}\right)^2 - \left(z_j \hat{\theta}_k^{\mathcal{S}} - q_{jk}\right)^2 , \end{aligned}$$

with the resulting estimate:

$$\hat{\theta}_{\text{LS}}^{\mathcal{T}} = \arg\min_{\theta \in \Theta} \max_{q \in \Delta_{K-1}^m} \hat{R}_{\text{LS}}^{\text{TCP}}(\theta \mid \hat{\theta}^{\mathcal{S}}, z, q) . \tag{6.8}$$

For fixed q, the minimization over θ has a closed form solution. For each class, the parameter vector is:

$$\begin{aligned}\frac{\partial}{\partial \theta_k} \hat{R}_{\text{LS}}^{\text{TCP}}(\theta \mid \hat{\theta}^{\mathcal{S}}, z, q) &= 0 \\ \frac{1}{m}\sum_{j=1}^{m} 2\, z_j^\top (z_j \theta_k - q_{jk}) &= 0 \\ \theta_k &= \Big(\sum_{j=1}^{m} z_j^\top z_j\Big)^{-1}\Big(\sum_{j=1}^{m} z_j^\top q_{jk}\Big).\end{aligned}$$

Keeping θ fixed, the gradient with respect to q_{jk} is linear:

$$\begin{aligned}\frac{\partial}{\partial q_{jk}} \hat{R}_{\text{LS}}^{\text{TCP}}(\theta \mid \hat{\theta}^{\mathcal{S}}, z, q) &= \frac{-2}{m}(z_j \theta_k - q_{jk}) - \frac{-2}{m}(z_j \hat{\theta}_k^{\mathcal{S}} - q_{jk}) \\ &= \frac{-2}{m} z_j \big(\theta_k - \hat{\theta}_k^{\mathcal{S}}\big).\end{aligned}$$

Note that the maximization over q is essentially driving the two sets of parameters apart. See Algorithm 3 for pseudo-code for TCP-LS.

Algorithm 3 TCP-LS

Input: source data x (size $n \times D$), labels y (size $n \times K$), target data z (size $m \times D$), learning rate α, convergence criterion ϵ.
Output: $\hat{\theta}_{\text{LS}}^{T} = (\theta_1, \ldots, \theta_K)$
for all classes **do**
  $\hat{\theta}_k^{\mathcal{S}} = \Big(\sum_i^n x_i^\top x_i\Big)^{-1}\Big(\sum_i^n x_i^\top y_{ik}\Big)$
end for
$t = 0$
$\theta_k^t = \hat{\theta}_k^{\mathcal{S}} \qquad \forall k$
$q_{jk}^t \leftarrow 1/K \qquad \forall j, k$
repeat
  for all classes **do**
    $\theta_k^{t+1} = \big(\sum_j^m z_j^\top z_j\big)^{-1}\big(\sum_j^m z_j^\top q_{jk}^t\big)$
    for all samples **do**
      $\nabla q_{jk} = -2 z_j \big(\theta_k^{t+1} - \hat{\theta}_k^{\mathcal{S}}\big)/m$
    end for
  end for
  $q^{t+1} \leftarrow \mathcal{P}\big(q^t - \alpha^t \nabla q\big)$
  $\alpha^{t+1} \leftarrow \alpha/t$
  $t \leftarrow t + 1$
until $\| \hat{R}_{\text{LS}}^{\text{TCP}}\big(\theta^{t+1} \mid \hat{\theta}^{\mathcal{S}}, z, q^{t+1}\big) - \hat{R}_{\text{LS}}^{\text{TCP}}\big(\theta^{t} \mid \hat{\theta}^{\mathcal{S}}, z, q^{t}\big) \| \leq \epsilon$

6.4. Discriminant analysis

As a generative classification model for the TCP risk, we chose the discriminant analysis model (DA). It fits a Gaussian distribution to each class, proportional to the class prior:

$$\mathcal{N}(x \mid \theta_k) = \pi_k \frac{1}{\sqrt{(2\Pi)^D |\Sigma_k|}} \exp(-\frac{1}{2}(x-\mu_k)\Sigma_k^{-1}(x-\mu_k)^\top),$$

where θ_k consists of the prior, mean and covariance matrix for the k-th class; $\theta_k = (\pi_k, \mu_k, \Sigma_k)$, $|\cdot|$ marks the determinant and the capital Π refers to the number. New samples x^* are classified according to maximum probability: $h(x^*) = \arg\max_{k \in \mathcal{Y}} \mathcal{N}(x^* \mid \theta_k)$. Each label is encoded as a vector, e.g. for $\mathcal{Y} = \{1,2,3\}, y_i = 2 \Leftrightarrow y_i = [0,1,0]$. The model is incorporated in the empirical risk minimization formulation by taking the negative log-likelihoods as the loss function:

$$\ell(\theta \mid x, y) = \sum_k^K -y_k \log \mathcal{N}(x \mid \theta_k).$$

6.4.1. Quadratic discriminant analysis

If one Gaussian distribution is fitted to each class separately, the resulting classifier is a quadratic function of the difference in means and covariances, and is hence called *quadratic discriminant analysis* (QDA). For target data z and soft labels q, the loss is formulated as:

$$\ell_{\text{QDA}}(\theta \mid z_j, q_j) = \sum_{k=1}^K -q_{jk} \log \mathcal{N}(z_j \mid \theta_k). \tag{6.9}$$

Plugging the loss from (6.9) into (6.5), the TCP-QDA risk becomes:

$$\begin{aligned}\hat{R}^{\text{TCP}}_{\text{QDA}}(\theta \mid \hat{\theta}^S, z, q) &= \frac{1}{m}\sum_{j=1}^m \ell_{\text{QDA}}(\theta \mid z_j, q_j) - \ell_{\text{QDA}}(\hat{\theta}^S \mid z_j, q_j) \\ &= \frac{1}{m}\sum_{j=1}^m \sum_{k=1}^K -q_{jk} \log \frac{\mathcal{N}(z_j \mid \theta_k)}{\mathcal{N}(z_j \mid \hat{\theta}^S_k)},\end{aligned} \tag{6.10}$$

where the estimate itself is:

$$\hat{\theta}^T_{\text{QDA}} = \arg\min_{\theta \in \Theta} \max_{q \in \Delta^m_{K-1}} \hat{R}^{\text{TCP}}_{\text{QDA}}(\theta \mid \hat{\theta}^S, z, q).$$

Minimization with respect to θ also has a closed-form solution for discriminant analysis models. For each class, the parameter estimates are:

$$\pi_k = \frac{1}{m}\sum_{j=1}^{m} q_{jk}\,,$$

$$\mu_k = \Big(\sum_{j=1}^{m} q_{jk}\Big)^{-1}\sum_{j=1}^{m} q_{jk} z_j\,,$$

$$\Sigma_k = \Big(\sum_{j=1}^{m} q_{jk}\Big)^{-1}\sum_{j=1}^{m} q_{jk}(z_j - \mu_k)^\intercal (z_j - \mu_k)\,.$$

One of the properties of a discriminant analysis model is that it requires the estimated covariance matrix Σ_k to be non-singular. It is possible for the maximizer over q in TCP-QDA to assign less samples than dimensions to one of the classes, causing the covariance matrix for that class to be singular. To prevent this, we regularize its estimation by first restricting Σ_k to minimal eigenvalues of 0 and then adding a scalar multiple of the identity matrix λI. Essentially, this constrains the estimated covariance matrix to a minimum size in each direction.

Keeping θ fixed, the gradient with respect to q_{jk} is linear:

$$\frac{\partial}{\partial q_{jk}} \hat{R}^{\text{TCP}}_{\text{QDA}}\big(\theta \mid \hat{\theta}^{\mathcal{S}}, z, q\big) = -\frac{1}{m}\log\frac{\mathcal{N}(z_j \mid \theta_k)}{\mathcal{N}(z_j \mid \hat{\theta}^{\mathcal{S}}_k)}\,.$$

Algorithm 4 lists pseudo-code for TCP-QDA.

6.4.2. Linear discriminant analysis

If the model is constrained to share a single covariance matrix for each class, the resulting classifier is a linear function of the difference in means and is hence termed linear discriminant analysis (LDA). This constraint is imposed through the weighted sum over class covariance matrices $\Sigma = \sum_k^K \pi_k \Sigma_k$.

6.4.3. Performance guarantee

The discriminant analysis model has a very surprising property: it obtains a *strictly* smaller risk than the source classifier. To our knowledge, this is the first time that such a performance guarantee can be given in the context of domain adaptation.

Theorem 1. *For a continuous target distribution, with more unique samples than features for every class, $m_k > D$, the empirical target risk of a discriminant analysis model $\hat{R}_{\text{DA}}$ with TCP estimated parameters $\hat{\theta}^{\mathcal{T}}$ is strictly smaller than the empirical target risk of a discriminant analysis model with parameters $\hat{\theta}^{\mathcal{S}}$ estimated on the source domain:*

$$\hat{R}_{\text{DA}}\big(\hat{\theta}^{\mathcal{T}} \mid z, u\big) < \hat{R}_{\text{DA}}\big(\hat{\theta}^{\mathcal{S}} \mid z, u\big)$$

Algorithm 4 TCP-QDA

Input: source data x (size $n \times D$), labels y (size $n \times K$), target data z (size $m \times D$), learning rate α, convergence criterion ϵ.
Output: $\hat{\theta}^{\mathcal{T}}_{\text{QDA}} = (\pi_1, \ldots, \pi_K, \mu_1, \ldots, \mu_K, \Sigma_1, \ldots, \Sigma_K)$
for all classes **do**
  $\hat{\pi}^{\mathcal{S}}_k = n^{-1} \sum_i^n y_{ik}$
  $\hat{\mu}^{\mathcal{S}}_k = \left(\sum_i^n y_{ik}\right)^{-1} \sum_i^n y_{ik} x_i$
  $\hat{\Sigma}^{\mathcal{S}}_k = \left(\sum_i^n y_{ik}\right)^{-1} \sum_i^n y_{ik} (x_i - \hat{\mu}^{\mathcal{S}}_k)^\top (x_i - \hat{\mu}^{\mathcal{S}}_k)$
end for
$t = 0$
$\theta^t_k = (\hat{\pi}^{\mathcal{S}}_k, \hat{\mu}^{\mathcal{S}}_k, \hat{\Sigma}^{\mathcal{S}}_k) \qquad \forall k$
$q^t_{jk} \leftarrow 1/K \qquad \forall j, k$
repeat
  for all classes **do**
    $\pi_k = m^{-1} \sum_j^m q^t_{jk}$
    $\mu_k = \left(\sum_j^m q^t_{jk}\right)^{-1} \sum_j^m q^t_{jk} z_j$
    $\Sigma_k = \left(\sum_j^m q^t_{jk}\right)^{-1} \sum_j^m q^t_{jk} (z_j - \mu_k)^\top (z_j - \mu_k)$
    $\theta^{t+1}_k = (\pi_k, \mu_k, \Sigma_k)$
    for all samples **do**
      $\nabla q_{jk} = -\log\left[\mathcal{N}(z_j \mid \theta^{t+1}_k) / \mathcal{N}(z_j \mid \hat{\theta}^{\mathcal{S}}_k)\right]$
    end for
  end for
  $q^{t+1} \leftarrow \mathcal{P}\left(q^t - \alpha^t \nabla q\right)$
  $\alpha^{t+1} \leftarrow \alpha / t$
  $t \leftarrow t + 1$
until $\| \hat{R}^{\text{TCP}}_{\text{QDA}}\left(\theta^{t+1} \mid \hat{\theta}^{\mathcal{S}}, z, q^{t+1}\right) - \hat{R}^{\text{TCP}}_{\text{QDA}}\left(\theta^t \mid \hat{\theta}^{\mathcal{S}}, z, q^t\right) \| \leq \epsilon$

The reader is referred to Appendix A for the proof. It follows similar steps as a robust guarantee for discriminant analysis in semi-supervised learning [37]. Note that as long as the same amount of regularization λ is added to both the source $\hat{\theta}^{\mathcal{S}}$ and the TCP classifier $\hat{\theta}^{\mathcal{T}}$, the guarantee also holds for a regularized model.

It should also be noted that the risks of TCP-LDA and TCP-QDA are always strictly smaller with respect to the given target samples, but not necessarily strictly smaller with respect to *new* target samples. Although, when the given target samples are a good representation of the target distribution, one does expect the adapted model to generalize well to new target samples.

6.5. Experiments

Our experiments compare the risks of the TCP classifiers with that of the source classifier and the corresponding oracle target classifier, as well as their performance with

respect to various state-of-the-art domain adaptive classifiers through their areas under the ROC-curve. In all experiments, all target samples are given, unlabeled, to the adaptive classifiers. They make predictions for those given target samples and their performance is evaluated with respect to those target samples' true labels. Cross-validation for regularization parameters was done by holding out source data, as that is the only data for which labels are available at training time. The range of values we tested was $\left[0\ 10^{-6}\ 10^{-5}\ 10^{-4}\ 10^{-3}\ 10^{-2}\ 10^{-1}\ 10^{0}\ 10^{1}\ 10^{2}\ 10^{3}\right]$.

6.5.1. Compared methods

We implemented transfer component analysis (TCA) [8], kernel mean matching (KMM) [26], robust covariate shift adjustment (RCSA) [29] and the robust bias-aware (RBA) classifier [25] for the comparison (see cited papers for more information). TCA and KMM are chosen because they are popular classifiers with clear assumptions. RCSA and RBA are chosen because they also employ minimax formulations but from different perspectives; RCSA as a worst-case and RBA as a moment-matching importance weighing. Their implementations details are discussed shortly below.

Transfer component analysis TCA assumes that there exists a common latent representation for both domains and aims to find this representation by means of a cross-domain nonlinear component analysis [8]. In our implementation, we employ a radial basis function kernel with a bandwidth of 1 and set the trade-off parameter μ to $1/2$. After mapping the data onto their transfer components, we train a logistic regressor on the mapped source data and apply it to the mapped target data.

Kernel mean matching KMM assumes that the class-posterior distributions are equal in both domains and that the support of the target distribution is contained within the source distribution [20, 26]. When the first assumption fails, KMM will have deviated from the source classifier in a manner that will not lead to better results on the target domain. When the second assumptions fails, the variance of the importance-weights increases to the point where a few samples receive large weights and all other samples receive very small weights, reducing the effective training sample size and leading to pathological classifiers. We use a radial basis function kernel with a bandwidth of 1, kernel regularization of 0.001 to favor estimates with lower variation over weights and upper bound the weights by $10\,000$. After estimating importance weights, we train a weighed least-squares classifier on the source samples.

Robust covariate shift adjustment RCSA also assumes equal class-posterior distributions and containment of the support of the target distribution within the source distribution, but additionally incorporates a worst-case labeling [29]. To be precise, it maximizes risk with respect to the importance weights. We used the author's publicly available code with 5-fold cross-validation for its hyperparameters. Interestingly, the authors also discuss a relation between covariate shift and model misspecification, as described by [38]. They argue for a two-step (estimate weights - train classifier) approach in a game-theoretical form [29, 39, 40], which is done by all importance-weighted classifiers here.

Robust bias-aware RBA assumes that the moments of the feature statistics are approximately equal up to a particular order [25]. In their formulation, the adversary plays a classifier whose class-posterior probabilities are used as a labeling of the target samples, but who is also constrained to match the moments with the source domain's statistics. The player then proposes an importance-weighted classifier that aims to perform well on both domains. Note that the constraints on the adversary are, among others, necessary to avoid the players switching strategies constantly. We implement RBA using first-order feature statistics for the moment-matching constraints, which was also done by the authors in their paper. Furthermore, we use a ratio of normal distributions for the weights and bound them above by 1000.

6.5.2. Sample selection bias setting

Sample selection bias settings occur when data is collected locally from a larger population. For regression problems, these settings are usually created through a parametric sampling of the feature space [5, 26]. We created something similar but for classification problems: samples are concentrated around a certain subset of the feature space, but with equal class priors as the whole set. For each class:

1. Find the sample closest to the origin; x_0.
2. Compute distance $d(x_0, x_k)$ to all other samples of the same class.
3. Draw without replacement $\pi_k n^S$ samples proportional to $\exp(-d(x_0, x_k))$.

where n^S denotes the total number of samples to draw and π_k refers to the class-prior distributions of the whole set. Note that drawing $\pi_k n$ samples from each class leads to approximately the same class prior distributions in the source domain as the target domain. We chose the squared Mahalanobis distance: $d(x_0, x_k) := (x_0 - x_k)\Sigma^{-1}(x_0 - x_k)^\top$, with the covariance matrix estimated on all data, since that takes scale differences between features into account. Figure 6.1 presents an example, showing the first two principal components of the pima diabetes dataset. Red/blue squares denote the selected source samples, black circles denote all samples and the green stars denote the seed points (x_0 for each class).

Data sets

We collected the following datasets from the UCI machine learning repository: cylinder bands printing (bands), car evaluation (car), credit approval (credit), ionosphere (iono), mammographic masses (mamm), pima diabetes (pima) and tic-tac-toe endgame (t3). Table 6.1 lists their characteristics. All missing values have been imputed to 0. For each dataset, we draw $n^S = 50$ samples as the source domain while treating all samples as the target domain.

Results

The risks (average negative log-likelihoods for the discriminant analysis models and mean squared errors for the least-squares classifiers) in Table 6.2 belong to the source classifiers, the TCP classifiers and the oracle target classifiers. The oracles represent the best possible result, as they comprise the risk of a classifier trained on all target samples with their true

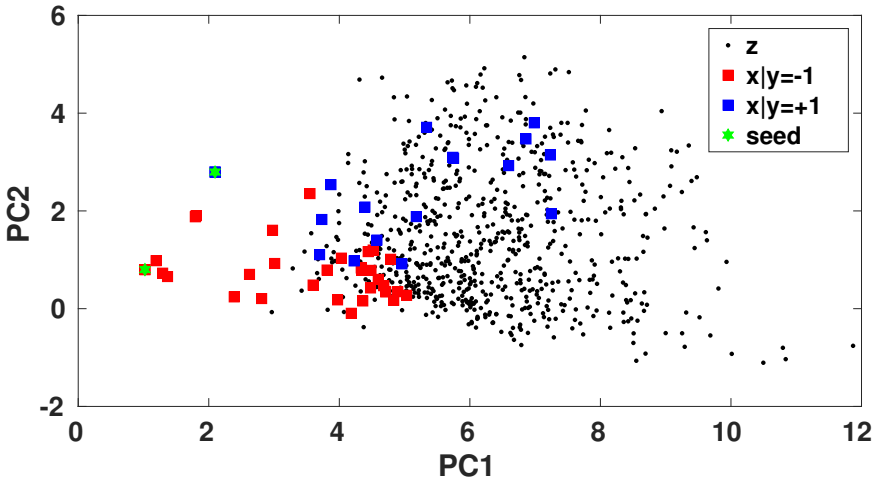


Figure 6.1: Example of a biased sampling. Shown are the first two principal components of the pima diabetes dataset, with all target samples in black, the selected source samples in red/blue and the samples closest to 0 of each class in green (seeds).

Table 6.1: Sample selection bias datasets characteristics.

	#Samples	#Features	#Missing	Class (-1|+1)
bands	539	39	569	312 | 227
car	1728	6	0	1210 | 518
credit	690	15	67	307 | 383
iono	351	34	0	126 | 225
mamm	961	5	162	516 | 445
pima	768	8	0	500 | 268
t3	958	9	0	332 | 626

labels. The results show varying degrees of improvement for the TCP classifiers. TCP-LDA approaches T-LDA more closely than the other two versions, with TCP-LS being the most conservative one. For the ionosphere and tic-tac-toe datasets, the improvement is quite dramatic, indicating that the source classifier is a poor model for the target domain. Note also that some overfitting might be occurring as TCP-QDA does not always have a lower risk than TCP-LDA, even though T-QDA does always have a lower risk than T-LDA.

Table 6.3 compares the performances of the adaptive classifiers on all datasets through their area under the ROC-curves (AUC). Although there is quite a variety between datasets, the variation between classifiers within a dataset is relatively small; all approaches perform similarly well. However, with our selection bias procedure, the moments of the target statistics do not match the source statistics (e.g. the target's variance is by construction always larger) which affect RBA's performance negatively. Interestingly, the TCP discriminant analysis models are quite competitive in cases where their improvement over the

Table 6.2: Risks (average negative log-likelihoods and mean squared errors) of the naive source classifiers (S-LDA, S-QDA, S-LS), the TCP classifiers (TCP-LDA, TCP-QDA, TCP-LS) and the oracle target classifiers (T-LDA, T-QDA, T-LS) on the sample selection bias datasets.

	S-LDA	TCP-LDA	T-LDA	S-QDA	TCP-QDA	T-QDA	S-LS	TCP-LS	T-LS
bands	-216.3	-218.4	-218.8	-215.3	-217.8	-219.1	1.170	1.109	0.827
car	17.16	2.850	2.148	57.39	18.77	2.049	1.968	1.205	0.672
credit	-80.04	-83.64	-83.65	-78.99	-83.73	-84.61	2.430	0.973	0.757
iono	199.5	-8.480	-8.782	26.30	-9.325	-18.78	17.06	0.815	0.350
mamm	8.133	-10.40	-11.22	31.66	-10.08	-11.68	0.818	0.668	0.580
pima	-15.92	-23.44	-24.15	-7.486	-23.09	-24.30	1.083	1.012	0.633
t3	18.77	6.136	4.734	117.3	39.13	4.611	1.401	1.401	0.849

source classifier was larger. Unfortunately, like RBA, the more conservative TCP-LS never outperforms all other methods simultaneously on any of the datasets. Still, in the average it reaches competitive performance overall. In summary, the TCP classifiers perform on par with the other adaptive classifiers.

Table 6.3: Sample selection bias datasets. Areas under the ROC-curves for a range of domain adaptive classifiers.

	TCA	KMM	RCSA	RBA	TCP-LS	TCP-LDA	TCP-QDA
bands	.578	**.620**	.562	.504	.588	.548	.589
car	.736	**.776**	.742	.684	.734	.758	.699
credit	**.716**	.694	.655	.702	.662	.646	.663
iono	.741	.817	.835	.687	.731	**.894**	.826
mamm	.656	.804	.749	.762	.836	.824	**.847**
pima	.691	.630	**.760**	.271	.692	.684	.637
t3	**.608**	.532	.439	.446	.520	.529	.606
mean	.675	.696	.677	.579	.680	**.698**	.695

6.5.3. Domain adaptation setting

We performed a set of experiments on a dataset that is naturally split into multiple domains: predicting heart disease in patients from hospitals in 4 different locations. It is a much more realistic setting because problem variables such as prior shift, class imbalance and proportion of imputed features are not controlled. As such, it is a harder problem than the sample selection bias setting. In this setting, the target domains often only have limited overlap with the source domain and can be very dissimilar. As the results will show, many of the assumptions that the state-of-the-art domain adaptive classifiers rely upon, do not hold and their performance degrades drastically.

Data set

The hospitals are the Hungarian Institute of Cardiology in Budapest (data collected by Andras Janosi), the University Hospital Zurich (collected by William Steinbrunn), the University Hospital Basel (courtesy of Matthias Pfisterer), the Veterans Affairs Medical Center in

Long Beach, California, USA, and the Cleveland Clinic Foundation in Cleveland, Ohio, USA (both courtesy of Robert Detrano), which will be referred to as Hungary, Switzerland, California and Ohio hereafter. The data from these hospitals can be considered domains as the patients are all measured on the same biometrics but show different distributions. For example, patients in Hungary are on average younger than patients from Switzerland (48 versus 55 years). Each patient is described by 13 features: age, sex, chest pain type, resting blood pressure, cholesterol level, high fasting blood sugar, resting electrocardiography, maximum heart rate, exercise-induced angina, exercise-induced ST depression, slope of peak exercise ST, number of major vessels in fluoroscopy, and normal/defective/reversible heart rate.

Table 6.4 describes the number of samples (n, m), total number of missing measurements that have been imputed ($mis_{\mathcal{S}}$, $mis_{\mathcal{T}}$) the class balance ($c_{\mathcal{S}}$, $c_{\mathcal{T}}$) and the empirical Maximum Mean Discrepancy (MMD) for all pairwise combinations of designating one domain as the source and another as the target. To elaborate: the empirical MMD measures how far apart two sets of samples are [26]:

$$\begin{aligned}\hat{\text{MMD}} &= \|n^{-1}\sum_{i}^{n}\phi(x_i) - m^{-1}\sum_{j}^{m}\phi(z_j)\|^2 \\ &= n^{-2}\sum_{i,i'}^{n}K(x_i,x_{i'}) - 2(nm)^{-1}\sum_{i,j}K(x_i,z_j) + m^{-2}\sum_{j,j'}^{m}K(z_j,z_{j'})\,.\end{aligned}$$

In order to compute it, we used a radial-basis function with a bandwidth of 1. An MMD of 0 means that the two sets are identical, while larger values indicate larger discrepancies between the two sets.

First of all, the sample size imbalance is not really a problem, as the largest difference occurs in the Ohio - Switzerland combination with 303 and 123 samples respectively. However, the fact that the classes are severely imbalanced in different proportions, for example going from 54% : 46% to 7% : 93% in Ohio - Switzerland, creates a very difficult setting. A shift in the prior distributions can be disastrous for some classifiers, such as RBA which relies on matching the source and target feature statistics. Furthermore, a sudden increase in the amount of missing values (unmeasured patient biometrics), such as in Ohio - California, means that a classifier relying on a certain feature for discrimination degrades when this feature is missing in the target domain. Additionally, the combinations Ohio - Switzerland and Switzerland - Hungary have an MMD that is two orders of magnitude larger than other combinations. Overall, looking at all three sets of descriptive statistics, the combinations Ohio - Switzerland and Switzerland - Hungary should pose the most difficulty for the adaptive classifiers.

Lastly, to further illustrate how the domains differ, we plotted histograms of the age and resting blood pressure of all patients, split by domain (see Figure 6.2). Not only are they different on average, they tend to differ in variance and skewness as well.

Table 6.4: Heart disease dataset properties, for all pairwise domain combinations (O='Ohio', C='California', H='Hungary' and S='Switzerland'). $\mathcal{S}$ denotes the source and $\mathcal{T}$ the target domain, n the amount of source and m the amount of target samples, $c_{\mathcal{S}}$ the class balance (-1,+1) in the source domain and $c_{\mathcal{T}}$ the class balance in the target domain. MMD denotes the empirical Maximum Mean Discrepancy between the source and target data.

$\mathcal{S}$	$\mathcal{T}$	n	m	$mis_{\mathcal{S}}$	$mis_{\mathcal{T}}$	$c_{\mathcal{S}}$	$c_{\mathcal{T}}$	MMD
O	H	303	294	6	782	164:139	188:106	0.0012
O	S	303	123	6	273	164:139	8:115	0.1602
O	C	303	200	6	698	164:139	51:149	0.0227
H	S	294	123	782	273	188:106	8:115	0.1384
H	C	294	200	782	698	188:106	51:149	0.0151
S	C	123	200	273	698	8:115	51:149	0.0804
H	O	294	303	782	6	188:106	164:139	0.0012
S	O	123	303	273	6	8:115	164:139	0.1602
C	O	200	303	698	6	51:149	164:139	0.0227
S	H	123	294	273	782	8:115	188:106	0.1384
C	H	200	294	698	782	51:149	188:106	0.0151
C	S	200	123	698	273	51:149	8:115	0.0804

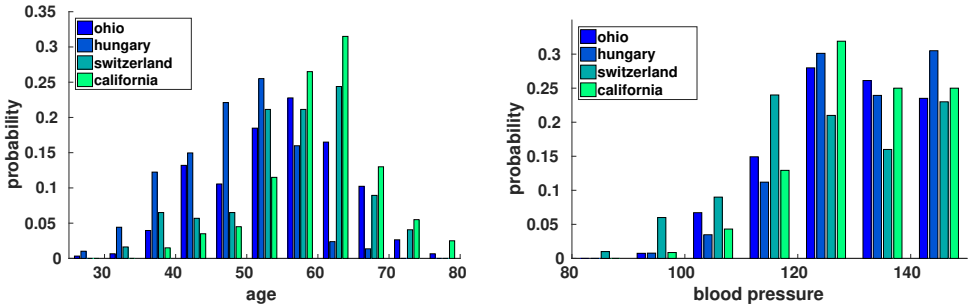


Figure 6.2: (Left top) Histogram of the age of patients in each domain, (right) histogram of the resting blood pressure of patients in each domain.

Results

Table 6.5 lists the target risks (average negative log-likelihoods for the discriminant analysis models and mean squared errors for the least-squares classifiers) with the given target samples' true labels for all source, TCP and oracle target classifiers. Note that the TCP risks range between the source and the oracle target risk. For some combinations TCP is extremely conservative, e.g. Switzerland - Ohio, Switzerland - Hungary for the least-squares case, and for others, it is much more liberal, e.g. Hungary - Switzerland, Hungary - Ohio, Hungary - California for the discriminant analysis models. In general, the discriminative model (TCP-LS) deviates much less and is much more conservative than the generative models (TCP-LDA and TCP-QDA). Note that the order of magnitude of the improvement with TCP-DA in the Hungary - Switzerland, Hungary - Ohio and Hungary - California settings is due to the fact that the two domains lie far apart; the target samples lie very far in the tails of the source models' Gaussian distribution and evaluate to very small likelihoods,

which become very large negative log-likelihoods.

Table 6.5: Heart disease dataset. Target risks (average negative log-likelihoods (left, middle) and mean squared errors (right)) for all pairwise combinations of domains (O='Ohio', C='California', H='Hungary' and S='Switzerland'; smaller values are better).

$\mathcal{S}$ $\mathcal{T}$	S-LDA	TCP-LDA	T-LDA	S-QDA	TCP-QDA	T-QDA	S-LS	TCP-LS	T-LS
O H	-53.55	-57.18	-57.35	-53.55	-57.20	-57.62	0.580	0.579	0.444
O S	-8.293	-16.76	-17.54	-8.293	-16.76	-17.54	1.449	1.449	0.213
O C	-37.84	-53.88	-54.69	-37.83	-53.73	-54.89	1.441	1.441	0.613
H S	-12.50	-16.08	-17.54	-12.80	-16.44	-17.54	1.068	1.068	0.213
H C	-41.70	-53.91	-54.69	-40.08	-54.45	-54.89	1.120	1.104	0.613
S C	494.9	-54.49	-54.69	498.9	-54.44	-54.89	0.904	0.904	0.671
H O	-48.91	-55.08	-55.23	-49.20	-54.84	-55.53	0.642	0.638	0.463
S O	709.9	-54.07	-55.23	709.9	-54.10	-55.53	1.700	1.700	0.696
C O	-49.21	-55.00	-55.23	-49.17	-55.05	-55.53	1.833	1.833	0.470
S H	649.9	-56.09	-57.35	650.3	-56.19	-57.62	2.102	2.102	0.740
C H	-53.05	-57.19	-57.35	-53.15	-57.17	-57.62	0.582	0.582	0.444
C S	-15.45	-17.43	-17.54	-15.47	-17.44	-17.54‘	0.415	0.415	0.236

Looking at the areas under the ROC-curves in Figure 6.6, one observes a different pattern in the classifier performances. TCA, KMM, RCSA and RBA perform much worse, often below chance level. It can be seen that, in some cases, the assumption of equal class-posterior distributions still holds approximately, as KMM and RCSA sometimes perform quite well, e.g. in Hungary - Ohio. TCA's performance varies around chance level, indicating that it is difficult to recover a common latent representation in these settings. That makes sense, as the domains lie further apart this time. RBA's performance drops most in cases where the differences in priors and proportions of missing values are largest, e.g. Hungary - California, which also makes sense as it is expecting similar feature statistics in both domains. TCP-LS performs very well in almost all cases; the conservative strategy is paying off. TCP-LDA is also performing very well, even outperforming TCP-QDA in all cases. The added flexibility of a covariance matrix per class is not beneficial because it is much more difficult to fit correctly. Note that the domain combinations are asymmetrical; for example, RCSA's performance is quite strong when Switzerland is the source domain and Ohio the target domain, but it's performance is much weaker when Ohio is the source domain and Switzerland the target domain. In some combinations, assumptions on how two domains are related to each other might be valid that are not valid in their reverse combinations. Overall, in this more general domain adaptation setting, our more conservative approach works best, as shown by the mean performances.

Visualization of the worst-case labeling

The adversary in TCP's minimax formulation maximizes the objective with respect to the probability q_{jk} that a sample j belongs to class k. However, note that the worst-case labeling corresponds to the labeling that maximizes the contrast: it looks for the labeling for which the difference between the source parameters and the current parameters is largest.

Table 6.6: Heart disease dataset. Area under the ROC-curve for all pairwise combinations of domains (O='Ohio', C='California', H='Hungary' and S='Switzerland'); larger values are better.

$\mathcal{S}$	$\mathcal{T}$	TCA	KMM	RCSA	RBA	TCP-LS	TCP-LDA	TCP-QDA
O	H	.699	.710	.372	.481	.881	**.882**	.817
O	S	.590	.551	.634	.670	**.714**	.671	.671
O	C	.496	.476	.560	.450	**.671**	.668	.476
H	S	.455	.501	.646	.602	**.668**	.665	.666
H	C	.528	.533	.585	.434	**.727**	.709	.662
S	C	.475	.573	.464	.603	**.605**	.546	.480
H	O	.616	.742	.751	.510	.864	**.876**	.863
S	O	.582	.353	.750	.449	**.753**	.589	.426
C	O	.484	.337	.551	.557	.671	**.831**	.828
S	H	.407	.370	.629	.484	.697	**724**	.604
C	H	.472	.427	.538	.616	.805	**.878**	.824
C	S	.511	.593	.462	.474	**.709**	.503	.535
mean		.526	.514	.578	.528	**.730**	.712	.654

It would be interesting to visualize this labeling at the saddle point. Figure 6.3 shows the first two principal components of Hungary, with the probabilities of belonging to class 1, i.e. $q_{jk=1}$. The top left figure shows the true labeling, the top right the probabilities for TCP-LS, the bottom left for TCP-LDA and the bottom right for TCP-QDA. In all three TCP cases the labeling is quite smooth and does not vary too much between neighboring points. One would expect a rough labeling, but note that labellings that are bad for the source classifier will most likely also be bad for the TCP classifier, and the resulting contrast will be small instead of maximal. The probabilities for TCP-LS lie closer to 0 and 1 than for TCP-LDA and TCP-QDA.

6.6. Discussion

Although the TCP classifiers are never worse than the source classifier by construction, they will not automatically lead to improvements in the error rate. This is due to the difference between optimizing a surrogate loss and evaluating the 0/1-loss [37, 41, 42]. There is no one-to-one mapping between the minimizer of the surrogate loss and the minimizer of the 0/1-loss.

One peculiar advantage of our TCP model is that we do not explicitly require source samples at training time. They are not incorporated in the risk formulation, which means that they do not have to be retained in memory. It is sufficient to receive the parameters of a trained classifier that can serve as a baseline. Our approach is therefore more efficient than for example importance-weighing techniques which require source samples for importance-weight estimation and subsequent training. Additionally, it would be interesting to construct a contrast with multiple source domains. The union of source classifiers might serve as a very good starting point for the TCP model.

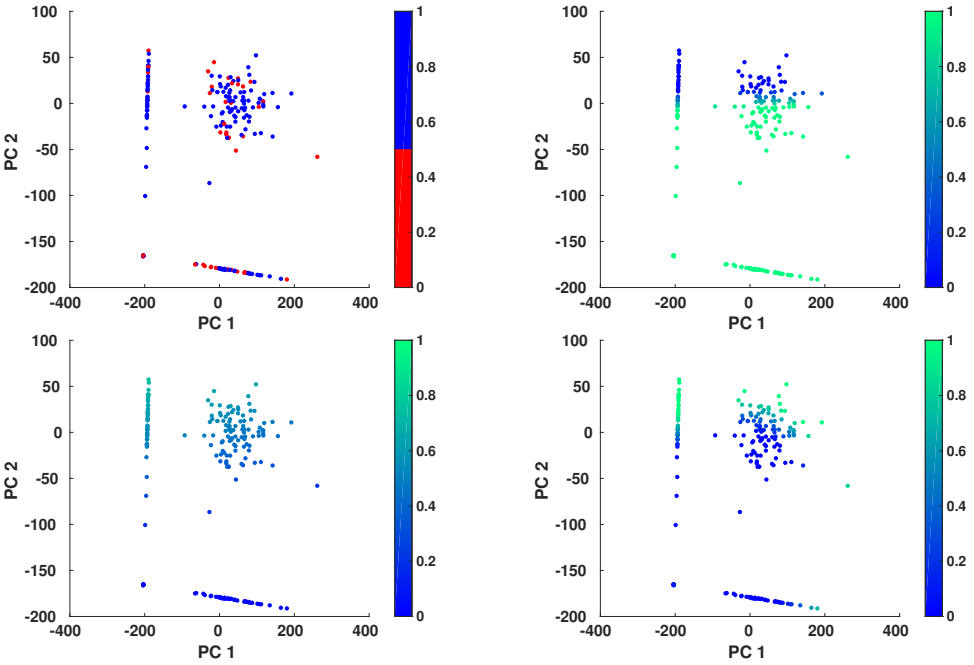


Figure 6.3: Scatter plots of the first two principal components of Hungary in the heart disease dataset. (Top left) True labeling, (top right) $q_{k=1}$ for TCP-LS, (bottom left) $q_{k=1}$ for TCP-LDA, (bottom right) $q_{k=1}$ for TCP-QDA.

For each adaptive classifier, regularization parameters are estimated through cross-validation on held-out source samples. However, this procedure is known to be biased as it does not account for domain dissimilarity [43, 44]. What is optimal with respect to held-out source samples, need not be optimal with respect to target samples. Performance of many adaptive models might be improved with appropriate adaptive validation techniques.

6.7. Conclusion

We have designed a risk minimization formulation for a domain-adaptive classifier whose performance, in terms of risk, is always at least as good as that of the non-adaptive source classifier. Furthermore, for the discriminant analysis case, its performance is always strictly better. Our target contrastive pessimistic model performs on par with state-of-the-art domain adaptive classifier on sample selection bias settings and outperforms them on more realistic domain adaptation problem settings.

6.8. Appendix A

Proof of Theorem 1. Let $\{(x_i, y_i)\}_{i=1}^n$ be a sample set of size n drawn iid from continuous distribution p_S, defined over D-dimensional real-valued input space $\mathcal{X} \subseteq \mathbb{R}^D$ and output space $\mathcal{Y} = \{1, \dots, K\}$ with K as the number of classes. Similarly, let $\{(z_j, u_j)\}_{j=1}^m$ be a sample set, of size m, drawn iid from continuous distribution p_T, defined over the same spaces. For the purposes of the following risk function, the labels of single samples, y_i and u_j, are encoded as 1 by K vectors, with the k-th element being the probability of belonging to the k-th class. Consider a discriminant analysis model parameterized either as $\theta = (\pi_1, \dots, \pi_K, \mu_1, \dots, \mu_K, \Sigma_1, \dots \Sigma_K)$ for QDA or $\theta = (\pi_1, \dots, \pi_K, \mu_1, \dots, \mu_K, \Sigma)$ for LDA. $\hat{R}_{\text{DA}}$ denotes empirical risk consisting of average negative Gaussian log-likelihoods weighted by labels:

$$\hat{R}_{\text{DA}}(\theta \mid x, y) = \frac{1}{m} \sum_{j=1}^{m} \sum_{k=1}^{K} -y_{ik} \log \mathcal{N}(x_i \mid \theta_k) \,.$$

Note that θ_k refers to (π_k, μ_k, Σ_k) in the case of QDA and to (π_k, μ_k, Σ) in the case of LDA. The sample covariance matrix, Σ_k for QDA and Σ for LDA, is required to be non-singular, which is guaranteed when there are more unique samples than features for every class, $m_k > D$. In the LDA case, $D + K$ unique samples are sufficient. Let $\hat{\theta}^S$ be the parameters estimated on labeled source data; $\hat{\theta}^S = \arg\min_{\theta \in \Theta} \hat{R}_{\text{DA}}(\theta \mid x, y)$.

Firstly, for fixed q, the minimized contrast between the target risk of any parameter θ and the source parameters $\hat{\theta}^S$ is non-positive, because both parameters sets are elements of the same parameter space, $\theta, \hat{\theta}^S \in \Theta$:

$$\min_{\theta \in \Theta} \hat{R}_{\text{DA}}(\theta \mid z, q) - \hat{R}_{\text{DA}}(\hat{\theta}^S \mid z, q) \leq 0 \,.$$

θ's that result in a larger target risk than that of $\hat{\theta}^S$ are not minimizers of the contrast. The maximum value it can attain is 0, which occurs when exactly the same parameters are found; $\theta = \hat{\theta}^S$. Considering that the contrast is non-positive for any labeling q, it is also non-positive with respect to the worst-case labeling:

$$\min_{\theta \in \Theta} \max_{q \in \Delta_{K-1}^m} \hat{R}_{\text{DA}}(\theta \mid z, q) - \hat{R}_{\text{DA}}(\hat{\theta}^S \mid z, q) \leq 0 \,. \tag{6.11}$$

Secondly, given that the empirical risk with respect to the true labeling is always less than or equal to the empirical risk with the worst-case labeling, $\hat{R}(\theta \mid z, u) \leq \max_q \hat{R}(\theta \mid z, q)$, the target contrastive risk (6.3) with the true labeling u is always less than or equal to the target contrastive pessimistic risk (6.5):

$$\min_{\theta \in \Theta} \hat{R}_{\text{DA}}(\theta \mid z, u) - \hat{R}_{\text{DA}}(\hat{\theta}^S \mid z, u) \leq \min_{\theta \in \Theta} \max_{q \in \Delta_{K-1}^m} \hat{R}_{\text{DA}}(\theta \mid z, q) - \hat{R}_{\text{DA}}(\hat{\theta}^S \mid z, q) \,. \tag{6.12}$$

Let $(\hat{\theta}^{\mathcal{T}}, q^*)$ be the minimaximizer of the target contrastive pessimistic risk on the right-handside of (6.12). Plugging these estimates in into (6.12) produces:

$$\hat{R}_{\text{DA}}\big(\hat{\theta}^{\mathcal{T}} \mid z, u\big) - \hat{R}_{\text{DA}}\big(\hat{\theta}^{\mathcal{S}} \mid z, u\big) \;\leq\; \hat{R}_{\text{DA}}\big(\hat{\theta}^{\mathcal{T}} \mid z, q^*\big) - \hat{R}_{\text{DA}}\big(\hat{\theta}^{\mathcal{S}} \mid z, q^*\big)\,. \tag{6.13}$$

Combining inequalities 6.11 and 6.13 gives:

$$\hat{R}_{\text{DA}}\big(\hat{\theta}^{\mathcal{T}} \mid z, u\big) - \hat{R}_{\text{DA}}\big(\hat{\theta}^{\mathcal{S}} \mid z, u\big) \leq 0\,.$$

Bringing the second term on the left-handside to the right-handside shows that the target risk of the TCP estimate is always less than or equal to the target risk of the source classifier's:

$$\hat{R}_{\text{DA}}\big(\hat{\theta}^{\mathcal{T}} \mid z, u\big) \leq \hat{R}_{\text{DA}}\big(\hat{\theta}^{\mathcal{S}} \mid z, u\big)\,. \tag{6.14}$$

However, equality of the two risks in 6.14 occurs with probability 0, which we will show in the following.

The total mean for the source classifier consists of the weighted combination of the class means, resulting in the overall source sample average:

$$\begin{aligned}
\mu^{\mathcal{S}} &= \sum_{k=1}^{K} \pi_k^{\mathcal{S}}\, \mu_k^{\mathcal{S}} \\
&= \sum_{k=1}^{K} \frac{\sum_i^n y_{ik}}{n} \left[\frac{1}{\sum_i^n y_{ik}} \sum_{i=1}^{n} y_{ik} x_i \right] \\
&= \frac{1}{n} \sum_{i=1}^{n} x_i\,.
\end{aligned} \tag{6.15}$$

The total mean for the TCP-DA estimator is similarly defined, resulting in the overall target sample average:

$$\begin{aligned}
\mu^{\mathcal{T}} &= \sum_{k=1}^{K} \pi_k^{\mathcal{T}}\, \mu_k^{\mathcal{T}} \\
&= \sum_{k=1}^{K} \frac{\sum_j^m q_{jk}}{m} \left[\frac{1}{\sum_j^m q_{jk}} \sum_{j=1}^{m} q_{jk} z_j \right] \\
&= \sum_{k=1}^{K} \frac{1}{m} \sum_{j=1}^{m} q_{jk} z_j
\end{aligned} \tag{6.16}$$

$$= \frac{1}{m} \sum_{j=1}^{m} z_j\,. \tag{6.17}$$

Note that since q consists of probabilities, the sum over classes $\sum_k^K q_{jk}$ in (6.16) is 1, for every sample j. Equal risks for these parameter sets, $\hat{R}_{\text{DA}}(\theta^{\mathcal{T}} \mid z, u) = \hat{R}_{\text{DA}}(\hat{\theta}^{\mathcal{S}} \mid z, u)$, implies equality of the total means, $\mu^{\mathcal{T}} = \mu^{\mathcal{S}}$. By Equations 6.15 and 6.17, equal total means imply equal sample averages: $m^{-1} \sum_j^m z_j = n^{-1} \sum_i^n x_i$. Given a set of source samples, drawing a set of target samples such that their averages are *exactly equal*, constitutes a single event under a probability density function:

$$p_{\mathcal{T}}\left(\mathcal{X}_1 = z_1, \ldots, \mathcal{X}_m = z_m \mid \frac{1}{m} \sum_{j=1}^{m} z_j = \frac{1}{n} \sum_{i=1}^{n} x_i \right).$$

By definition, single events under continuous distributions have probability 0. Therefore, a strictly smaller risk occurs almost surely:

$$\hat{R}_{\text{DA}}(\hat{\theta}^{\mathcal{T}} \mid z, u) < \hat{R}_{\text{DA}}(\hat{\theta}^{\mathcal{S}} \mid z, u).$$

□

References

[1] J. Quionero-Candela, M. Sugiyama, A. Schwaighofer, and N. D. Lawrence, *Dataset shift in machine learning* (MIT Press, 2009).

[2] J. G. Moreno-Torres, T. Raeder, R. Alaiz-RodríGuez, N. V. Chawla, and F. Herrera, *A unifying view on dataset shift in classification*, Pattern Recognition **45**, 521 (2012).

[3] B. Gong, Y. Shi, F. Sha, and K. Grauman, *Geodesic flow kernel for unsupervised domain adaptation*, in *Conference on Computer Vision and Pattern Recognition* (IEEE, 2012) pp. 2066–2073.

[4] H. Hachiya, M. Sugiyama, and N. Ueda, *Importance-weighted least-squares probabilistic classifier for covariate shift adaptation with application to human activity recognition*, Neurocomputing **80**, 93 (2012).

[5] H. Shimodaira, *Improving predictive inference under covariate shift by weighting the log-likelihood function*, Journal of Statistical Planning and Inference **90**, 227 (2000).

[6] C. Cortes, M. Mohri, M. Riley, and A. Rostamizadeh, *Sample selection bias correction theory*, in *Algorithmic Learning Theory* (Springer, 2008) pp. 38–53.

[7] S. Bickel, M. Brückner, and T. Scheffer, *Discriminative learning under covariate shift*, Journal of Machine Learning Research **10**, 2137 (2009).

[8] S. Pan, I. Tsang, J. Kwok, and Q. Yang, *Domain adaptation via transfer component analysis*, Neural Networks **22**, 199 (2011).

[9] S. Ben-David, J. Blitzer, K. Crammer, and F. Pereira, *Analysis of representations for domain adaptation*, in *Advances in Neural Information Processing Systems* (2007) pp. 137–144.

[10] S. Ben-David, T. Lu, T. Luu, and D. Pál, *Impossibility theorems for domain adaptation*, in *International Conference on Artificial Intelligence and Statistics* (2010) pp. 129–136.

[11] S. Ben-David, J. Blitzer, K. Crammer, A. Kulesza, F. Pereira, and J. W. Vaughan, *A theory of learning from different domains*, Machine Learning **79**, 151 (2010).

[12] R. Gopalan, R. Li, and R. Chellappa, *Domain adaptation for object recognition: an unsupervised approach*, in *International Conference on Computer Vision* (IEEE, 2011) pp. 999–1006.

[13] M. Baktashmotlagh, M. Harandi, B. Lovell, and M. Salzmann, *Domain adaptation on the statistical manifold*, in *Conference on Computer Vision and Pattern Recognition* (2014) pp. 2481–2488.

[14] V. M. Patel, R. Gopalan, R. Li, and R. Chellappa, *Visual domain adaptation: a survey of recent advances*, IEEE Signal Processing Magazine **32**, 53 (2015).

[15] B. Fernando, A. Habrard, M. Sebban, and T. Tuytelaars, *Unsupervised visual domain adaptation using subspace alignment*, in *International Conference on Computer Vision* (2013) pp. 2960–2967.

[16] W. M. Kouw, L. J. Van Der Maaten, J. H. Krijthe, and M. Loog, *Feature-level domain adaptation*, Journal of Machine Learning Research **17**, 1 (2016).

[17] G. W. Imbens and D. B. Rubin, *Causal inference in statistics, social, and biomedical sciences* (Cambridge University Press, 2015).

[18] J. J. Heckman, *Sample selection bias as a specification error (with an application to the estimation of labor supply functions)*, (1977).

[19] B. Zadrozny, *Learning and evaluating classifiers under sample selection bias*, in *International Conference on Machine Learning* (ACM, 2004) p. 114.

[20] A. Gretton, A. Smola, J. Huang, M. Schmittfull, K. Borgwardt, and B. Schölkopf, *Covariate shift by kernel mean matching*, in *Dataset Shift in Machine Learning*, edited by Q. Candela, M. Sugiyama, A. Schwaighofer, and N. Lawrence (MIT Press, 2009) pp. 131–160.

[21] A. Storkey, *When training and test sets are different: characterizing learning transfer*, Dataset Shift in Machine Learning , 3 (2009).

[22] B. Sch'olkopf, D. Janzing, J. Peters, E. Sgouritsa, K. Zhang, and J. Mooij, *On causal and anticausal learning*, in *International Conference on Machine Learning* (2012) pp. 1255–1262.

[23] B. Gong, K. Grauman, and F. Sha, *Connecting the dots with landmarks: discriminatively learning domain-invariant features for unsupervised domain adaptation*, in *International Conference on Machine Learning* (2013) pp. 222–230.

[24] C. Cortes and M. Mohri, *Domain adaptation and sample bias correction theory and algorithm for regression*, Theoretical Computer Science **519**, 103 (2014).

[25] A. Liu and B. Ziebart, *Robust classification under sample selection bias*, in *Advances in Neural Information Processing Systems* (2014) pp. 37–45.

[26] J. Huang, A. Gretton, K. M. Borgwardt, B. Schölkopf, and A. J. Smola, *Correcting sample selection bias by unlabeled data*, in *Advances in Neural Information Processing Systems* (2007) pp. 601–608.

[27] A. B. Owen, *Monte Carlo theory, methods and examples* (2013).

[28] J. O. Berger, *Statistical decision theory and Bayesian analysis* (Springer, 2013).

[29] J. Wen, C.-N. Yu, and R. Greiner, *Robust learning under uncertain test distributions: relating covariate shift to model misspecification*, in *International Conference on Machine Learning* (2014) pp. 631–639.

[30] J. H. Krijthe and M. Loog, *Projected estimators for robust semi-supervised classification*, Machine Learning , 1 (2017).

[31] L. I. Kuncheva, J. C. Bezdek, and R. P. Duin, *Decision templates for multiple classifier fusion: an experimental comparison*, Pattern Recognition **34**, 299 (2001).

[32] A. Cherukuri, B. Gharesifard, and J. Cortes, *Saddle-point dynamics: conditions for asymptotic stability of saddle points*, SIAM Journal on Control and Optimization **55**, 486 (2017).

[33] Y. Chen and X. Ye, *Projection onto a simplex*, arXiv preprint arXiv:1101.6081 (2011).

[34] L. Condat, *Fast projection onto the simplex and the ℓ_1 ball*, Preprint HAL **1056171** (2014).

[35] J. Friedman, T. Hastie, and R. Tibshirani, *The elements of statistical learning*, Vol. 1 (Springer, 2001).

[36] M. Mohri, A. Rostamizadeh, and A. Talwalkar, *Foundations of Machine Learning* (MIT Press, 2012).

[37] M. Loog, *Contrastive pessimistic likelihood estimation for semi-supervised classification*, IEEE Transactions on Pattern Analysis and Machine Intelligence **38**, 462 (2016).

[38] H. White, *Consequences and detection of misspecified nonlinear regression models*, Journal of the American Statistical Association **76**, 419 (1981).

[39] P. D. Grünwald and A. P. Dawid, *Game theory, maximum entropy, minimum discrepancy and robust Bayesian decision theory*, Annals of Statistics , 1367 (2004).

[40] M. Sugiyama and K.-R. Müller, *Model selection under covariate shift*, in *International Conference on Artificial Neural Networks* (Springer, 2005) pp. 235–240.

[41] P. Bartlett, *Prediction algorithms: complexity, concentration and convexity*, in *IFAC Symposium on System Identification* (2003) pp. 1507–1517.

[42] P. L. Bartlett, M. I. Jordan, and J. D. McAuliffe, *Convexity, classification, and risk bounds*, Journal of the American Statistical Association **101**, 138 (2006).

[43] M. Sugiyama, M. Krauledat, and K.-R. Müller, *Covariate shift adaptation by importance weighted cross validation*, Journal of Machine Learning Research **8**, 985 (2007).

[44] W. M. Kouw and M. Loog, *On regularization parameter estimation under covariate shift*, in *International Conference on Pattern Recognition* (IEEE, 2016) pp. 426–431.

7

Discussion

This chapter reflects on the work presented in this thesis as well. Several findings are discussed and a series of open questions is presented. Additionally, the benefit of domain adaptation to open science is considered and a future step towards dynamical domain adaptation is explored.

In Chapter 1, I outlined my research question: when and how can a statistical classifier generalize from a source to a target domain? This question is very general and the work presented in this thesis is insufficient to fully answer it. For the broadest case of domain shift, the answer is simple: it is impossible for a classifier to generalize from a source domain to *all* possible target domains. For particular cases, it depends on the relationship between the domains. There are many ways in which two domains can be related to each other, each with many possible of ways of exploiting that information for designing a classifier. These ways have not all been found. But, on reflection, some observations can be made and some new questions can be asked, which are presented in the following subsections.

With complexity of domain relationships, I mean how many variables change and how much they change. In this regard, the simplest change is the case of prior shift: only one discrete variable changes. However, even this case can be complicated to deal with. The rarer a class, the more difficult it is to estimate it. Furthermore, it is not entirely clear whether source samples should be reweighted to match the class proportions of the target domain or whether they should be balanced in order to facilitate training the classifier. Moreover, it is not entirely clear how they should be up- or downweighted: although theoretically a sample should be assigned a different loss, performance improvements have been reported for methods that upsample and interpolate between source samples. That implies an assumption of smooth variation in feature space and raises the question of whether this can always be assumed.

After prior shift, covariate shift is the most constrained case. In its simplest form, only one covariate (i.e. feature) changes. This has been extensively studied and many open questions such as how far the variable can shift, how many samples are required to estimate the importance-weight and how the classifier behaves under importance-weight estimation errors, have been addressed already [1, 2]. It seems that the most important things to check before attempting a method is: does the assumption of equal class-posterior distributions hold and if not, how strongly is it violated? Are the domains so far apart that the weights will become bimodal? Do you have enough source and target samples to estimate importance-weights? If weight estimation is done parametrically, do you have enough samples to prevent low probabilities in the denominator and if done non-parametrically, do you have enough samples to perform hyperparameter estimation (e.g. kernel bandwidth selection for kernel density estimators)? Is the sample selection variable smooth? Is there model misspecification (for weight or selection variable estimation and for training the classifier)?

Further open questions include: how does data preprocessing affect importance weight estimation versus classifier training? Should each domain be normalized separately to bring the domains closer together thereby avoiding weight bimodality or should this be avoided because it induces a violation of the covariate shift assumption? Is it ok to transform only the features that have not shifted between domains? Does the assumption of equal class-posterior distributions hold for a part of feature space? Can multiple source domains aid in weight estimation? Are hybrid distributions (joint distributions made up of

the product of discrete distributions for discrete features and continuous distributions for continuous features) necessary or useful for weight estimation?

The simplest version of concept shift, is where only one of the conditional distributions $p(y_k \mid x_d)$ changes. However, even such a situation is impossible to estimate from data without observation of at least one labeled sample in each domain, for each class. My advice would be to estimate it from meta-data: if the experimenter recorded *how* data in each domain is annotated, for instance when crowd-sourced annotators have to explain why they assigned a sample to a particular class, then it is possible to find discrepancies and correspondences between annotation strategies. This additional information might allow for correcting sets of labels and reduce the shift in concept. However, concrete methods for doing so are very application-specific.

The rest of the types of relationships are much more complicated, because multiple changes occur at the same time. For example, for subspace mappings, there are both changes in the data distributions and changes in the class-posterior distributions, and possible changes in class proportions as well. Since these are less constrained, they are harder to study. This makes it harder to predict whether a specific adaptive method will be successful for a given a domain shift problem. Furthermore, it is still unclear what the effects of sample sizes or estimation errors are for methods based on subspace mappings, domain-invariant spaces, domain manifolds, low-joint-error, etc. It would be very informative to study these factors.

In conclusion, I would argue that there is still a lot to be done before domain-adaptive classifiers become practical, everyday tools. At the moment, there are too many researchers proposing methods to address very specific cases (sometimes even just between two datasets) and only a handful of researchers working on answering theoretical questions. This is a shame, as advances in theory often shape successful methods. With this in mind, the next two subsections present some open questions that I find interesting.

7.1. Validity of the covariate shift assumption

The current assumption in covariate shift, namely $p_T(y \mid x) = p_S(y \mid x)$, might be too restrictive to ever be valid in nature. The assumption is often interpreted as the decision boundary being in the same location in both domains, but considering that they are distributions, the functions need to be equal for the *whole* sample space. Both Figure 2.1 in Chapter 2 and Figure 3.1 from Chapter 3 show examples of the shape of the posterior distributions. Equal class-posterior distributions is a much more difficult condition to satisfy than equal decision boundaries. As such, there are many occasions where the assumption is made but is not actually valid, leading to detrimental performances of importance-weighted classifiers (c.f. Chapter 6).

Fortunately, some experiments have indicated that there is some robustness to a violation of the covariate shift assumption. It would be very interesting to perform a perturbation analysis and see if a less restrictive assumption can be found. This might take the form

of decision boundaries lying within a specified ϵ distance from each other. Or it might be possible to incorporate the distance between decision boundaries as a slack variable, which would influence importance-weight estimation directly. Since a less restrictive condition would be easier to satisfy, methods relying on it would be more robust in practice.

7.2. More specific domain discrepancy metrics

Most measures that describe discrepancies between domains are very general; they are either distribution-free or classifier-agnostic. General measures produce looser generalization bounds than more specific measures. As new insights are gained into causes of domain shift, new, more precise metrics should be developed. These can contain prior knowledge on the problem at hand: for example, in natural language processing, one often encodes text documents in bag-of-word or n-gram features. General measures such as the Maximum Mean Discrepancy might show small values for essentially entirely different contexts. A more specific measure, such as the total variation distance between Poisson distributions, would take the discreteness and sparseness of the feature space into account. Consequently, it would be more descriptive and it would be preferable for natural language processing domains. Such specific forms of domain discrepancy metrics would lead to tighter generalization bounds, stronger guarantees on classifier performance and more practical adaptive classifiers.

Finding domain discrepancies specific to a task or type of data is not a trivial task. A good place to start is to look at methods that incorporate explicit descriptions of their adaptations. For instance, a subspace mapping method explicitly describes what makes the two domains different (e.g. lighting or background). Looking at the types of adaptations they recover would be informative as to what types of discrepancies are useful for specific applications. I think therefore that methods with explicit descriptions of ”transfer” could be exploited for finding more specific domain discrepancy metrics.

7.3. Open access and institution-variation

Being able to classify a dataset by downloading a source domain and training a domain-adaptive classifier instead of annotating samples, can save a tremendous amount of time, money and effort. It increases the value of existing datasets. Not only does it save annotation costs, but it can also increase statistical power by providing more data. I believe the development of domain-adaptive or transfer learning methods, creates a larger incentive for researchers to make their data publicly available.

It is not uncommon to hear that different research groups working in the same field do not use each other’s data. The argument is that the other group is located in a different environment, experiments differently or uses a different measuring device, and that their data is therefore not ”suitable” [3]. For example, in biostatistics, gene expression micro-array data can exhibit ”batch effects” [4]. These can be caused by the amplification reagent, time of day, or even atmospheric ozone level [5]. In some datasets, batch effects are the most dominant source of variation and are easily identified by clustering algorithms

[4]. However, additional information such as local weather, laboratory conditions, or experimental protocol, should be available. That information could be exploited to correct for the batch effect. The more knowledge we have of possible confounding variables, the better we would be able to model the transfer from one batch to the other. Considering the financial costs of genome sequencing experiments, the ability to combine datasets from multiple research centers without batch effects is very desirable. Larger benefits from open access further encourage data sharing.

7.4. Sequential adaptation

Successful adaptation is defined as an improvement over the performance of the original system. As may be understood from this thesis, it is not clear which conditions have to be fulfilled in order for the system to perform well. It seems that in cases where it is difficult to describe how two populations relate to each other, adaptive systems become highly uncertain. Conversely, the more similar the populations are, the likelier it is that the system adapts well. It would, for example, be easier to adapt to predict heart disease in adolescents based on adults, then it would be to adapt to infants. But that raises the question: can we design a system that first adapts to an intermediate population and only then adapts to the final target population? In other words, a system that *sequentially* adapts?

Intermediate domains are often available, but overlooked. When incorporated, these would present a series of changes instead of one large jump. For example, adapting from European hospitals to predicting illnesses in Asian hospitals is difficult. But a sequential adaptive system starting in western Europe would first adapt to eastern Europe, followed by the Middle-East, then to west Asia and finally reaching a population of eastern Asian patients. If the domain shifts are not too dissimilar in each transition, then adaptation should be easier.

Of course, the sequential strategy also raises a number of extra questions: will adaptation errors accumulate? How should the possible performance gain be traded off against the additional computational cost? Will performance feedback be necessary? Some of these questions have been addressed in dynamical learning settings, such as reinforcement learning or multi-armed bandits [6, 7]. The analysis of sequential adaptive systems could build upon their findings.

The sequential adaptation setting also shares some overlap with sequential Monte Carlo sampling, for time series prediction and state-space models [8]. In that setting, a continuous signal that changes its characteristics over time is modeled and extrapolated. One method, called *particle filtering*, actually employs importance-weighting over time [9, 10]. Each sample is weighted with its importance with respect to the signal in the next time-step. However, sudden large changes would cause the same sampling variance problems as discussed in Chapter 3. Considering the similarity to importance-weighing in covariate shift, developments in particle filtering could be very useful to sequential domain adaptation. In summary, a lot is known about dynamical learning, which should not be neglected in designing a sequential domain-adaptive classifier.

7.5. Conclusion

I hope to have convinced the reader that domain adaptation is an interesting topic of research within machine learning and artificial intelligence. Progress in the design and analysis of classifiers that generalize to target domains would be beneficial to all areas where supervised learning is already being used, especially in areas where annotation is expensive and similar datasets are available. There is still a lot of work to be done before these methods become practical, but with it come many exciting challenges as well.

References

[1] C. Cortes, M. Mohri, M. Riley, and A. Rostamizadeh, *Sample selection bias correction theory*, in *Algorithmic Learning Theory* (Springer, 2008) pp. 38–53.

[2] C. Cortes, Y. Mansour, and M. Mohri, *Learning bounds for importance weighting*, in *Advances in Neural Information Processing Systems* (2010) pp. 442–450.

[3] J. T. Leek, R. B. Scharpf, H. C. Bravo, D. Simcha, B. Langmead, W. E. Johnson, D. Geman, K. Baggerly, and R. A. Irizarry, *Tackling the widespread and critical impact of batch effects in high-throughput data*, Nature Reviews: Genetics **11** (2010).

[4] W. E. Johnson, C. Li, and A. Rabinovic, *Adjusting batch effects in microarray expression data using empirical Bayes methods*, Biostatistics **8**, 118 (2007).

[5] T. L. Fare, E. M. Coffey, H. Dai, Y. D. He, D. A. Kessler, K. A. Kilian, J. E. Koch, E. LeProust, M. J. Marton, M. R. Meyer, *et al.*, *Effects of atmospheric ozone on microarray data quality*, Analytical Chemistry **75**, 4672 (2003).

[6] R. S. Sutton and A. G. Barto, *Reinforcement learning: an introduction*, Vol. 1 (MIT Press, 1998).

[7] P. Whittle, *Multi-armed bandits and the gittins index*, Journal of the Royal Statistical Society. Series B , 143 (1980).

[8] O. Cappé, S. J. Godsill, and E. Moulines, *An overview of existing methods and recent advances in sequential monte carlo*, Proceedings of the IEEE **95**, 899 (2007).

[9] A. Doucet, N. De Freitas, K. Murphy, and S. Russell, *Rao-Blackwellised particle filtering for dynamic Bayesian networks*, in *Conference on Uncertainty in Artificial Intelligence* (Morgan Kaufmann Publishers Inc., 2000) pp. 176–183.

[10] P. M. Djuric, J. H. Kotecha, J. Zhang, Y. Huang, T. Ghirmai, M. F. Bugallo, and J. Miguez, *Particle filtering*, IEEE Signal Processing Magazine **20**, 19 (2003).

Notation

Please refer to this list for a description of the mathematical notation in this thesis. Note that individual chapters may deviate when necessary.

$\mathcal{X}$	Input space, for example $\mathbb{R}^D$.
$\mathcal{Y}$	Output space, for example $\{-1, +1\}$ or $\{1, \ldots, K\}$.
D	Dimensionality of input space.
K	Number of classes, i.e. $\lvert\mathcal{Y}\rvert$.
$p_{\mathcal{S}}$	Source probability distribution function.
$p_{\mathcal{T}}$	Target probability distribution function.
$\mathcal{S}$	Source domain: $(\mathcal{X}, \mathcal{Y}, p_{\mathcal{S}})$.
$\mathcal{T}$	Target domain: $(\mathcal{X}, \mathcal{Y}, p_{\mathcal{T}})$.
X	Source data, indexed by i for samples, d for features and k for classes.
x	Source data sample, indexed by d for features.
Z	Target data, indexed by j for samples, d for features and k for classes.
z	Target data sample, indexed by d for features.
y	Source labels $y \in \mathcal{Y}$, indexed by i for samples and k for classes.
u	Target labels $u \in \mathcal{Y}$, indexed by j for samples and k for classes.
$\mathcal{D}_{\mathcal{S}}$	Dataset of labeled source samples: $\mathcal{D}_{\mathcal{S}}^n = \{(x_i, y_i)\}_{i=1}^n$.
$\mathcal{D}_{\mathcal{T}}$	Dataset of labeled target samples: $\mathcal{D}_{\mathcal{T}}^n = \{(z_j, u_j)\}_{j=1}^m$.
$\mathbb{E}$	Expectation, or expected value, of a distribution.
$\mathbb{V}$	Variance of a distribution.
$\mathbb{C}$	Covariance between two variables.
$\mathcal{H}$	Hypothesis space of classification functions.
h	Classification function; $h : \mathcal{X} \to \mathcal{Y}$.
θ	Classification function parameters.
ℓ	Loss function, which compares a prediction $h(x_i)$ to a true label y_i.
R	Risk function, i.e. the expected loss: $R(h) = \mathbb{E}\ell(h)$.
$\hat{R}$	Empirical risk, i.e. the average loss: $\hat{R}(h \lvert \mathcal{D}_{\mathcal{S}}^n)$.
w	Importance weights; $w(x_i)$ or w_i, indexed by i for samples.
L^p	Order of regularization, e.g. L^2-regularization.
λ	Regularization parameter.
D	Divergence between two distributions: $\mathrm{D}(p_{\mathcal{S}}, p_{\mathcal{T}})$.
$\hat{\mathrm{D}}$	Empirical divergence between two datasets: $\hat{\mathrm{D}}(X, Z)$.
d	Distance between two samples; $d(x, z)$.
ϕ	Basis function; $\phi(x)$.
κ	Kernel function; $\kappa(x, x') = \phi(x)\phi(x')^\top$.
$\mathcal{N}$	Normal distribution, i.e. $\mathcal{N}(x \mid \mu, \Sigma)$.

List of Publications

6. W.M. Kouw & M. Loog. *Effects of sampling skewness in importance-weighted risk estimators on model selection*, ArXiv, 1804.07344, 2018.

5. W.M. Kouw & M. Loog. *Reducing sampling variance in covariate shift using control variates*, ArXiv, 1710.06514, 2017.

4. W.M. Kouw, M. Loog, L.W. Bartels & A.M. Mendrik. *MR acquisition-invariant representation learning*, ArXiv, 1709.07944, 2017.

3. W.M. Kouw & M. Loog. *Target contrastive pessimistic risk for robust domain adaptation*, ArXiv, 1706.08082, 2017.

2. W.M. Kouw & M. Loog. *On regularization parameter estimation under covariate shift*, International Conference on Pattern Recognition, 426 - 431, 2016.

1. W.M. Kouw, J.H. Krijthe, M. Loog & L.J.P. van der Maaten. *Feature-level domain adaptation*, Journal of Machine Learning Research, 17 (171), 1-32, 2016.

Acknowledgements

Over the past four years, several people have, directly and indirectly, contributed to this thesis. I would like to acknowledge their support in the following:

First of all, my financial support: this thesis has partly been made possible by a grant by the Netherlands Organization for Scientific Research (NWO; grant 612.001.301) and partly by a grant by the European Commision (FP7-ICT-2011.8.2). Both of these belong to Laurens van der Maaten, without whom my PhD would not have been possible. Laurens, thank you for taking me in and for our interesting discussions.

Next, I would like to thank everyone in the Pattern Recognition & Bioinformatics Group: Ahmed, Alex S, Alex M, Alexey, Amin, Amogh, Arlin, Bart, Bob, Thies, Christian, Christine, David, Ekin, Erdogan, Erik, Gorkem, Hamdi, Hayley, Jan, Jasper, Jeroen, Jesse, Joana, John, Laura, Lu, Marc, Marcel, Marco, Osman, Paola, Robbert, Ruud, Sjoerd, Saskia, Soufiane, Stavros, Sepideh, Seyran, Silvia, Tamim, Taygun, Thies, Thomas, Tom, Veronika, Wenjie, Yanxia, Yazhou, and everyone else. I've learned so much from all of you, thank you for making me feel at home.

Ahmed, Sjoerd, I'm a fan of your work; beautiful brain pics! Erdogan, thank you for your sobering views, it was sorely needed sometimes. Alexey, Amin, thank you for kind hospitality when I came bothering you guys in your office. Thies, for god's sake, shut up about your damn mushrooms. ;) Christian, I've really started to love your humor. Good luck on the rest of your PhD and remember to take a break every now and then. Silvia, you're a dedicated researcher and a great teacher, which I have a lot of respect for. Hamdi, I enjoyed working with you, even if I was a bit difficult at times. Hayley, I've always thought that you're doing really cool work, even if I was critical at times. It's inspiring to see someone with your ambition. Good luck in the future and I still hope to get a collaboration project started at some point. Jan, thank you for all the times you patiently listened to my ideas / rambling, and for all your corny dad jokes. Thomas, your stories about superbugs and some of the biology you guys work on, still scare me to this day. If I had a bit more knowledge on bioinformatics, I would've loved to have done a PhD with you. Marcel, your practical views on science, funding and collaboration have stuck with me, and will be very valuable advice when/if I pursue a professorship of my own. Thank you for that, and for helping me focus.

Ekin, Laura, Sally, there's nothing like blowing off steam with you guys. Thanks for all the laughs! I will *never* forget: Laura trying to make us appreciate tequila, Sally with his freedom flask hidden somewhere, Ekin getting sick in weird ways and places, playing panda-tiger-bear points, the out-of-context quotes on the whiteboard, underground dugtrio's and a lot of stuff that isn't appropriate material for a thesis. But I won't have to remember! ...because there are videos of it... way, way too many videos...

Wenjie, Taygun, Yazhou, Alex, and Tom, thank you for all the exciting discussions, the moments we were sweating in front of a whiteboard and the nerdy PR/ML jokes. David, your enthusiasm and dedication for both pattern recognition and teaching is inspiring. Your questions at talks show us that curiosity is more important than pride. Veronika, thank you for the warm welcome when I started, the practical advice on workshops, conferences, and publishing over the years and of course, the parties and beers. Jesse, the times we were messing around with formulas, talking about all the cool areas of math or computer science that we haven't explored yet or were trying to make sense of difficult papers, were some of the best times of my PhD. I feel very fortunate for having you as an office buddy and a friend.

Marco, it's been a wild ride, and I'm very happy that I shared it with you. You are what every mentor should be: supportive, encouraging, thorough, precise, critical, honest, down-to-earth, inspiring and full of passion for pattern recognition. We're lucky to have someone to discuss ideas with, someone who keeps an eye on scientific principles and integrity, and someone to compete with for the dirtiest and most shocking joke. I feel absolutely no regret for always taking as much time from you as I could, and I'm proud to have been one of your students.

Adriënne, you always manage to cheer me up and motivate me for the task at hand. Our meetings always lead to dozens of ideas; I'm glad our paper turned out as cool as it did and I hope we get to make some more. Finally, I also appreciate our more deeper discussions on what science is and what it should be.

Rob, Sanne, David, en Bas, bedankt voor alle leuke avonden en feesten. Rob, Sanne, ik vond het altijd prachtig om te zien hoe jullie je interesses achterna gingen. Het gaf me het vertrouwen dat ik ook de goede kant op aan het gaan was. Heel veel succes in de toekomst!

Tobias, Tom, ik ben dankbaar dat ik jullie had om alles een beetje te relativeren. Dat was soms hard nodig. Jullie geduld als ik weer te laat of chagrijnig was, is zeer gewaardeerd.

Pap, mam, ik ben heel erg dankbaar voor de kansen die jullie me gegeven hebben om me op wat voor manier dan ook te ontwikkelen. Bedankt ook voor jullie nieuwsgierigheid en dat ik altijd over van alles kan vertellen. Wessel, Jasper, Roland, Lenneke, en Iris, bedankt voor jullie geduld wanneer ik te laat was of als ik weer een (te) lang verhaal opzette. Corlijne, heel knap dat je het met me uitgehouden hebt in Den Haag, terwijl ik zo vaak ongezellig was 's avonds. Ik was in ieder geval erg blij dat je er was. Lidy, heel erg bedankt voor je bemoedigende woorden en dat je zo blij voor me bent als ik met goed nieuws aan kom.

Als laatste, Nika, bedankt dat je er altijd voor me bent en bedankt voor je eindeloze geduld als ik weer met mezelf in de knoop zit. Jij hebt meer vertrouwen in mij dan ikzelf heb, wat me door de moeilijke momenten heen gesleept heeft. Je bent heel speciaal voor mij en ik ben blij dat ik de rest van mijn leven met jou mag delen.

Curriculum Vitæ

Wouter was born on the 11th of March 1989 in Breda. He joined the Stedelijk Gymnasium Breda, with a specialization in Science & Technology plus Computer Science and an external degree in English from Cambridge University. He had many interests, ranging from philosophy and literature to chemistry and aerospace engineering, and due to a robotics project with his father, began focusing on intelligent behavior. Wondering how humans and animals see, decide and learn, he decided to study Biological Psychology in Maastricht. He finished his Bachelor degree with an exchange semester at the University of Guelph, in Canada, with a thesis on neural circuits that integrate multi-modal sensory information.

Continuing his studies on neural information processing, he was admitted to the Research Master program in Cognitive Neuroscience in Maastricht. After a research assistantship in pattern analysis of functional MRI data of visual information processing, he started an internship at the Computational Vision & Neuroscience Group of the Werner Reichardt Centre for Integrative Neuroscience in Tübingen, Germany. For his master thesis, Wouter worked on a biologically plausible computational model of directing eye-movement based on visual saliency.

His experiences with statistical models, probability distributions and algorithms during his internship, made Wouter decide to pursue machine intelligence for his doctoral research. He found a position at the Department of Intelligent Systems at TU Delft, where he worked in pattern recognition and machine learning. Projects included the design of domain-adaptive classifiers, analyses of sampling variance and validation under covariate shift. After 2,5 years, he visited Cornell University to work on the intersection of sample selection bias and causal inference.

After his PhD, Wouter became a Research Software Engineer at the Netherlands eScience Center, where he worked on a number of projects: CT-scan bone detection for 3D printing of body parts, segmentation of waterways in aerial photographs, recognizing violent events in transcribed interviews of WWII survivors and mixing neural networks to mimic radiologists' disagreement in radiotherapy treatment planning. In October 2017 his research proposal for the Niels Stensen Fellowship was honored and he will start his post-doctoral research in June 2018.