TECHNICAL REPORT

An introduction to domain adaptation and transfer learning

Wouter M. Kouw, Marco Loog

Delft University of Technology
Van Mourik Broekmanweg 6, 2628 XE Delft, the Netherlands
wmkouw@gmail.com

Abstract. In machine learning, if the training data is an unbiased sample of an underlying distribution, then the learned classification function will make accurate predictions for new samples. However, if the training data is *not* an unbiased sample, then there will be differences between how the training data is distributed and how the test data is distributed. Standard classifiers cannot cope with changes in data distributions between training and test phases, and will not perform well. *Domain adaptation* and *transfer learning* are sub-fields within machine learning that are concerned with accounting for these types of changes. Here, we present an introduction to these fields, guided by the question: when and how can a classifier generalize from a source to a target domain? We will start with a brief introduction into risk minimization, and how transfer learning and domain adaptation expand upon this framework. Following that, we discuss three special cases of data set shift, namely prior, covariate and concept shift. For more complex domain shifts, there are a wide variety of approaches. These are categorized into: importance-weighting, subspace mapping, domain-invariant spaces, feature augmentation, minimax estimators and robust algorithms. A number of points will arise, which we will discuss in the last section. We conclude with the remark that many open questions will have to be addressed before transfer learners and domain-adaptive classifiers become practical.

Keywords: Machine Learning · Pattern Recognition · Domain Adaptation · Transfer Learning · Covariate Shift · Sample Selection Bias.

1 Introduction

Intelligent systems learn from data to recognize patterns, predict outcomes and make decisions [113,80]. In data-abundant problem settings, such as recognizing objects in images, these systems achieve super-human levels of performance [95]. Their strength lies in their ability to process large amounts of examples and obtain a detailed estimate of what *does* and *does not* constitute the object of interest. Machine learning has become popular due to large-scale data collection and open access of data sets. Learning algorithms are now incorporated into self-driving cars [201], drone guidance [81], computer-assisted diagnosis [122], online commerce [142], satellite cartography [207], exo-planet discovery [9], and machine translation [216], among others.

"Intelligence" refers to a computer's ability to *learn* to perform a task [165]. Supervised systems learn through *training*, where the system is rewarded or punished based on whether it produces the right output for a given input [25,151]. In order to train an intelligent system, a set of matching inputs and outputs is required. Most often, inputs consist of complicated objects such as images while outputs consists of decisions such as 'yes' or 'no', or classes such as 'healthy', 'at risk', and 'disease'. The system will consider many classification functions on the set of inputs and select the function that produced the smallest error cost. If the examples in the data set 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 'disease'. Then, a system can be trained by finding the decision function that produces the best diagnoses. If the patients are an accurate reflection of the population of all possible patients, then the trained system will produce accurate diagnoses for 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. For example, data collected from older patients is biased with respect to the total human population. If data is biased, then the system will think that certain outcomes are more likely to occur. For example, it might consider certain levels of blood pressure to be normal, when they would actually indicate a health risk for younger patients. 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 [98,97,100]. However, generalizing towards wider populations is perhaps too ambitious. Instead, machine learning researchers are targeting specific other populations. For instance, can we use information from *adult* humans to train an intelligent system for diagnosing *infant* heart disease?

Such problem settings are known as *domain adaptation* or *transfer learning* settings [16,160,158]. The population of interest is called the *target domain*, for which labels are usually not available and training a classifier is not possible. However, if data from a similar population 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* or a *transfer learner* (the difference will be defined in Section 3). Generalizing across distributions is difficult and it is not clear which conditions have to be satisfied for a classifier to perform well. We therefore focus on the question: when and how can a statistical classifier generalize from a source to a target domain?

1.1 Relevance

A more detailed example of adaptation is the following: in clinical imaging settings, radiologists manually annotate tissues, abnormalities, and pathologies 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 [196,79,123]. Consequently, systems trained on data from other scanners would fail to perform well on the new scanner. An adaptive system could find correspondences in images between scanners, and change its decisions accordingly. Thus it avoids the time, funds and energy needed to annotate images from the new scanner [196,79,125].

Other examples of domain adaptation and transfer learning in fields that employ machine learning include: in bioinformatics, adaptive approaches have been successful in sequence classification [205,149], gene expression analysis [38,210], and biological network reconstruction [153,118]. Most often, domains correspond to different model organisms or different data-collecting research institutes [211]. 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 [35]. The system will have to adapt to the new setting, until another component is replaced. 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 [81,207]. Computer vision systems that recognize activities have to adapt across different surroundings as well as different groups of people [195,93,75]. 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 [27]. Similarly, online movie reviews are linguistically different from tweets [161]. Sentiment classification relies heavily on context as well; different words are used to express whether someone likes a book versus whether an electronic gadget [30,96].

In some situations, the target is a sub-population of the source domain. For instance, in *personalized* systems, the target is a single individual while the source might be a set of individuals. One of the first types of personalized systems are spam filters: they are often initialized using data from many individuals but will adapt to specific users over time [154]. Male users receive different kinds of spam than female users for instance, which the system can detect based purely on text statistics. Alternatively, in speaker recognition, an initial speaker-independent system can adapt to new speakers [135]. Similarly, general face recognition systems can be adapted to specific persons [214] and person-independent activity recognition algorithms can be specialized to particular individuals [76].

1.2 Outline

This report is guided by the questions: *when* and *how* can a classifier generalize from a source to a target domain? The matter of *when* is discussed through generalization error bounds in Sections 2.1, 3.2 and 5.1, and through types of data shifts in Section 4. The matter of *how* is discussed in Section 5, where we categorize a series of approaches.

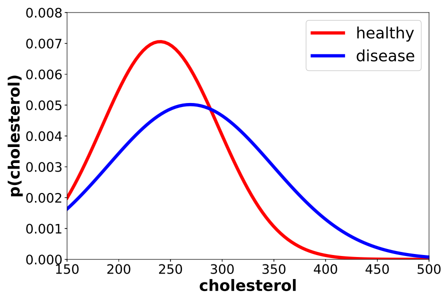
The remainder of the report is structured as follows: Firstly, we will give a brief overview of statistical classification in the following Section. Readers familiar with the material may skip this Section. Secondly, Section 3 describes how domain adaptation and transfer learning fit within the risk minimization framework. We present stricter definitions of "domains" and "adaptation" as well. Thirdly, Section 4 discusses various types of simple data set shifts. Simple in the sense that various aspects of the resulting distributions remain constant between domains and one does not require fully labeled data in each domain. Fourthly, Section 5 presents an overview of approaches for more complex cases of domain shifts. We discuss popular algorithms briefly, but bear in mind that this list is not exhaustive. Lastly, Section 6 discusses a few ideas and questions that have come to mind while reviewing the literature, and Section 7 draws a few conclusions.

2 Classification

This section is a brief overview of classification and risk minimization. For broader overviews, see [70,141]. Readers familiar with this material may skip to Section 3.

2.1 Risk minimization

One of the most well-known frameworks for the design, construction and analysis of intelligent systems is *risk minimization* [70,151]. In order to represent an object digitally, we measure one or more *features*. For example, we measure the level of cholesterol of patients in a hospital study. A feature captures information about the object; some levels occur more often than others. These variations over cholesterol x can be described by a probability distribution $p(x)$. Now, suppose we would like to predict whether these patients will develop heart disease. In order to decide between the prognosis 'healthy' and the prognosis 'disease', the system must determine which of the two is more probable for a given level of cholesterol, i.e. $p(\text{healthy}|x) > p(\text{disease}|x)$ or $p(\text{healthy}|x) < p(\text{disease}|x)$ [187]. Figure 1 (left) describes two probability distributions as a function of cholesterol level; the red distribution corresponds to 'healthy' and the blue to 'disease'.



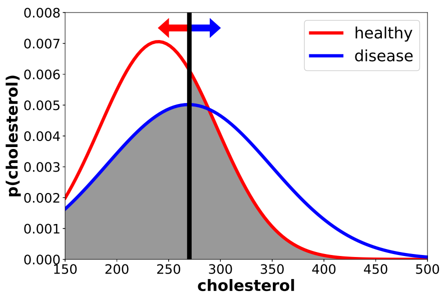


Fig. 1. Example of a classification problem. (Left) Probability distributions of healthy and ill patients, as a function of their levels of cholesterol. (Right) Classifier, with the black line indicating its decision boundary, and the error consisting of the gray shaded area under the distributions.

A decision-making problem can be abstractly described as the task of assigning 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. $h(x) = \text{healthy}$. Its output is 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 heart disease prognosis problem, a classification function for a 1-dimensional problem can be seen as a threshold, illustrated in Figure 1 (right) by the black vertical line. It designates everything to the left as 'healthy' and everything to the right as 'disease'. Hence, all heart disease patients left of the line and all healthy patients to the right are misclassified. Note that the current threshold need not be optimal. The classification error is visualized as the gray region under the distributions and can be written as:

$$\begin{aligned} e(h) = &\int_{\mathcal{X}} [h(x) \neq \text{healthy}]\ p(x \mid \text{healthy})\ p(\text{disease})\ \mathrm{d}x \\ &+ \int_{\mathcal{X}} [h(x) \neq \text{disease}]\ p(x \mid \text{disease})\ p(\text{disease})\ \mathrm{d}x\,, \end{aligned}$$

where $h(x)$ refers to the decision made by the classifier. $p(\text{healthy})$ and $p(\text{disease})$ refer to the probability of encountering healthy patients and heart disease patients in general, while $p(x \mid \text{healthy})$ and $p(x \mid \text{disease})$ refer to the probabilities of observing a particular level of cholesterol x given that the patient is healthy or has a disease (also known as the *class-conditional* distributions), respectively.

The classifier should be able to make a decision over all possible cholesterol levels $\mathcal{X}$. Since cholesterol is a continuous variable, the decision function is integrated over all possible levels. 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 healthy patient will develop heart disease and the second term describes how often it thinks that a patient with heart disease will be healthy. Summing these two terms constitutes the overall classification error $e(h)$.

If 'healthy' and 'disease' are encoded as a variable y, then the classification error can be written as follows:

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

where $p(x, y) = p(x \mid y)p(y)$. $\mathcal{Y}$ numerically represents the set of classes, in this case $\mathcal{Y} = \{\text{healthy} = -1, \text{disease} = +1\}$. Objects are often not described by one feature but by multiple measured properties. As such, x is a D-dimensional random vector $\mathcal{X} \subseteq \mathbb{R}^D$.

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's prediction for a certain object $h(x)$ and the object's true class y. The classification error, $e(h)$, is also known as the 0/1 *loss*, denoted $\ell_{0/1}$. It has value 0 whenever the prediction is equal to the true label and value 1 whenever they are not equal; $\ell_{0/1}(h(x), y) = [h(x) \neq y]$. However, this function is hard to work with; the function has a constant and discontinuous derivative, which means that gradient-based methods cannot be applied. One would have to evaluate each classifier separately in order to find the best one. Most often, there are infinitely many classifiers to consider, which implies infinitely many evaluations.

Other loss functions are the *quadratic / squared* loss, $\ell_{\text{qd}}(h(x), y) = (h(x) - y)^2$, the *logistic* loss $\ell_{\log}(h(x), y) = yh(x) - \log \sum_{y' \in \mathcal{Y}} \exp(y'h(x))$ 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 through a convex function [13]. Convex loss functions are easier to optimize, but do not necessarily lead to the same solution than if the error function was optimized directly. Overall, the choice of a loss function can have 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:

$$R(h) = \mathbb{E}_{\mathcal{X},\mathcal{Y}} \left[\ell(h(x), y)\right] , \tag{1}$$

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:

$$h^* = \arg\min_h \, \mathbb{E}_{\mathcal{X},\mathcal{Y}} \left[\ell(h(x), y)\right] .$$

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 efficient optimization procedures can be used [34].

Empirical risk 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 data set $\mathcal{D}^n = \{(x_i, y_i)\}_{i=1}^n$, 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:

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

$\hat{R}$ is called the *empirical risk* function. It evaluates classifiers *given* a particular data set. Note that the true risk R from Equation 1 does not depend on observed data. Minimizing the empirical risk with respect to a classifier for a particular data set, is called *training* the classifier:

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

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.

In general, 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 [202,32]. It can be shown that the resulting classifier will converge to the optimal classifier [197,151]. 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, and the optimization error, i.e. the difference between the true minimizer and the one obtained through the optimization procedure [33,151].

Generalization Ultimately, we are not interested in the error of the trained classifier on the given data set, but in the error on all possible future samples $e(h) = \mathbb{E}_{\mathcal{X},\mathcal{Y}}[h(x) \neq y]$. The difference between the true error and the empirical error is known as the *generalization error*: $e(h) - \hat{e}(h)$ [11,151]. Ideally, we would like to know if the generalization error will be small, i.e., that our classifier will be *approximately correct*. However, because classifiers are functions of data sets, and data sets are random, we can only describe how *probable* it is that our classifier will be approximately correct. We can say that, with probability $1 - \delta$, where $\delta > 0$, the following inequality bolds (Theorem 2.2 from [151]):

$$e(h) - \hat{e}(h) \ \leq \ \sqrt{\frac{1}{2n}\Big(\log |\mathcal{H}| + \log \frac{2}{\delta} \Big)} \, . \tag{2}$$

where $|\mathcal{H}|$ denotes the cardinality of the finite hypothesis space, or the number of classification functions that are being considered [193,119,151]. This result is known as a Probably Approximately Correct (PAC) bound. In words, the difference between the true error, $e(h)$, and the empirical error, $\hat{e}(h)$, of a classifier is less than the square root of the logarithm of the size of the hypothesis space $|\mathcal{H}|$, plus the log of 2 over δ, normalized by twice the sample size n. In order to achieve a similar result for the case of an infinite hypothesis space (e.g. linear classifiers), a measure of the complexity of the hypothesis space is required.

Generalization error bounds are interesting because they analyze what a classifier's performance depends on. In this case, it suggest choosing a smaller or simpler hypothesis space when the sample size is low. Many variants of bounds exist. Some use different measures of complexity, such as Rademacher complexity [14] or Vapnik-Chervonenkis dimensions [20,197], while others use concepts from Bayesian inference [147,131,15].

Bounds can incorporate assumptions on the problem setting [12,151,54]. For example, one can assume that the posterior distributions in each domain are equal and obtain a bound for a classifier that exploits that assumption (c.f. Equation 6). Assumptions restrict the problem setting, i.e., settings where that assumption is invalid are disregarded. This often means that the bound is tighter and a more accurate description of the behaviour of the classifier can be found. Such results have inspired new algorithms in the past, such as Adaboost or the Support Vector Machine [69,47].

Regularization Generalization error bounds tell us that the complexity, or flexibility, of a classifier has to be traded off with the number of available training samples [61,197,54]. In particular, a flexible model can minimize the error on a given data set completely, but will be too specific to generalize to new samples. This is known as *overfitting*. Figure 2 (left) illustrates an example of 2-dimensional classification problem with 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 [188,25]. Figure 2 (middle) visualizes an example of a properly regularized classifier, that will probably generalize well to new samples. Figure 2 (right) shows an example of a too heavily regularized classifier, also known as an "underfitted" classifier.

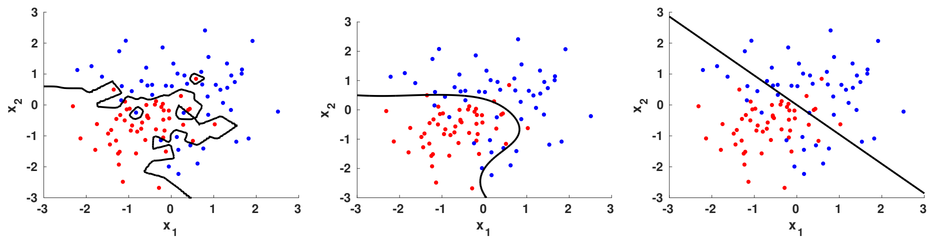


Fig. 2. Examples of classifier complexities. (Left) Overfitted classifier, (middle) well-fitted classifier, (right) underfitted classifier.

3 Domain adaptation and transfer learning

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 $\mathbb{R}^D$ and are sometimes referred to as feature vectors, or points in feature space. Outputs are classes, which can be binary, in which case $\mathcal{Y}$ corresponds to $\{-1, +1\}$, or 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. *Transfer learning* is defined as the general case where the domains are freely allowed to differ in sample space, label space, distribution or all. 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 feature spaces [117,88]. *Domain adaptation* is defined as the particular case where the sample and label spaces remain unchanged and only the probability distributions change.

3.1 Notation

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 $\mathcal{T}$. For example, $p_\mathcal{T}(x, y)$ 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 data set is referred to as $\{(x_i, y_i)\}_{i=1}^{n}$. 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 data set $\{(z_j, u_j)\}_{j=1}^{m}$. All vectors are row vectors, except if otherwise stated. Capital letters denote matrices, for example X as the whole data set of n source samples of D-dimensional vectors.

3.2 Cross-domain generalization error

Returning to the question: *when* can a classifier generalize from a source to a target domain? The generalization error bound from Section 2.1 describes how much a classifier trained on samples from a distribution will generalize to new samples from that distribution. But it 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 [102,151,32]. Since Hoeffding's inequality does not hold in a cross-domain setting, the standard generalization error bound does not hold either.

Nonetheless, it is possible to derive generalization error bounds if more is known on the relationship between $\mathcal{S}$ and $\mathcal{T}$ [4,29,43,144,16,215,78]. For example, one of the first error bounds relies on the condition that the ideal hypothesis on both domains has low error [17,16]. 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 joint error. If it 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 [16]:

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

where the probability Pr can be computed through integration: $\mathrm{Pr}_{\mathcal{S}}[h \neq h'] = \int_{\mathcal{X}} [h(x) \neq h'(x)] p_{\mathcal{S}}(x) \mathrm{d}x$. 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 [120,17,16].

Given the error of the ideal joint hypothesis, $e^*_{\mathcal{S},\mathcal{T}} = \min_{h \in \mathcal{H}} \ [e_{\mathcal{S}}(h) + e_{\mathcal{T}}(h)]$, and the $\mathcal{H}\Delta\mathcal{H}$-divergence, a bound on the difference between the true target error, $e_{\mathcal{T}}$ of a trained source classifier, $\hat{h}_{\mathcal{S}} = \arg\min_h \hat{R}_{\mathcal{S}}(h)$, and that of the optimal target classifier, $h^*_{\mathcal{T}} = \arg\min_h R_{\mathcal{T}}(h)$, can be found. This bound has the following form (Theorem 3, [16]):

$$e_{\mathcal{T}}(\hat{h}_{\mathcal{S}}) - e_{\mathcal{T}}(h^*_{\mathcal{T}}) \ \leq e^*_{\mathcal{S},\mathcal{T}} + \frac{1}{2} d_{\mathcal{H}\Delta\mathcal{H}}(p_{\mathcal{S}}, p_{\mathcal{T}}) + \mathcal{C}(\mathcal{H}) \, ,$$

which holds with probability $1 - \delta$, for $\delta > 0$. $\mathcal{C}(\mathcal{H})$ 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 [198]. Overall, this bound states that, the larger $e^*_{\mathcal{S},\mathcal{T}}$ and $d_{\mathcal{H}\Delta\mathcal{H}}$ are for a given problem setting, the less a source classifier will generalize to the target domain.

But the above bound describes the performance of a *non-adaptive* classifier. Now, the challenge is to devise an adaptation strategy that leads to tighter generalization error bounds. This will not possible for the general case, but will be possible if the problem can be simplified through restrictions on the difference between domains. In the following, we discuss common simple data set shifts, for which adaptation strategies have been proposed with generalization error bounds.

4 Common data shifts

We are ultimately interested in minimizing the target risk $R_{\mathcal{T}}$. So how does the source domain relate to this? One of the most straightforward ways to incorporate the source distribution in the target risk is as follows:

$$\begin{aligned} R_{\mathcal{T}}(h) &= \sum_{y \in Y} \int_{\mathcal{X}} \ell(h(x) \mid y)\ p_{\mathcal{T}}(x,y)\ \mathrm{d}x \\ &= \sum_{y \in Y} \int_{\mathcal{X}} \ell(h(x) \mid y)\ p_{\mathcal{T}}(x,y)\ \frac{p_{\mathcal{S}}(x,y)}{p_{\mathcal{S}}(x,y)}\ \mathrm{d}x \\ &= \sum_{y \in Y} \int_{\mathcal{X}} \ell(h(x) \mid y)\ p_{\mathcal{S}}(x,y)\ \frac{p_{\mathcal{T}}(x,y)}{p_{\mathcal{S}}(x,y)}\ \mathrm{d}x\,. \end{aligned} \tag{3}$$

Which can be estimated using the sample average with:

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

Note that the samples are drawn according to the source distribution, not the target distribution: $(x_i, y_i) \sim p_{\mathcal{S}}(x, y)$. $p_{\mathcal{T}}(x_i, y_i)$ describes the probability of those source samples under the target distribution. To estimate these probabilities, we would need labeled data from both domains, which is not available.

Joint distributions can be decomposed in two ways: $p(x, y) = p(x \mid y)p(y)$ and $p(x, y) = p(y|x)p(x)$. If only one of the components differs between domains, then the resulting setting is a special case of data set shift. In $p(x, y) = p(x \mid y)p(y)$, the class-conditional distributions are weighted by the prior distribution, i.e. the relative proportion of each class. If the conditional distributions remain constant and only the prior distributions differ, then it is said to be a case of "prior shift". In $p(x, y) = p(y \mid x)p(x)$, there are two special cases. In the first, the posterior distributions are equivalent but the data distributions differ between domains. This is known as "covariate shift". If the data distributions remain constant, but the posteriors change, then it is a case of "concept shift". For prior and covariate shift, it is not necessary to obtain labeled data in both domains. The following subsections discusses these three simple types of shifts in more detail.

4.1 Prior shift

For prior shift, the prior probabilities of the classes are different, $p_{\mathcal{S}}(y) \neq p_{\mathcal{T}}(y)$, but the conditional distributions are equivalent, $p_{\mathcal{S}}(x|y) = p_{\mathcal{T}}(x|y)$. This can occur in for example fault detection settings, where a new maintenance policy might cause less faults [110], or in the detection of oil spills before versus after an incident [128].

Figure 3 illustrates an example of prior shift. The prior probabilities for the positive and negative class are both 1/2 in the source domain, but 2/3 versus 1/3 in the target domain. The data and posterior distributions in each domain remain equivalent, which results in a change in the conditional distributions. The positive class outweighs the negative class in probability.

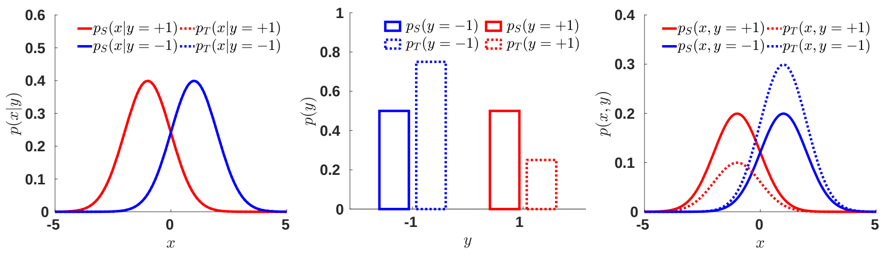


Fig. 3. An example of prior shift. (Left) The conditional distributions in each domain are different. (Middle) The prior distribution changes from 1/2 for both classes in the source domain to 3/4 and 1/4 in the target domain. (Right) The joint distribution for the source domain is balanced, while the negative class outweighs the positive class in the target domain.

The knowledge that the conditionals are equivalent can be exploited by canceling them out of the ratio of joint probability distributions [167]:

$$R_{\mathcal{T}}(h) = \sum_{y \in Y} \int_{\mathcal{X}} \ell(h(x), y) \frac{\cancel{p_{\mathcal{T}}(x \mid y)}\ p_{\mathcal{T}}(y)}{\cancel{p_{\mathcal{S}}(x \mid y)}\ p_{\mathcal{S}}(y)} p_{\mathcal{S}}(x, y)\ \mathrm{d}x \,, \tag{4}$$

where the ratio $p_{\mathcal{T}}(y)/p_{\mathcal{S}}(y)$ represent the change in class proportions. Using samples drawn from the source distribution and a function that re-weights each class, we can estimate the target risk using the following estimator:

$$\hat{R}_{\mathcal{T}}(h) = \frac{1}{n} \sum_{i=1}^{n} \ell(h(x_i), y_i) \ w(y_i) \, .$$

Using this approach, no unlabeled target samples are necessary, only target labels.

Class-based weighting has been extensively studied from an alternative perspective: when it is more difficult to collect data from one class than the other [94]. For example, in a few countries, women above a certain age are given the opportunity to be tested for breast cancer [168]. The vast majority that responds does not show signs of cancerous tissue and only a small minority is tested positive. On this data set, the classifier could always predict 'healthy' and achieve a low error. But then it misses exactly the cases of interest. To avoid this unwanted behaviour, samples from the rare class could be re-weighted to be more important [62,37,63].

4.2 Covariate shift

Covariate shift is one of the most studied forms of data set shift. It occurs most often when there is a form of sample selection bias [98,133,100]. Selection bias is defined as the altered probability of being sampled [97,42]. For example, suppose you would visit a city where most people live in the center and the habitation density decreases as a function of the distance from the center. You are interested in whether people think that the city is overpopulated. If you would sample on the main square you will mostly encounter people who live in the center, and you would likely get a lot of 'yes' answers. Inhabitants who live further away, who would say 'no', are under-represented in the data. The results of your survey will likely differ from if you would sample door-to-door.

From a domain adaptation perspective, the biased sampling corresponds to the source domain and the target domain to the unbiased sample. Re-weighting individual samples would correspond to correcting the over-representation of people living in the center and under-representation of people living further away.

Similar to sample selection bias, another cause for covariate shift is missing data [164,137]. In practice, data can be missing as measurement devices fail or because of subject dropout. 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. The collected results are dependent on the probability of 'not-missing', and adaptation consists of correcting for under- or over-observed samples.

Figure 4 presents an example of a case of covariate shift. On the left are shown the data distributions in each domain. The source distribution is centered on zero, while the target is centered on -1. The posterior distributions are the same (middle), but the joint distributions differ.

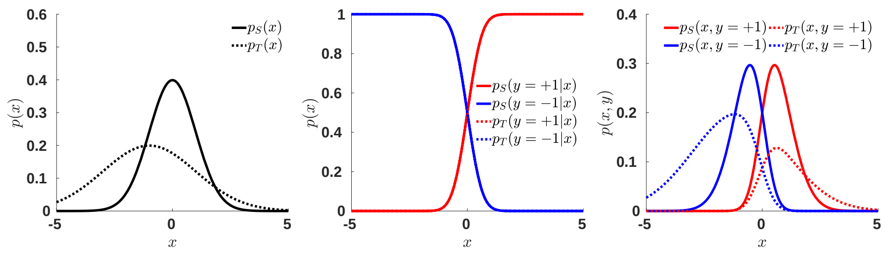


Fig. 4. An example of covariate shift. (Left) The target data distribution has shifted away from the source data distribution. (Middle) The posterior distributions are equal. (Right) The resulting joint distributions are shifted as well.

The knowledge that the posterior distributions are equivalent can be exploited by canceling them in the ratio of joint distributions in (3):

$$R(h) = \sum_{y \in Y} \int_{\mathcal{X}} \ell(h(x), y) \frac{\cancel{p_{\mathcal{T}}(y \mid x)}\ p_{\mathcal{T}}(x)}{\cancel{p_{\mathcal{S}}(y \mid x)}\ p_{\mathcal{S}}(x)} p_{\mathcal{S}}(x, y)\ \mathrm{d}x\,, \tag{5}$$

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

4.3 Concept shift

In the case of concept shift, the data distributions remain constant while the posteriors change. For instance, 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. In [2], the classes are originally defined as "remission" and "complications". But, at test time, other aspects are counted as a form of "complication" and are so labeled. What constitutes the positive and negative class, and by extension the posterior distributions, has changed.

Figure 5 shows an example of setting with concept shift. In this case, the posterior distribution in the target domain has shifted away from the source distribution, towards the negative part of feature space. In the resulting joint distribution, the decision boundary has shifted to the left as well.

The knowledge that the data distributions remain equal can be exploited through:

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

But adaptation in this setting is still impossible without labeled target data. To estimate conditional distributions, one requires simultaneous observations of both variables.

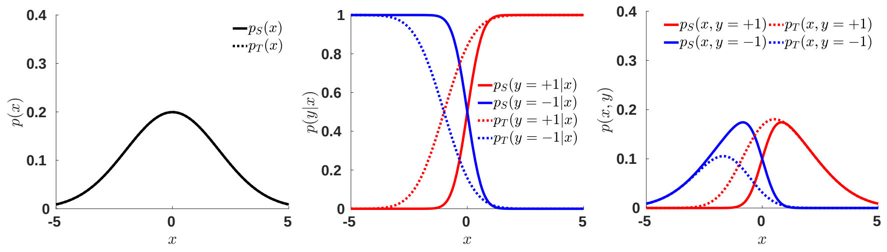


Fig. 5. An example of concept shift. (Left) The data distributions remain constant. (Middle) The target posterior distributions are shifted to the left of the source posteriors. (Right) The resulting joint distributions are shifted as well.

Concept shift is related to *data drift*, where classifiers are deployed in non-stationary environments [206]. For smoothly varying non-stationarities, such as time-series, there is again additional information that can be exploited: the shifts are ordered and are relatively small between neighboring time steps. If the classifier receives feedback after each time step, the drift can be modeled and the next timestep's shift can be predicted [55,72,57,41].

4.4 General domain shifts

In general cases of domain shift, one or more of the above shifts will have occurred. As can be imagined, this is the most difficult setting and learning will often not be possible at all [18,19]. 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 [17,18,16,26], constrained worst-case labellings [203,138], low-data-divergence [17,18,16], the existence of a domain manifold [86,7,160], conditional independence of class and target given source data [126] and unconfoundedness [109]. Some of these domain relationships are discussed in the following Section.

5 Approaches

There are many approaches to transfer learning and domain adaptation. Some of the more prominent ones are discussed here.

5.1 Importance-weighting

Weighting samples based on their importance to the target domain is mostly used in covariate shift. Although the step in Equation 5 seems straightforward, one still needs to show that an importance-weighted classifier will learn, i.e. improve with more samples. For that, a generalization error bound is needed

[44]. The difference between the true target error of a classifier, $e_{\mathcal{T}}(h)$, and the empirical weighted source error, $\hat{e}_{\mathcal{W}}(h)$, is (c.f. Theorem 3, [44]):

$$e_{\mathcal{T}}(h) - \hat{e}_{\mathcal{W}}(h) \leq \ 2^{5/4} \ \sqrt{\mathrm{D}_2(p_{\mathcal{T}} \| p_{\mathcal{S}})} \ \sqrt[3/8]{\frac{c}{n} \log \frac{2ne}{c} + \frac{1}{n} \log \frac{4}{\delta}} \,, \tag{6}$$

which holds with probability at least $1 - \delta$, for $\delta > 0$. $\mathrm{D}_{2R}(p_{\mathcal{T}} \| p_{\mathcal{S}})$ is the 2-order Rényi divergence [194], an information-theoretic domain discrepancy measure, and c refers to the pseudo-dimension of $\mathcal{H}$, a measure of describing the complexity of the hypothesis space [199]. c and $\mathrm{D}_{2R}(p_{\mathcal{T}} \| p_{\mathcal{S}})$ are required to be finite and the weights cannot be equal to 0. $\mathrm{D}_{2R}(p_{\mathcal{T}} \| p_{\mathcal{S}})$ will diverge to infinity when the domains are too far apart, in which case the above generalization error bound does not hold.

We can now see that the difference between the adaptive classifier obtained using importance-weighting depends on the complexity of the classifier, the sample size and the divergence between the domains. In other words: for a fixed choice of hypothesis space (e.g. linear classifiers), as the divergence between the domains increases, the sample size needs to increase at a certain rate as well, in order to maintain the same probability of being approximately correct.

Weight estimation Given that importance-weighting is a valid strategy for domain adaptation, the question is how to estimate importance weights appropriately. Depending on the problem setting, some methods estimate the numerator and denominator of the ratio of probabilities separately, and others estimate the ratio directly.

Each probability distribution in the ratio can be estimated using a Gaussian distribution [175]. But this tends to have a negative effect on the variance of the importance weights [46,44,45]. For example, suppose 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 $\sigma^2_{\mathcal{T}}$. Then, the weights consist of $p_{\mathcal{T}}(x)/p_{\mathcal{S}}(x) = \ \mathcal{N}(x \,|\, 0, \sigma^2_{\mathcal{T}}) \ / \ \mathcal{N}(x \,|\, 0, 1) = \ \sigma^{-1}_{\mathcal{T}} \exp(x^2(-1 + \sigma^2_{\mathcal{T}})/(2\sigma^2_{\mathcal{T}}))$. 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 nearly-zero weights. Consequently, at training time, the classifier will focus on this one important sample and will effectively ignore the others. The resulting classifier is often pathological and will not generalize well.

A non-parametric alternative is to use kernel density estimation (KDE) [23,213,7]. In KDE, a distribution is estimated by placing a kernel function on top of each sample. This kernel function is 1 in its center and drops off exponentially based on the distance away from the center. The probability density of any point in the space is the average kernel distance to all known points. KDE can be used to estimate weights by first estimating the density of a source sample with respect to all target samples. Secondly, the density of the source samples is estimated and lastly, the target density is divided by the source density.

The advantage of KDE is that it is possible to control the variance properties of the resulting ratio through the design of the kernel. It contains what is known as a *kernel bandwidth* parameter. This bandwidth parameter roughly corresponds to how wide the kernel is. It is a hyperparameter, and although it gives some control, it is not clear how it should be set.

Instead of estimating the data distribution in each domain separately, one could estimate the ratio directly. Methods that directly estimate importance weights are usually based on minimizing some type of discrepancy between the weighted source and the target distributions: $\mathrm{D}\left[w, p_{\mathcal{S}}, p_{\mathcal{T}}\right]$ [184]. However, just minimizing this objective with respect to w might cause highly varying or unusually scaled values [190]. This unwanted behaviour can be combated through incorporating a property of the reweighed source distribution:

$$\begin{aligned} 1 &= \int_{\mathcal{X}} p_{\mathcal{T}}(x) \mathrm{d}x \\ &= \int_{\mathcal{X}} w(x) p_{\mathcal{S}}(x) \mathrm{d}x \\ &\approx \frac{1}{n} \sum_{i=1}^{n} w(x_i)\, , \end{aligned} \tag{7}$$

for samples drawn from the source distribution, $x_i \sim p_{\mathcal{S}}$. 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: $\mid n^{-1} \sum_i^n w(x_i) - 1 \mid \leq \epsilon$.

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:

$$\begin{aligned} \underset{w \in W}{\text{minimize}} \quad & \mathrm{D}\left[w, p_{\mathcal{S}}, p_{\mathcal{T}}\right] \\ \text{s.t.} \quad & w_i \geq 0 \\ & \mid \frac{1}{n} \sum_i^n w_i - 1 \mid \leq \epsilon\, . \end{aligned} \tag{8}$$

Note that w is now a variable and not a function of the sample $w(x)$. Depending on the choice of discrepancy measure, this optimization problem could be linear, quadratic or be even further constrained.

A popular measure of domain discrepancy is the Maximum Mean Discrepancy, which is based on the two-sample problem from statistics [68,31,90]. It measures the distance between two means after subjecting the samples to the continuous function that pulls them maximally apart. In practice, kernel functions are used, which, under certain conditions, are able to approximate any continuous function arbitrary well [5,172,174,22]. The empirical discrepancy measure, including the reweighed source samples, can be expressed as [89]:

$$\hat{\mathrm{D}}^2_{\mathrm{MMD}}[w, X, Z] = \frac{1}{n^2} \sum_{i,i'}^{n} w_i \kappa(x_i, x_{i'}) w_{i'} - \frac{2}{mn} \sum_{i}^{n} \sum_{j}^{m} w_i \kappa(x_i, z_j) + C\,, \tag{9}$$

where κ are the kernel functions and C are constants not relevant to the optimization problem. Minimizing the empirical MMD with respect to the importance weights, is called Kernel Mean Matching (KMM) [108,89]. Depending on if the weights are upper bounded, convergence rates for KMM can be computed [90,89,213]

Another common measure of distribution discrepancies is the Kullback-Leibler Divergence [130,48,143], leading to the Kullback-Leibler Importance Estimation Procedure (KLIEP) [182,180,183]. The KL-divergence between the true target distribution and the importance-weighted source distribution can be simplified as:

$$\begin{aligned}\mathrm{D}_{\mathrm{KL}}\left[w, p_{\mathcal{S}}, p_{\mathcal{T}}\right] &= \int_{\mathcal{X}} p_{\mathcal{T}}(x) \log \frac{p_{\mathcal{T}}(x)}{p_{\mathcal{S}}(x) w(x)} \mathrm{d}x \\ &= \int_{\mathcal{X}} p_{\mathcal{T}}(x) \log \frac{p_{\mathcal{T}}(x)}{p_{\mathcal{S}}(x)} \mathrm{d}x - \int_{\mathcal{X}} p_{\mathcal{T}}(x) \log w(x) \mathrm{d}x\,. \end{aligned} \tag{10}$$

Since the first term in the right-hand side of (10) 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: $\mathbb{E}_{\mathcal{T}}[\,\log w(x)] \approx m^{-1} \sum_{j}^{m} \log w(z_j)$. The weights are formulated as a functional model consisting of an inner product of weights α and basis functions ϕ, i.e. $w(x) = \alpha^\top \phi(x)$ [181]. This allows them to apply the importance-weight function to both the test samples in the KLIEP objective from (10) and to the training samples for the constraint in (7).

One could also minimize the squared error between the estimated weights and the actual ratio of distributions [116,115]:

$$\begin{aligned}\mathrm{D}_{\mathrm{LS}}[w, p_{\mathcal{S}}, p_{\mathcal{T}}] =& \frac{1}{2} \int_{\mathcal{X}} \left(w(x) - \frac{p_{\mathcal{T}}(x)}{p_{\mathcal{S}}(x)} \right)^2 p_{\mathcal{S}}(x) \mathrm{d}x \\ =& \frac{1}{2} \int_{\mathcal{X}} w(x)^2 p_{\mathcal{S}}(x) \mathrm{d}x - \int_{\mathcal{X}} w(x) p_{\mathcal{T}}(x) \mathrm{d}x + C\,, \end{aligned} \tag{11}$$

where C is again a term containing constants not relevant to the optimization problem. 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, $w(x) = \alpha^\top \phi(x)$, gives $1/2\ \alpha^\top \mathbb{E}_{\mathcal{S}}[\phi(x)\phi(x)^\top]\alpha - \mathbb{E}_{\mathcal{T}}[\phi(x)]$. Replacing the expected values with sample averages allows for plugging in this objective into the nonparametric weight estimator in (8). This technique is called Least-Squares Importance Fitting (LSIF).

Lastly, directly estimating importance weights can be done through tessellating the feature space into Voronoi cells [178,155,140,127]. Each cell is a polygon of variable size and denotes an area of equal probability [150,132]. 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. To estimate importance weights, first, one forms the Voronoi cell V_i of each source sample x_i. The cell consists of the part of feature space that lies closest to x_i and can be obtained through a 1-nearest-neighbour procedure. The ratio of target over source is then approximated by counting the number of target samples z_j that lie within the cell: $w(x_i) = |V_i \cap \{z_j\}_{j=1}^m|$, where $\cup$ denotes the intersection between the Voronoi cell and the set of target samples and $| \cdot |$ denotes the cardinality of this set.

It does not require hyperparameter optimization, but there is still the option to perform Laplace smoothing, where a 1 is added to each cell [66]. This ensures that no source samples are given a weight of 0 and are thus completely discarded.

5.2 Subspace mappings

Domains may lie in different subspaces [189,65], in which case there would exist a mapping from one domain to the other [88,160]. For example, the mapping may correspond to a rotation, an affine transformation, or a more complicated nonlinear transformation [65,159]. In problems in computer vision and natural language processing, the data can be very high-dimensional and the domains might look completely different from each other. For example, photos of animals in zoos versus pictures of animals online. Unfortunately, using a too flexible transformation to capture this mapping, can lead to overfitting. This means the method will work well for the given target samples but fail for new target samples. Note that any structural relationships between domains, such as equal posterior distributions will no longer be valid if the domains are mapped separately.

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 rotate the source components to align with the target components: $C_{\mathcal{S}}W$ [65]. W is the linear transformation matrix, which consists of the transposed source components times the target components $W = C_{\mathcal{S}}^\top C_{\mathcal{T}}$. However, each domain contains noise components and these should not affect the matching procedure. Subspace Alignment (SA) can be used to find an optimal subspace dimensionality [65]. This technique is attractive because its flexibility is limited, making it quite robust to unusual problem settings. It is computationally not expensive, easily implemented and intuitive to explain. It has been extended a number of times: there is a landmark-based kernelized alignment [3], a subspace distribution alignment technique [185] and a semi-supervised variant [212].

Instead of aligning the domains based on directions of variance, one could model the structure of the data with graph-based methods [50,51]. Data is first summarized using a subset called the exemplars, obtained through clustering. From the set of exemplars, two sets of hyper-graphs are constructed. These two hyper-graphs are matched along their first, second and third orders [50,51]. First-order matching finds a linear transformation matrix that minimizes the difference between the mapped source and the target hyper-graphs, similar to Subspace Alignment. Higher-order moments consider other forms of geometric and structural information, beyond pairwise distances between exemplars [52,50]. These can be found using tensor-based algorithms [50,60].

Metrics A number of methods aim to find transformations that aid specific subsequent classifiers [166,129,77]. For example, Information-Theoretic Metric Learning (ITML) [166]. In ITML, a Mahalanobis metric is learned for use in a later nearest-neighbour classifier [53,166]. Metrics describe ways of computing distances between points in vector spaces. The standard Euclidean metric, $d_E^2(x,z) = (x-z)(x-z)^\top$, consists of the inner product of the difference between two vectors. The Mahalanobis metric, $d_W^2(x,z) = (x-z)^\top W(x-z)$, weights the inner product by the curvature of the space W. In fact, first transforming the space and then measuring distances with the Euclidean metric is equivalent to measuring distances with the Mahalanobis metric according to the square root of the transformation matrix W: $(W^{1/2}x - W^{1/2}z)^\top (W^{1/2}x - W^{1/2}z) = (x-z)^\top W(x-z)$. Hence, first transforming the space and then training a nearest-neighbour classifier is equivalent to training a Mahalanobis nearest-neighbour.

In order to avoid that the Mahalanobis metric pushes samples from different classes closer together for the sake of mapping source to target, it is necessary to include some constraints [10]. If a small number of target labels is available, then these could be used as correspondence constraints. They 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 data sets. One descriptor might span a source feature space of dimensionality $D_\mathcal{S}$ while another spans a feature space of dimensionality $D_\mathcal{T}$. ITML is symmetric in the sense that it requires both domains to be of the same dimensionality. It is hence a domain adaptation technique, according to our definition in Section 3. But it 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 $W^{D_\mathcal{S} \times D_\mathcal{T}}$ to find general mappings between feature spaces of different dimensionalities [129,105]. It is hence a transfer learning approach, according to our definitions.

Another means of estimating a cross-domain metric is to use the Fisher criterion. 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, $\bar{\mu}$ 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 [67,148]. The Fisher criterion can be used to extract a set of basis vectors that maintains class separability, much like a supervised form of PCA [146]. FIDOS, a FIsher based feature extraction method for DOmain Shift [56], is its extension to minimize domain separability while maintaining class separability. 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 has a trade-off parameter to fine-tune the balance between the two objectives.

Outside of estimating metrics for subsequent metric-based classifiers, there are also techniques that align class margins for subsequent maximum-margin classifiers [103].

Manifolds One can make an even stronger assumption than that of the existence of a mapping from source to target domain: that there exists a manifold of transformations between the source and target domain [83,7,104,217]. A manifold is a curved lower-dimensional subspace embedded in a larger vector space. A transformation manifold corresponds to a space of parameters, where each point generates a possible domain. For example, it might consist of a set of camera optics parameters, where each setting would measure the data in one vector space basis [86,7]. This assumption is useful when selecting a set of transformations as a geodesic path along the manifold [85,84,87,6,36].

One of the first approaches to incorporate a transformation manifold looked at the idea of learning incremental small subspace transformations instead of a single large transformation [86]. Incremental learning was originally explored in situations with time-dependent concept shifts [169]. In our 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 D-dimensional vector space can be described by the *Grassmann* manifold [208,1,191,218]. Each point on the Grassmannian generates a basis that forms a subspace [85,218,84]. One can move from one subspace to another by following the path along the surface of the manifold, also known as a geodesic. Computing the direction of the geodesic is performed using matrix exponentials or something called spline flow [71,36]. Given the geodesic, 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. But the iteration can be replaced by integrating over the entire path. This produces a Geodesic Flow Kernel (GFK) which can be used in a nearest-neighbour classifier [85,84].

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 [7]. Geodesics along the statistical manifold describe how to transform one probability distribution into another. The length of the geodesic along the statistical manifold, called the Hellinger distance, can be used as a measure of domain discrepancy [99,57]. The Hellinger distance is closely related to the total variation distance between two distributions, which is used in a number of other papers [17,16,145]. Adaptation consists of importance-weighting samples or transforming parameters to minimize the Hellinger distance [7,8].

5.3 Finding domain-invariant spaces

The problem with transformations between domains is that the classifier remains in a domain-specific representation. Ideally, we would like to represent the data in a space which is domain-invariant.

Most domain-invariant projection techniques stem from the computer vision and (biomedical) imaging communities, where domain adaptation and transfer learning problems 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 [6,103]. In medical imaging, there exists a true representation of a patient and each MR scanner's image is a different noisy observation [196,123]. Since properties of the acquisition device are known, it is possible to design techniques specific to each type of instrument.

Instead of finding principal components in each domain, one could also find common components. If the domains are similar in a few components, then both sets could be projected onto those components. In order to find common components where the sets are not too dissimilar, the Maximum Mean Discrepancy is employed [157,159]. First, the MMD measure is rewritten as a joint domain kernel K [156]. Now, a projection C is introduced, which could be of lower dimensionality than the original number of features. From the joint domain kernel, components are extracted by minimizing the size of all projected data, where size corresponds to the *trace* of the resulting matrix. The trivial solution to such a problem set up is that the projection maps everything to 0. To avoid this, a constraint is added ensuring that the projection remains confined to a certain size. The resulting optimization problem becomes:

$$\begin{aligned} \underset{C}{\text{minimize}} \quad & \text{trace}(C^\top \text{KLK} C) \\ \text{s.t.} \quad & C^\top \text{KHK} C = I \, , \end{aligned} \tag{12}$$

where 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 [159]. A regularization term trace$(C^\top C)$ along with a trade-off parameter, can be added as well. The solution to this optimization problem is an eigenvalue decomposition [170,171].

The Domain-Invariant Projection approach from [6], later renamed to Distribution Matching Embedding (DME) [8], aims to find a projection matrix W that minimizes the MMD as follows:

$$\hat{\mathrm{D}}^2_{\mathrm{DME}}[w, X, Z] = \frac{1}{n^2} \sum_{i,i'}^{n} \kappa(x_i W, x_{i'} W) - \frac{2}{mn} \sum_{i}^{n} \sum_{j}^{m} \kappa(x_i W, z_j) + C \, ,$$

where C is a set of constant terms. The projection is constrained to remain orthonormal; $W^\top W = I$. Note that this formulation is similar to Kernel Mean Matching (c.f. Equation 9), except that the projection operation is inside the kernel as opposed to the weighting function which is outside the kernel.

Although the MMD encourages moments of distributions to be similar, it does not encourage smoothness in the new space. To this end, a regularization term can be added that punishes the within-class variance in the domain-invariant space. Futhermore, DME is still limited by its use of a linear projection matrix. A nonlinear projection is more flexible and more likely to recover a truly domain-invariant space. The Nonlinear Distribution-Matching Embedding achieves this by performing the linear projection in kernel space; $\phi(x)W$ [8]. An alternative to finding a projection with the MMD is to use a Hellinger distance instead [8].

Alternatively, MMD's kernel can be learned as well [152]. Instead of using a universal kernel to measure discrepancy, it is also possible to find a basis function for which the two sets of distributions are as similar as possible. Considering that different distributions generate different means in kernel space, it is possible to describe a distribution of kernel means [179,22]. The variance of this meta-distribution, termed *distributional variance*, should then be minimized. 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 [92,121]. Maintaining this central subspace during optimization ensures that classes remain separable in the new domain-invariant space. Overall, as this approach finds kernel components that minimize distributional-variance, it is coined Domain-Invariant Component Analysis (DICA) [152]. It has been expanded on for the specific case of spectral kernels by [139].

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. Transfer Subspace Learning aims to find that subspace through minimizing a Bregman divergence to both domains [176].

The reconstruction error from the subspace to original space should be minimal [173]. Interestingly, this objective is similar to that of an autoencoder, where the mapping is constructed through a neural network: $\|\phi(\phi(xW)W^{-1}) - z\|$ [101]. Deep autoencoders stack multiple layers of nonlinear functions to achieve flexible transformations [200]. They are mostly used in settings where large amounts of data are available [82,39].

Although the kernel approaches have the capacity to recover any nonlinear mapping, they require multiple computations of matrices as large as the number of samples. They therefore do not scale well. For larger data sets, one might employ neural networks. They also have the capacity to recover any nonlinear mapping but scale much better in terms of the number of samples [24,163]. 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 layer complexity and stacking multiple layers on top of each other, under certain conditions, any possible transformation can be achieved [49,107,106,12]. Its optimization procedure, known as backpropagation, finds a transformation to a space in which the data is maximally linearly separable. By using different loss functions in the top layer, different forms of transformations can be achieved [73]. Domain-Adversarial Neural Networks (DANN) have one classification-error minimizing top layer and one domain-error maximizing top layer [74]. Essentially, the network finds a domain-invariant space when its domain classifier cannot recognize from which domain a new sample has come. The idea of maximizing domain-confusion while minimizing classification error has inspired more approaches [192,114].

5.4 Feature augmentation

In natural language processing (NLP), text corpora show different word frequency statistics. People express themselves differently in different contexts [28,111]. For instance, the word 'useful' occurs more often to denote positive sentiment in kitchen appliance reviews than in book reviews [30]. 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 [26]. Domains present a large problem for the field, as a document or sentence system cannot generalize across context.

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 [177]. 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. The word in the target domain that correlates most with the pivot word is said to correspond structurally to the original word in the source domain word. It could be a good indicator of positive versus negative sentiment as well [27]. Adaptation consists of augmenting the bag-of-words encoding with pivot words, learning correspondences and training on the augmented feature space [27,26,40,136].

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 [30,26]. 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 [40,136].

5.5 Minimax estimators

When any of the aforementioned approaches are applied to settings where their assumptions are violated, then adaptation could be detrimental to performance [162,124]. To ensure a robust level of performance, a worst-case setting can be assumed. Worst-case settings are often formalized as minimax optimization problems [21,91]. The variable that we want to be robust against is maximized before the risk is minimized with respect to classifier parameters. Minimax estimators behave conservatively and are not as powerful as standard estimators when their assumptions are valid.

A straightforward example of a minimax estimator is the Robust Bias-Aware classifier [138]. In this case, the uncertain quantity corresponds to the target domain's posterior distribution. However, given full freedom, the adversary would flip the labels of the classifier in every round to maximize the classifier's risk and convergence would not be achieved. A constraint is imposed, telling the adversary that it needs to pick posteriors that match the moments of the source distribution's feature statistics:

$$\begin{aligned} \min_{h \in \mathcal{H}} \max_{g \in \mathcal{H}} \quad & \frac{1}{m} \sum_{j=1}^{m} \ell(h(z_j), g(z_j)) \\ \text{s.t.} \quad & \frac{1}{m} \sum_{j=1}^{m} g(z_j) = \frac{1}{n} \sum_{i=1}^{n} y_i \, , \end{aligned} \tag{13}$$

where the constraint states that the first-order moment (i.e. the sample average) of the adversary's posteriors should be equal to first-order moment of the source label distribution. Higher-order moments can be included as constraints as well. This minimax estimator returns high confidence predictions in regions with high probability mass under the source distribution and uniform class predictions in regions with low source probability mass.

Importance-weighted classifiers are sensitive to poorly estimated weights. To construct a robust version, one could minimize risk with respect to worst-case weights [203]:

$$\begin{aligned} \min_{h \in \mathcal{H}} \max_{w \in W} \quad & \frac{1}{n} \sum_{i=1}^{n} \ell(h(x_i), y_i) w_i \\ \text{s.t.} \quad & w_i \geq 0 \\ & \mid \frac{1}{n} \sum_{i}^{n} w_i - 1 \mid \leq \epsilon \, . \end{aligned}$$

The weights will have to be constrained as otherwise they would tend to infinity. These constraints match those of the non-parametric weight estimators (c.f. Equation 8). By training a classifier under worst-case weights, it will behave more conservatively.

5.6 Robust algorithms

Conservatism can also be expressed through *algorithmic robustness* [209]. 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.

This notion can be employed to construct a robust adaptive algorithm [145]. Such an algorithm will try to find a separating hyperplane such that the hinge loss on the source set is similar to the loss on the target set. Essentially, the classifier is discouraged from picking support vectors in areas with low probability mass under the target distribution. 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.

6 Discussion

A number of points come to mind, as well as ideas for future work.

6.1 Validity of the covariate shift assumption

The current assumption in covariate shift, namely $p_{\mathcal{T}}(y \,|\, x) = p_{\mathcal{S}}(y \,|\, x)$, might be too restrictive to ever be valid in nature. The assumption is often interpreted as that the decision boundary should be in the same location in both domains, but that is false. The posterior distributions need to be equal for the *whole* feature space. 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.

Fortunately, some experiments have indicated that there is some robustness to a violation of the covariate shift assumption. It would be interesting to perform a perturbation analysis to study a classifier's behaviour as a function of the deviation from equal posteriors. Perhaps decision boundaries that lie within a specified ϵ distance from each other, are not too far apart to change a classifier's behaviour. That would mean importance-weighting is more widely applicable than its assumption implies. A less restrictive condition would be easier to satisfy, and methods that depend on it would be less at risk of negative transfer.

6.2 Application-specific discrepancies

Most measures that describe discrepancies between domains are 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, 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. Methods with explicit descriptions of "transfer" could be exploited for finding more specific domain discrepancy metrics.

6.3 Open access and institution-variation

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" [134]. For example, in biostatistics, gene expression micro-array data can exhibit "batch effects" [112]. These can be caused by the amplification reagent, time of day, or even atmospheric ozone level [64]. In some data sets, batch effects are the most dominant source of variation and are easily identified by clustering algorithms

[112]. 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 of possible confounding variables, the better the ability to model the transfer from one batch to the other. Considering the financial costs of genome sequencing experiments, the ability to combine data sets from multiple research centers without batch effects is desirable.

This can be taken a step further: being able to classify a data set by downloading a source domain and training a domain-adaptive classifier instead of annotating samples, will save everyone time, funds and effort. Not only does it increase the value of existing data sets, it can also increase statistical power by providing more data. The development of domain-adaptive or transfer learning methods creates a larger incentive for researchers to make their data publicly available.

6.4 Sequential adaptation

"Successful" adaptation is defined as an improvement over the performance of the original system, i.e. a positive transfer. But settings where the domains are too dissimilar are too difficult. The more similar the populations are, the likelier it is that the system adapts well. But that raises the question: can a system be designed that first adapts to an intermediate population and only then adapts to the final target population? In other words, a system that *sequentially* adapts?

When incorporated, data from intermediate domains 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.

Note that this differs from the domain manifolds approach in Section 5.2, in that those assume that there exists a space of transformations mapping source to target and everything in between. Here, that is not necessary; intermediate domain data could assist any type of method, not just mappings. For instance, it would be possible to perform a series of importance-weighting steps. Such an approach overlaps with *particle filtering* somewhat [59,58]. Particle filters weight the signals in each time-step with respect to the signals in the next time-step. Considering the similarity to importance-weighing in covariate shift, particle filters could be directly applicable to sequential domain adaptation.

But the sequential strategy also raises a number of extra questions: will adaptation errors accumulate? How should performance gain be traded off against 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 [186,204]. The design of sequential adaptive systems could build upon their findings, but more work shall need to be done.

7 Conclusion

We posed the question: when and how can a classifier generalize from a source to a target domain? In the most general case, where there are no constraints on the difference between the source and target domain, no guarantees of performance above chance level can be given. For particular cases, certain guarantees can be made, but these still depend on the domain dissimilarity. Such dependencies lead to many new questions.

Covariate shift is perhaps the most constrained case. In its simplest form, only one covariate / feature changes. This has been extensively studied and many questions, such as how the generalization error depends on the distance between domains, have been addressed. It seems that the most important thing to check before attempting a method is: does the assumption of equal class-posterior distributions hold and if not, how strongly is it violated?

Following that, one could select an importance weight estimator based on the following questions: are the domains so far apart that the weights will become bimodal? Do you have enough source and target samples for complex estimators? If weight estimation is done parametrically, do you have enough samples to prevent low probabilities in the denominator. If done non-parametrically, do you have enough samples to perform hyperparameter estimation?

But a number of additional questions remain: 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? Does the assumption of equal class-posterior distributions hold for only a part of feature space?

General cases of domain shift are more complicated. Multiple factors change at the same time and cannot be always be uniquely identified. For example, if there are both changes in the data distributions and changes in the class-posterior distributions, then these cannot be identified without *some* target labels. Since domain shifts are less constrained, they are harder to study. It is thus difficult to predict whether a specific adaptive method will be successful for a given 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 error of the ideal joint hypothesis, etc. It would be informative to study these factors.

In conclusion, we would argue that there is still a lot to be done before transfer learners and domain-adaptive classifiers become practical tools.

Acknowledgements

The author would like thank Marcel Reinders, Tom Viering and Soufiane Mouragui for feedback and discussions.

References

1. P-A Absil, Robert Mahony, and Rodolphe Sepulchre. Riemannian geometry of grassmann manifolds with a view on algorithmic computation. *Acta Applicandae Mathematicae*, 80(2):199–220, 2004.
2. Rocío Alaiz-Rodríguez and Nathalie Japkowicz. Assessing the impact of changing environments on classifier performance. In *Conference of the Canadian Society for Computational Studies of Intelligence*, pages 13–24. Springer, 2008.
3. Rahaf Aljundi, Rémi Emonet, Damien Muselet, and Marc Sebban. Landmarks-based kernelized subspace alignment for unsupervised domain adaptation. In *Conference on Computer Vision and Pattern Recognition*, pages 56–63, 2015.
4. Rie Kubota Ando and Tong Zhang. A framework for learning predictive structures from multiple tasks and unlabeled data. *Journal of Machine Learning Research*, 6(Nov):1817–1853, 2005.
5. Nachman Aronszajn. Theory of reproducing kernels. *Transactions of the American Mathematical Society*, 68(3):337–404, 1950.
6. M Baktashmotlagh, MT Harandi, BC Lovell, and M Salzmann. Unsupervised domain adaptation by domain invariant projection. In *International Conference on Computer Vision*, pages 769–776, 2013.
7. Mahsa Baktashmotlagh, Mehrtash Harandi, Brian Lovell, and Mathieu Salzmann. Domain adaptation on the statistical manifold. In *Conference on Computer Vision and Pattern Recognition*, pages 2481–2488, 2014.
8. Mahsa Baktashmotlagh, Mehrtash Harandi, and Mathieu Salzmann. Distribution-matching embedding for visual domain adaptation. *Journal of Machine Learning Research*, 17(1):3760–3789, 2016.
9. Nicholas M Ball and Robert J Brunner. Data mining and machine learning in astronomy. *International Journal of Modern Physics D*, 19(07):1049–1106, 2010.
10. Aharon Bar-Hillel, Tomer Hertz, Noam Shental, and Daphna Weinshall. Learning a Mahalanobis metric from equivalence constraints. *Journal of Machine Learning Research*, 6(Jun):937–965, 2005.
11. Peter Bartlett. Prediction algorithms: complexity, concentration and convexity. In *IFAC Symposium on System Identification*, pages 1507–1517, 2003.
12. Peter 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(2):525–536, 1998.
13. Peter L Bartlett, Michael I Jordan, and Jon D McAuliffe. Convexity, classification, and risk bounds. *Journal of the American Statistical Association*, 101(473):138–156, 2006.
14. Peter L Bartlett and Shahar Mendelson. Rademacher and Gaussian complexities: risk bounds and structural results. *Journal of Machine Learning Research*, 3(Nov):463–482, 2002.
15. Luc Bégin, Pascal Germain, François Laviolette, and Jean-Francis Roy. PAC-Bayesian theory for transductive learning. In *Artificial Intelligence and Statistics*, pages 105–113, 2014.
16. Shai Ben-David, John Blitzer, Koby Crammer, Alex Kulesza, Fernando Pereira, and Jennifer Wortman Vaughan. A theory of learning from different domains. *Machine Learning*, 79(1-2):151–175, 2010.
17. Shai Ben-David, John Blitzer, Koby Crammer, and Fernando Pereira. Analysis of representations for domain adaptation. In *Advances in Neural Information Processing Systems*, pages 137–144, 2007.

18. Shai Ben-David, Tyler Lu, Teresa Luu, and Dávid Pál. Impossibility theorems for domain adaptation. In *International Conference on Artificial Intelligence and Statistics*, pages 129–136, 2010.
19. Shai Ben-David and Ruth Urner. On the hardness of domain adaptation and the utility of unlabeled target samples. In *Algorithmic Learning Theory*, pages 139–153. Springer, 2012.
20. Shai Bendavid, Nicolo Cesabianchi, David Haussler, and Philip M Long. Characterizations of learnability for classes of (0,..., n)-valued functions. *Journal of Computer and System Sciences*, 50(1):74–86, 1995.
21. James O Berger. *Statistical decision theory and Bayesian analysis*. Springer, 2013.
22. Alain Berlinet and Christine Thomas-Agnan. *Reproducing kernel Hilbert spaces in probability and statistics*. Springer, 2011.
23. Steffen Bickel, Michael Brückner, and Tobias Scheffer. Discriminative learning under covariate shift. *Journal of Machine Learning Research*, 10:2137–2155, 2009.
24. Christopher M Bishop. *Neural networks for pattern recognition*. Oxford University Press, 1995.
25. Christopher M Bishop. *Pattern recognition and machine learning*. Springer, 2006.
26. J Blitzer, S Kakade, and DP Foster. Domain adaptation with coupled subspaces. In *International Conference on Artificial Intelligence and Statistics*, pages 173–181, 2011.
27. J Blitzer, R McDonald, and F Pereira. Domain adaptation with structural correspondence learning. In *Empirical Methods in Natural Language Processing*, pages 120–128, 2006.
28. John Blitzer. *Domain adaptation of natural language processing systems*. PhD thesis, University of Pennsylvania, 2008.
29. John Blitzer, Koby Crammer, Alex Kulesza, Fernando Pereira, and Jennifer Wortman. Learning bounds for domain adaptation. In *Advances in Neural Information Processing Systems*, pages 129–136, 2008.
30. John Blitzer, Mark Dredze, Fernando Pereira, et al. Biographies, bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification. In *Association for Computational Linguistics*, volume 7, pages 440–447, 2007.
31. Karsten M Borgwardt, Arthur Gretton, Malte J Rasch, Hans-Peter Kriegel, Bernhard Schölkopf, and Alex J Smola. Integrating structured biological data by kernel maximum mean discrepancy. *Bioinformatics*, 22(14):e49–e57, 2006.
32. Stéphane Boucheron, Gábor Lugosi, and Pascal Massart. *Concentration inequalities: a nonasymptotic theory of independence*. Oxford University Press, 2013.
33. Olivier Bousquet and Léon Bottou. The tradeoffs of large scale learning. In *Advances in Neural Information Processing Systems*, pages 161–168, 2008.
34. Stephen Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge University Press, 2004.
35. Wahyu Caesarendra, Gang Niu, and Bo-Suk Yang. Machine condition prognosis based on sequential monte carlo method. *Expert Systems with Applications*, 37(3):2412–2420, 2010.
36. Rui Caseiro, Joao F Henriques, Pedro Martins, and Jorge Batista. Beyond the shortest path: unsupervised domain adaptation by sampling subspaces along the spline flow. In *Conference on Computer Vision and Pattern Recognition*, pages 3846–3854, 2015.
37. Nitesh V Chawla, Kevin W Bowyer, Lawrence O Hall, and W Philip Kegelmeyer. Smote: synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*, 16:321–357, 2002.

38. Austin Chen and Zone-Wei Huang. A new multi-task learning technique to predict classification of leukemia and prostate cancer. *Medical Biometrics*, pages 11–20, 2010.
39. M Chen, Z Xu, F Sha, and KQ Weinberger. Marginalized denoising autoencoders for domain adaptation. In *International Conference on Machine Learning*, pages 767–774, 2012.
40. Minmin Chen, Kilian Q Weinberger, and John Blitzer. Co-training for domain adaptation. In *Advances in Neural Information Processing Systems*, pages 2456–2464, 2011.
41. Yuxin Chen, S Hamed Hassani, Amin Karbasi, and Andreas Krause. Sequential information maximization: When is greedy near-optimal? In *Conference on Learning Theory*, pages 338–363, 2015.
42. William G Cochran and Donald B Rubin. Controlling bias in observational studies: a review. *Sankhyā: Indian Journal of Statistics, Series A*, pages 417–446, 1973.
43. 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*, pages 169–178, 2015.
44. Corinna Cortes, Yishay Mansour, and Mehryar Mohri. Learning bounds for importance weighting. In *Advances in Neural Information Processing Systems*, pages 442–450, 2010.
45. Corinna Cortes and Mehryar Mohri. Domain adaptation and sample bias correction theory and algorithm for regression. *Theoretical Computer Science*, 519:103–126, 2014.
46. Corinna Cortes, Mehryar Mohri, Michael Riley, and Afshin Rostamizadeh. Sample selection bias correction theory. In *Algorithmic Learning Theory*, pages 38–53. Springer, 2008.
47. Corinna Cortes and Vladimir Vapnik. Support vector machine. *Machine learning*, 20(3):273–297, 1995.
48. Thomas M Cover and Joy A Thomas. *Elements of information theory*. John Wiley & Sons, 2012.
49. G Cybenko. Approximation by superposition of sigmoidal functions. *Mathematics of Control, Signals and Systems*, 2(4):303–314, 1989.
50. Debasmit Das and C.S. George Lee. Unsupervised domain adaptation using regularized hyper-graph matching. In *International Conference on Image Processing*, pages 3758–3762, 10 2018.
51. Debasmit Das and CS George Lee. Graph matching and pseudo-label guided deep unsupervised domain adaptation. In *International Conference on Artificial Neural Networks*, pages 342–352. Springer, 2018.
52. Debasmit Das and CS George Lee. Sample-to-sample correspondence for unsupervised domain adaptation. *Engineering Applications of Artificial Intelligence*, 73:80–91, 2018.
53. Jason V Davis, Brian Kulis, Prateek Jain, Suvrit Sra, and Inderjit S Dhillon. Information-theoretic metric learning. In *International Conference on Machine Learning*, pages 209–216, 2007.
54. Luc Devroye, László Györfi, and Gábor Lugosi. A probabilistic theory of pattern recognition, 2013.
55. Thomas 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)*, pages 15–30. Springer, 2002.

56. Cuong V Dinh, Robert PW Duin, Ignacio Piqueras-Salazar, and Marco Loog. Fidos: A generalized fisher based feature extraction method for domain shift. *Pattern Recognition*, 46(9):2510–2518, 2013.
57. Gregory Ditzler and Robi Polikar. Hellinger distance based drift detection for nonstationary environments. In *IEEE Symposium on Computational Intelligence in Dynamic and Uncertain Environments*, pages 41–48, 2011.
58. Petar M Djuric, Jayesh H Kotecha, Jianqui Zhang, Yufei Huang, Tadesse Ghirmai, Mónica F Bugallo, and Joaquin Miguez. Particle filtering. *IEEE Signal Processing Magazine*, 20(5):19–38, 2003.
59. Arnaud Doucet, Nando De Freitas, Kevin Murphy, and Stuart Russell. Rao-Blackwellised particle filtering for dynamic Bayesian networks. In *Conference on Uncertainty in Artificial Intelligence*, pages 176–183. Morgan Kaufmann Publishers Inc., 2000.
60. Olivier Duchenne, Francis Bach, In-So Kweon, and Jean Ponce. A tensor-based algorithm for high-order graph matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(12):2383–2395, 2011.
61. Andrzej Ehrenfeucht, David Haussler, Michael Kearns, and Leslie Valiant. A general lower bound on the number of examples needed for learning. *Information and Computation*, 82(3):247–261, 1989.
62. Charles Elkan. The foundations of cost-sensitive learning. In *International Joint Conference on Artificial Intelligence*, volume 17, pages 973–978, 2001.
63. Andrew Estabrooks, Taeho Jo, and Nathalie Japkowicz. A multiple resampling method for learning from imbalanced data sets. *Computational Intelligence*, 20(1):18–36, 2004.
64. Thomas L Fare, Ernest M Coffey, Hongyue Dai, Yudong D He, Deborah A Kessler, Kristopher A Kilian, John E Koch, Eric LeProust, Matthew J Marton, Michael R Meyer, et al. Effects of atmospheric ozone on microarray data quality. *Analytical Chemistry*, 75(17):4672–4675, 2003.
65. B Fernando, A Habrard, M Sebban, and T Tuytelaars. Unsupervised visual domain adaptation using subspace alignment. In *International Conference on Computer Vision*, pages 2960–2967, 2013.
66. David A Field. Laplacian smoothing and delaunay triangulations. *International Journal for Numerical Methods in Biomedical Engineering*, 4(6):709–712, 1988.
67. Ronald A Fisher. The statistical utilization of multiple measurements. *Annals of Human Genetics*, 8(4):376–386, 1938.
68. Robert Fortet and E Mourier. Convergence de la répartition empirique vers la répartition théorique. In *Annales Scientifiques de l'École Normale Supérieure*, volume 70, pages 267–285, 1953.
69. Yoav Freund and Robert E Schapire. A desicion-theoretic generalization of on-line learning and an application to boosting. In *European Conference on Computational Learning Theory*, pages 23–37. Springer, 1995.
70. Jerome Friedman, Trevor Hastie, and Robert Tibshirani. *The elements of statistical learning*, volume 1. Springer, 2001.
71. Kyle A Gallivan, Anuj Srivastava, Xiuwen Liu, and Paul Van Dooren. Efficient algorithms for inferences on grassmann manifolds. In *IEEE Workshop on Statistical Signal Processing*, pages 315–318, 2003.
72. João Gama and Gladys Castillo. Learning with local drift detection. In *International Conference on Advanced Data Mining and Applications*, pages 42–55. Springer, 2006.

73. Yaroslav Ganin and Victor Lempitsky. Unsupervised domain adaptation by backpropagation. In *International Conference on Machine Learning*, pages 1180–1189, 2015.
74. Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. Domain-adversarial training of neural networks. *Journal of Machine Learning Research*, 17(59):1–35, 2016.
75. Ekin Gedik and Hayley Hung. Speaking status detection from body movements using transductive parameter transfer. In *International Joint Conference on Pervasive and Ubiquitous Computing*, pages 69–72. ACM, 2016.
76. Ekin Gedik and Hayley Hung. Personalised models for speech detection from body movements using transductive parameter transfer. *Personal and Ubiquitous Computing*, pages 1–15, 2017.
77. Bo Geng, Dacheng Tao, and Chao Xu. Daml: domain adaptation metric learning. *IEEE Transactions on Image Processing*, 20(10):2980–2989, 2011.
78. Pascal Germain, Amaury Habrard, François Laviolette, and Emilie Morvant. A new PAC-Bayesian perspective on domain adaptation. In *International Conference on Machine Learning*, pages 859–868, 2016.
79. Mohsen Ghafoorian, Alireza Mehrtash, Tina Kapur, Nico Karssemeijer, Elena Marchiori, Mehran Pesteie, Charles RG Guttmann, Frank-Erik de Leeuw, Clare M Tempany, Bram van Ginneken, et al. Transfer learning for domain adaptation in MRI: application in brain lesion segmentation. *arXiv:1702.07841*, 2017.
80. Zoubin Ghahramani. Probabilistic machine learning and artificial intelligence. *Nature*, 521(7553):452–459, 2015.
81. Alessandro Giusti, Jérôme Guzzi, Dan C Cireşan, Fang-Lin He, Juan P Rodríguez, Flavio Fontana, Matthias Faessler, Christian Forster, Jürgen Schmidhuber, Gianni Di Caro, et al. A machine learning approach to visual perception of forest trails for mobile robots. *IEEE Robotics and Automation Letters*, 1(2):661–667, 2016.
82. X Glorot, A Bordes, and Y Bengio. Domain adaptation for large-scale sentiment classification: a deep learning approach. In *International Conference on Machine Learning*, pages 513–520, 2011.
83. B Gong, K Grauman, and F Sha. Reshaping visual datasets for domain adaptation. In *Advances in Neural Information Processing Systems*, pages 1286–1294, 2013.
84. Boqing Gong, Kristen Grauman, and Fei Sha. Connecting the dots with landmarks: discriminatively learning domain-invariant features for unsupervised domain adaptation. In *International Conference on Machine Learning*, pages 222–230, 2013.
85. Boqing Gong, Yuan Shi, Fei Sha, and Kristen Grauman. Geodesic flow kernel for unsupervised domain adaptation. In *Conference on Computer Vision and Pattern Recognition*, pages 2066–2073. IEEE, 2012.
86. Raghuraman Gopalan, Ruonan Li, and Rama Chellappa. Domain adaptation for object recognition: an unsupervised approach. In *International Conference on Computer Vision*, pages 999–1006. IEEE, 2011.
87. Raghuraman Gopalan, Ruonan Li, and Rama Chellappa. Unsupervised adaptation across domain shifts by generating intermediate data representations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(11):2288–2302, 2014.

88. Raghuraman Gopalan, Ruonan Li, Vishal M Patel, Rama Chellappa, et al. Domain adaptation for visual recognition. *Foundations and Trends ® in Computer Graphics and Vision*, 8(4):285–378, 2015.
89. A Gretton, AJ Smola, J Huang, M Schmittfull, KM Borgwardt, and B Schölkopf. Covariate shift by kernel mean matching. In Quiñonero Candela, M Sugiyama, A Schwaighofer, and ND Lawrence, editors, *Dataset Shift in Machine Learning*, pages 131–160. MIT Press, 2009.
90. Arthur Gretton, Karsten M Borgwardt, Malte Rasch, Bernhard Schölkopf, and Alex J Smola. A kernel method for the two-sample-problem. In *Advances in Neural Information Processing Systems*, pages 513–520, 2007.
91. Peter D Grünwald and A Philip Dawid. Game theory, maximum entropy, minimum discrepancy and robust Bayesian decision theory. *Annals of Statistics*, pages 1367–1433, 2004.
92. Quanquan Gu and Jie Zhou. Learning the shared subspace for multi-task clustering and transductive transfer classification. In *International Conference on Data Mining*, pages 159–168. IEEE, 2009.
93. Hirotaka Hachiya, Masashi Sugiyama, and Naonori Ueda. Importance-weighted least-squares probabilistic classifier for covariate shift adaptation with application to human activity recognition. *Neurocomputing*, 80:93–101, 2012.
94. Haibo He and Edwardo A Garcia. Learning from imbalanced data. *IEEE Transactions on Knowledge and Data Engineering*, 21(9):1263–1284, 2009.
95. Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: surpassing human-level performance on imagenet classification. In *International Conference on Computer Vision*, pages 1026–1034, 2015.
96. Ruining He and Julian McAuley. Ups and downs: modeling the visual evolution of fashion trends with one-class collaborative filtering. In *International Conference on World Wide Web*, pages 507–517. International World Wide Web Conferences Steering Committee, 2016.
97. James Heckman. Varieties of selection bias. *American Economic Review*, 80(2):313–318, 1990.
98. James J Heckman. Sample selection bias as a specification error (with an application to the estimation of labor supply functions), 1977.
99. Ernst Hellinger. Neue begründung der theorie quadratischer formen von unendlichvielen veränderlichen. *Journal für die Reine und Angewandte Mathematik*, 136:210–271, 1909.
100. Jon C Helton, Jay Dean Johnson, Cedric J Sallaberry, and Curt B Storlie. Survey of sampling-based methods for uncertainty and sensitivity analysis. *Reliability Engineering & System Safety*, 91(10):1175–1209, 2006.
101. Geoffrey E Hinton and Richard S Zemel. Autoencoders, minimum description length and helmholtz free energy. In *Advances in Neural Information Processing Systems*, pages 3–10, 1994.
102. Wassily Hoeffding. Probability inequalities for sums of bounded random variables. *Journal of the American Statistical Association*, 58(301):13–30, 1963.
103. J Hoffman, E Rodner, J Donahue, T Darrell, and K Saenko. Efficient learning of domain-invariant image representations. *International Conference on Learning Representations*, 2013.
104. Judy Hoffman, Trevor Darrell, and Kate Saenko. Continuous manifold based adaptation for evolving visual domains. In *Conference on Computer Vision and Pattern Recognition*, pages 867–874, 2014.

105. Judy Hoffman, Erik Rodner, Jeff Donahue, Brian Kulis, and Kate Saenko. Asymmetric and category invariant feature transformations for domain adaptation. *International Journal of Computer Vision*, 109(1-2):28–41, 2014.
106. Kurt Hornik. Approximation capabilities of multilayer feedforward networks. *Neural Networks*, 4(2):251–257, 1991.
107. Kurt Hornik, Maxwell Stinchcombe, and Halbert White. Multilayer feedforward networks are universal approximators. *Neural Networks*, 2(5):359–366, 1989.
108. Jiayuan Huang, Arthur Gretton, Karsten M Borgwardt, Bernhard Schölkopf, and Alex J Smola. Correcting sample selection bias by unlabeled data. In *Advances in Neural Information Processing Systems*, pages 601–608, 2007.
109. Guido W Imbens and Donald B Rubin. *Causal inference in statistics, social, and biomedical sciences*. Cambridge University Press, 2015.
110. Nathalie Japkowicz, Catherine Myers, Mark Gluck, et al. A novelty detection approach to classification. In *International Joint Conference on Artificial Intelligence*, volume 1, pages 518–523, 1995.
111. Jing Jiang. *Domain adaptation in natural language processing*. University of Illinois at Urbana-Champaign, 2008.
112. W Evan Johnson, Cheng Li, and Ariel Rabinovic. Adjusting batch effects in microarray expression data using empirical Bayes methods. *Biostatistics*, 8(1):118–127, 2007.
113. Michael I Jordan and Tom M Mitchell. Machine learning: trends, perspectives, and prospects. *Science*, 349(6245):255–260, 2015.
114. Konstantinos Kamnitsas, Christian Baumgartner, Christian Ledig, Virginia Newcombe, Joanna Simpson, Andrew Kane, David Menon, Aditya Nori, Antonio Criminisi, Daniel Rueckert, et al. Unsupervised domain adaptation in brain lesion segmentation with adversarial networks. In *International Conference on Information Processing in Medical Imaging*, pages 597–609. Springer, 2017.
115. Takafumi Kanamori, Shohei Hido, and Masashi Sugiyama. Efficient direct density ratio estimation for non-stationarity adaptation and outlier detection. In *Advances in Neural Information Processing Systems*, pages 809–816, 2009.
116. Takafumi Kanamori, Shohei Hido, and Masashi Sugiyama. A least-squares approach to direct importance estimation. *Journal of Machine Learning Research*, 10:1391–1445, 2009.
117. Andrej Karpathy and Li Fei-Fei. Deep visual-semantic alignments for generating image descriptions. In *Conference on Computer Vision and Pattern Recognition*, pages 3128–3137, 2015.
118. Tsuyoshi Kato, Kinya Okada, Hisashi Kashima, and Masashi Sugiyama. A transfer learning approach and selective integration of multiple types of assays for biological network inference. In *Computational Knowledge Discovery for Bioinformatics Research*, pages 188–202. IGI Global, 2012.
119. Michael J Kearns and Umesh Virkumar Vazirani. *An introduction to computational learning theory*. MIT Press, 1994.
120. Daniel Kifer, Shai Ben-David, and Johannes Gehrke. Detecting change in data streams. In *International Conference on Very Large Data Bases*, pages 180–191. VLDB Endowment, 2004.
121. Minyoung Kim and Vladimir Pavlovic. Central subspace dimensionality reduction using covariance operators. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(4):657–670, 2011.
122. Igor Kononenko. Machine learning for medical diagnosis: history, state of the art and perspective. *Artificial Intelligence in Medicine*, 23(1):89–109, 2001.

123. WM Kouw, M Loog, LW Bartels, and AM Mendrik. Mr acquisition-invariant representation learning. *ArXiv*, (2017), 2017.
124. Wouter M. Kouw and Marco Loog. Target contrastive pessimistic risk for robust domain adaptation. *ArXiv*, (2017), 2017.
125. Wouter M Kouw, Marco Loog, Lambertus Wilbert Bartels, and Adriënne M Mendrik. Learning an mr acquisition-invariant representation using siamese neural networks. 2019.
126. Wouter M Kouw, Laurens JP Van Der Maaten, Jesse H Krijthe, and Marco Loog. Feature-level domain adaptation. *Journal of Machine Learning Research*, 17(171):1–32, 2016.
127. Jan Kremer, Fabian Gieseke, K Steenstrup Pedersen, and Christian Igel. Nearest neighbor density ratio estimation for large-scale applications in astronomy. *Astronomy and Computing*, 12:67–72, 2015.
128. Miroslav Kubat, Robert C Holte, and Stan Matwin. Machine learning for the detection of oil spills in satellite radar images. *Machine Learning*, 30(2-3):195–215, 1998.
129. Brian Kulis, Kate Saenko, and Trevor Darrell. What you saw is not what you get: domain adaptation using asymmetric kernel transforms. In *Conference on Computer Vision and Pattern Recognition*, pages 1785–1792. IEEE, 2011.
130. Solomon Kullback and Richard A Leibler. On information and sufficiency. *Annals of Mathematical Statistics*, 22(1):79–86, 1951.
131. John Langford and John Shawe-Taylor. PAC-Bayes & margins. In *Advances in Neural Information Processing Systems*, pages 439–446, 2003.
132. Erik Learned-Miller. Hyperspacings and the estimation of information theoretic quantities. *University of Massachusetts-Amherst Technical Report: Amherst, MA, USA*, 2004.
133. Lung-Fei Lee. Some approaches to the correction of selectivity bias. *Review of Economic Studies*, 49(3):355–372, 1982.
134. Jeffrey T Leek, Robert B Scharpf, Héctor Corrada Bravo, David Simcha, Benjamin Langmead, W Evan Johnson, Donald Geman, Keith Baggerly, and Rafael A Irizarry. Tackling the widespread and critical impact of batch effects in high-throughput data. *Nature Reviews: Genetics*, 11(10), 2010.
135. CJ Leggetter and PC Woodland. Maximum likelihood linear regression for speaker adaptation of continuous density hidden markov models. *Computer Speech & Language*, 9(2):171–185, 1995.
136. W Li, L Duan, D Xu, and IW Tsang. Learning with augmented features for supervised and semi-supervised heterogeneous domain adaptation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(6):1134–1148, 2014.
137. Roderick JA Little and Donald B Rubin. *Statistical analysis with missing data*. John Wiley & Sons, 2014.
138. Anqi Liu and Brian Ziebart. Robust classification under sample selection bias. In *Advances in Neural Information Processing Systems*, pages 37–45, 2014.
139. Mingsheng Long, Jianmin Wang, Jiaguang Sun, and S Yu Philip. Domain invariant transfer kernel learning. *IEEE Transactions on Knowledge and Data Engineering*, 27(6):1519–1532, 2015.
140. Marco Loog. Nearest neighbor-based importance weighting. In *International Workshop on Machine Learning for Signal Processing*, pages 1–6. IEEE, 2012.
141. Marco Loog. Supervised classification: Quite a brief overview. In *Machine Learning Techniques for Space Weather*, pages 113–145. Elsevier, 2018.

142. Jie Lu, Dianshuang Wu, Mingsong Mao, Wei Wang, and Guangquan Zhang. Recommender system application developments: a survey. *Decision Support Systems*, 74:12–32, 2015.
143. David JC MacKay. *Information theory, inference and learning algorithms*. Cambridge University Press, 2003.
144. Yishay Mansour, Mehryar Mohri, and Afshin Rostamizadeh. Domain adaptation: Learning bounds and algorithms. In *Conference on Learning Theory*, 2009.
145. Yishay Mansour and Mariano Schain. Robust domain adaptation. *Annals of Mathematics and Artificial Intelligence*, 71(4):365–380, 2014.
146. Aleix M Martínez and Avinash C Kak. PCA versus LDA. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(2):228–233, 2001.
147. David McAllester. Simplified PAC-Bayesian margin bounds. *Lecture Notes in Computer Science*, pages 203–215, 2003.
148. Geoffrey McLachlan. *Discriminant analysis and statistical pattern recognition*, volume 544. John Wiley & Sons, 2004.
149. Suyu Mei, Wang Fei, and Shuigeng Zhou. Gene ontology based transfer learning for protein subcellular localization. *BMC Bioinformatics*, 12(1):1, 2011.
150. Erik G Miller. A new class of entropy estimators for multi-dimensional densities. In *International Conference on Acoustics, Speech, and Signal Processing*, volume 3, pages III–297. IEEE, 2003.
151. Mehryar Mohri, Afshin Rostamizadeh, and Ameet Talwalkar. *Foundations of Machine Learning*. MIT Press, 2012.
152. Krikamol Muandet, David Balduzzi, and Bernhard Schölkopf. Domain generalization via invariant feature representation. In *International Conference on Machine Learning*, pages 10–18, 2013.
153. Marcel Nassar, Rami Abdallah, Hady Ali Zeineddine, Elias Yaacoub, and Zaher Dawy. A new multitask learning method for multiorganism gene network estimation. In *International Symposium on Information Theory*, pages 2287–2291. IEEE, 2008.
154. M Taufiq Nuruzzaman, Changmoo Lee, and Deokjai Choi. Independent and personal SMS spam filtering. In *International Conference on Computer and Information Technology*, pages 429–435. IEEE, 2011.
155. Atsuyuki Okabe. *Spatial tessellations*. Wiley Online Library, 1992.
156. Sinno Jialin Pan, James T Kwok, and Qiang Yang. Transfer learning via dimensionality reduction. In *Association for the Advancement of Artificial Intelligence*, volume 8, pages 677–682, 2008.
157. Sinno Jialin Pan, Xiaochuan Ni, Jian-Tao Sun, Qiang Yang, and Zheng Chen. Cross-domain sentiment classification via spectral feature alignment. In *International Conference on World wide web*, pages 751–760. ACM, 2010.
158. Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, 22(10):1345–1359, 2010.
159. SJ Pan, IW Tsang, JT Kwok, and Q Yang. Domain adaptation via transfer component analysis. *Neural Networks*, 22(2):199–210, 2011.
160. Vishal M Patel, Raghuraman Gopalan, Ruonan Li, and Rama Chellappa. Visual domain adaptation: a survey of recent advances. *IEEE Signal Processing Magazine*, 32(3):53–69, 2015.
161. VMK Peddinti and P Chintalapoodi. Domain adaptation in sentiment analysis of twitter. In *AAAI Conference on Artificial Intelligence*, 2011.
162. Foster Provost and Tom Fawcett. Robust classification for imprecise environments. *Machine Learning*, 42(3):203–231, 2001.

163. Raúl Rojas. *Neural networks: a systematic introduction*. Springer, 2013.
164. DB Rubin. Inference and missing data. *Biometrika*, 63(3):581–592, 1976.
165. Stuart Jonathan Russell and Peter Norvig. Artificial intelligence: a modern approach, 2009.
166. Kate Saenko, Brian Kulis, Mario Fritz, and Trevor Darrell. Adapting visual category models to new domains. *European Conference on Computer Vision*, pages 213–226, 2010.
167. Marco Saerens, Patrice Latinne, and Christine Decaestecker. Adjusting the outputs of a classifier to new a priori probabilities: a simple procedure. *Neural Computation*, 14(1):21–41, 2002.
168. Winifred Scaf-Klomp. Screening for breast cancer. *Attendance and psychological consequences. Rijksuniversiteit Groningen, Groningen*, 142, 1997.
169. Jeffrey C Schlimmer and Richard H Granger. Incremental learning from noisy data. *Machine Learning*, 1(3):317–354, 1986.
170. Bernhard Schölkopf, Alexander Smola, and Klaus-Robert Müller. Kernel principal component analysis. In *International Conference on Artificial Neural Networks*, pages 583–588. Springer, 1997.
171. Bernhard Schölkopf, Alexander Smola, and Klaus-Robert Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, 10(5):1299–1319, 1998.
172. Bernhard Schölkopf and Alexander J Smola. *Learning with kernels: support vector machines, regularization, optimization, and beyond*. MIT Press, 2002.
173. M Shao, D Kit, and Y Fu. Generalized transfer subspace learning through low-rank constraint. *International Journal of Computer Vision*, 109(1-2):74–93, 2014.
174. John Shawe-Taylor and Nello Cristianini. *Kernel methods for pattern analysis*. Cambridge University Press, 2004.
175. Hidetoshi Shimodaira. Improving predictive inference under covariate shift by weighting the log-likelihood function. *Journal of Statistical Planning and Inference*, 90(2):227–244, 2000.
176. Si Si, Dacheng Tao, and Bo Geng. Bregman divergence-based regularization for transfer subspace learning. *IEEE Transactions on Knowledge and Data Engineering*, 22(7):929–942, 2010.
177. Grigori Sidorov, Francisco Velasquez, Efstathios Stamatatos, Alexander Gelbukh, and Liliana Chanona-Hernández. Syntactic n-grams as machine learning features for natural language processing. *Expert Systems with Applications*, 41(3):853–860, 2014.
178. Bernard W Silverman. *Density estimation for statistics and data analysis*, volume 26. CRC Press, 1986.
179. A Smola, A Gretton, L Song, and B Schölkopf. A hilbert space embedding for distributions. In *Algorithmic Learning Theory*, pages 13–31, 2007.
180. Masashi Sugiyama, Matthias Krauledat, and Klaus-Robert Müller. Covariate shift adaptation by importance weighted cross validation. *Journal of Machine Learning Research*, 8:985–1005, 2007.
181. Masashi Sugiyama and Klaus-Robert Müller. Input-dependent estimation of generalization error under covariate shift. *Statistics & Decisions*, 23(4/2005):249–279, 2005.
182. Masashi Sugiyama and Klaus-Robert Müller. Model selection under covariate shift. In *International Conference on Artificial Neural Networks*, pages 235–240. Springer, 2005.

183. Masashi Sugiyama, Shinichi Nakajima, Hisashi Kashima, Paul V Buenau, and Motoaki Kawanabe. Direct importance estimation with model selection and its application to covariate shift adaptation. In *Advances in Neural Information Processing Systems*, pages 1433–1440, 2008.
184. Masashi Sugiyama, Taiji Suzuki, and Takafumi Kanamori. Density ratio estimation in machine learning, 2012.
185. Baochen Sun and Kate Saenko. Subspace distribution alignment for unsupervised domain adaptation. In *British Machine Vision Conference*, pages 24–1, 2015.
186. Richard S Sutton and Andrew G Barto. *Reinforcement learning: an introduction*, volume 1. MIT Press, 1998.
187. Sergios Theodoridis, Aggelos Pikrakis, Konstantinos Koutroumbas, and Dionisis Cavouras. *Introduction to pattern recognition: a matlab approach*. Academic Press, 2010.
188. Andreĭ Nikolaevich Tikhonov. Solution of incorrectly formulated problems and the regularization method. In *Soviet Mathematics Doklady*, volume 5, pages 1035–1038, 1963.
189. Antonio Torralba and Alexei A Efros. Unbiased look at dataset bias. In *Conference on Computer Vision and Pattern Recognition*, pages 1521–1528. IEEE, 2011.
190. Yuta Tsuboi, Hisashi Kashima, Shohei Hido, Steffen Bickel, and Masashi Sugiyama. Direct density ratio estimation for large-scale covariate shift adaptation. *Journal of Information Processing*, 17:138–155, 2009.
191. Pavan Turaga, Ashok Veeraraghavan, and Rama Chellappa. Statistical analysis on stiefel and grassmann manifolds with applications in computer vision. In *Conference on Computer Vision and Pattern Recognition*, pages 1–8. IEEE, 2008.
192. Eric Tzeng, Judy Hoffman, Trevor Darrell, and Kate Saenko. Simultaneous deep transfer across domains and tasks. In *International Conference on Computer Vision*, pages 4068–4076, 2015.
193. Leslie G Valiant. A theory of the learnable. *Communications of the ACM*, 27(11):1134–1142, 1984.
194. Tim Van Erven and Peter Harremos. Rényi divergence and kullback-leibler divergence. *IEEE Transactions on Information Theory*, 60(7):3797–3820, 2014.
195. Tim Van Kasteren, Gwenn Englebienne, and Ben JA Kröse. Transferring knowledge of activity recognition across sensor networks. In *Pervasive Computing*, volume 10, pages 283–300. Springer, 2010.
196. A van Opbroek, MA Ikram, MW Vernooij, and M De Bruijne. Transfer learning improves supervised image segmentation across imaging protocols. *IEEE Transactions on Medical Imaging*, 34(5):1018–1030, 2015.
197. VN Vapnik. *The nature of statistical learning theory*. Springer, 2000.
198. VN Vapnik and V Vapnik. *Statistical learning theory*, volume 2. Wiley New York, 1998.
199. Mathukumalli Vidyasagar. *A theory of learning and generalization*. Springer-Verlag New York, Inc., 2002.
200. Pascal Vincent, Hugo Larochelle, Isabelle Lajoie, Yoshua Bengio, and Pierre-Antoine Manzagol. Stacked denoising autoencoders: learning useful representations in a deep network with a local denoising criterion. *Journal of Machine Learning Research*, 11(Dec):3371–3408, 2010.
201. Jianqiang Wang, Lei Zhang, Dezhao Zhang, and Keqiang Li. An adaptive longitudinal driving assistance system based on driver characteristics. *IEEE Transactions on Intelligent Transportation Systems*, 14(1):1–12, 2013.

202. Larry Wasserman. *All of statistics: a concise course in statistical inference.* Springer, 2013.
203. Junfeng Wen, Chun-Nam Yu, and Russell Greiner. Robust learning under uncertain test distributions: relating covariate shift to model misspecification. In *International Conference on Machine Learning*, pages 631–639, 2014.
204. Peter Whittle. Multi-armed bandits and the gittins index. *Journal of the Royal Statistical Society. Series B*, pages 143–149, 1980.
205. Christian Widmer, Nora C Toussaint, Yasemin Altun, Oliver Kohlbacher, and Gunnar Rätsch. Novel machine learning methods for MHC Class I binding prediction. In *International Conference on Pattern Recognition in Bioinformatics*, pages 98–109. Springer, 2010.
206. Gerhard Widmer and Miroslav Kubat. Learning in the presence of concept drift and hidden contexts. *Machine Learning*, 23(1):69–101, 1996.
207. Marc Wieland and Massimiliano Pittore. Performance evaluation of machine learning algorithms for urban pattern recognition from multi-spectral satellite images. *Remote Sensing*, 6(4):2912–2939, 2014.
208. Yung-Chow Wong. Differential geometry of grassmann manifolds. *Proceedings of the National Academy of Sciences*, 57(3):589–594, 1967.
209. Huan Xu and Shie Mannor. Robustness and generalization. *Machine Learning*, 86(3):391–423, 2012.
210. Qian Xu, Hong Xue, and Qiang Yang. Multi-platform gene-expression mining and marker gene analysis. *International Journal of Data Mining and Bioinformatics*, 5(5):485–503, 2011.
211. Qian Xu and Qiang Yang. A survey of transfer and multitask learning in bioinformatics. *Journal of Computing Science and Engineering*, 5(3):257–268, 2011.
212. Ting Yao, Yingwei Pan, Chong-Wah Ngo, Houqiang Li, and Tao Mei. Semi-supervised domain adaptation with subspace learning for visual recognition. In *Conference on Computer Vision and Pattern Recognition*, pages 2142–2150, 2015.
213. Yao-Liang Yu and Csaba Szepesvári. Analysis of kernel mean matching under covariate shift. In *International Conference on Machine Learning*, pages 1147–1154, 2012.
214. G Zen, E Sangineto, E Ricci, and N Sebe. Unsupervised domain adaptation for personalized facial emotion recognition. In *International Conference on Multimodal Interaction*, pages 128–135, 2014.
215. C Zhang, L Zhang, and J Ye. Generalization bounds for domain adaptation. In *Advances in Neural Information Processing Systems*, pages 3320–3328, 2012.
216. Jiajun Zhang and Chengqing Zong. Deep neural networks in machine translation: an overview. *IEEE Intelligent Systems*, 30(5):16–25, 2015.
217. Kun Zhang, Mingming Gong, and Bernhard Schölkopf. Multi-source domain adaptation: a causal view. In *Association for the Advancement of Artificial Intelligence*, pages 3150–3157, 2015.
218. Jingjing Zheng, Ming-Yu Liu, Rama Chellappa, and P Jonathon Phillips. A grassmann manifold-based domain adaptation approach. In *International Conference on Pattern Recognition*, pages 2095–2099. IEEE, 2012.