A Primer on Domain Adaptation

Seri Lee

January 11, 2021

Standard supervised machine learning assumes that the distribution of the source samples used to train an algorithm is the same as teh one of the target samples on which it is supposed to make predictions. However, this is hardly ever the case in practice. The myriad of methods available and the unfortunate lack of a clear and universally accepted terminology can make the topic of domain adaptation daunting for the newcomer. This review aims at a coherent presentation of four important special cases: (1) prior shift, (2) covariate shift, (3) concept shift, and (4) subspace mapping.

Introduction

Classical theory of ML assumes that the new observations from the test set are drawn from the same population as those from the training set. This however is an ideal situation rarely met in practice. The training set is then said to be biased with respect to the test set. Depending on the situation, this bias can either be known or unknown. Domain adaptation (DA) is a collection of methods that aims at compensating for the statistical asymmetry between the train set and the test set. There are several reasons that make it difficult for a newcomer to build a coherent overview of domain adaptation:

- Research papers on DA use a specialized vocabulary and concepts that will not apply elsewhere.
- There is no universal terminology agreement for referring to different types of domain adaptation.
- DA is sometimes confused with transfer learning (TL). While TL deals with transferring knowledge gained on one task to use it on another, DA deals with one single task for which the training and test observations have different statistical properties.

In this review, my aim is to try to fill in the above gaps. My commitment is to describe 4 important practical cases of DA.

The four special cases of DA I consider are described below. Henceforth, the source domain will refer to the population from which traning observations are drawn while the target domain will refer to the population from which test observations are drawn.

1. **Prior shift** refers to a situation in which the label distributions are different in the source and target domains. The class conditional

distributions of the features given the label are however assumed to be identical in both domains.

2. **Covariate shift** refers to a situation in which the distributions of the features are different but known in the source and target domains. The conditional dependence of the labels on features are however assumed to be the same in both domains.

3. **Concept shift** refers to the case where the dependence of the label upon the features differs between the target and the source domains, often depending on time. The distribution of the features are nevertheless assumed to be the same in both domains.

4. **Subspace mapping** describes a situation where observations are distributed alike as physical objects in the source and target domains but where the features are different and related by an unknown change of coordinates.

I examine these 4 instances of DA in turn, using the two main theoretical frameworks of Machine Learning (ML), namely the PAC learning framework of statistical learning and the maximum likelihood principle. The PAC theory formulates the aim of ML as making good predictions directly from samples of a probability distribution from which nothing is assumed. The maximum likelihood principle on the other hand assumes that samples are generated from some parametrized probability distribution whose parameters are to be optimized to make the observed samples as likely as possible.

Statistical Learning Recap

PAC learning framework

The classical mathematical framework for defining statistical learning is called *Probably Approximately Correct* (PAC) learning.

We assume that the observations are defined as pairs (x, y) of features x in some feature space $\mathcal{X} \subset \mathbb{R}^p$ and of labels y in a label space $\mathcal{Y}$ which could be either a part of $\mathbb{R}$ for a regression or a set of labels for a classification. The relationship between x and y is described by an unknown joint probability distribution $p(x, y)$.

Suppose that information is given to us only in the form of a sample $S = \{(x_1, y_1), \dots, (x_m, y_m)\}$ of observations drawn from this unknown distribution $p(x, y)$. Our aims is to use this information S to find a "good" approximation of the dependence of y on x as function $h : \mathcal{X} \to \mathcal{Y}$.

We select his function from a collection of $\mathcal{H}$, the hypothesis class. To assess the quality of a predictor h we assume moreover that we are given a loss function $l : \dagger \times \mathcal{Y} \to \mathbb{R}$ which measures the discrepancy $l(y, \hat{y})$ between an observed value y and a prediction $\hat{y} = h(x)$. Common choices are the l_{0-1} loss which counts the number of points where $y \neq y'$ for a binary classifier and $l_{LS} := |y - y'|^2$ for least square regression.

The true risk $R[h]$ associated with a predictor $h \in \mathcal{H}$ is then defined as the expectation of this loss l when averaged over the unknown distribution p:

$$R[h] := \mathbb{E}_{(x,y)\sim p}[l(y, h(x))] \tag{1}$$

This is the quantity that we want to minimize over predictors $h \in \mathcal{H}$ using some machine learning algorithm. We are thus led to the following definition:

A hypothesis class $\mathcal{H}$ is *learnable* if the following is true: there exist an algorithm A which, for any given precision $\epsilon > 0$ and $\delta > 0$, takes a sample S of size m, whose observations are sampled from p, and returns a predictor $h_S = A(S) \in \mathcal{H}$ such that

$$R[h_S] \leq \min_{h \in \mathcal{H}} R[h] + \epsilon \tag{2}$$

with a confidence $1 - \delta$ provided $m := |S|$ is large enough.

In other words, the algorithm A should manage to pick a predictor whose risk is ϵ-close to the optimal h that can be achieved within the class $\mathcal{H}$. If the actual relationship beween x and $y = h(x)$ is deterministic and if moreover this h belongs to $\mathcal{H}$, then the right hand side of (2) simply reduces to ϵ.

In intuitive terms: a class $\mathcal{H}$ of functions is termed *learnable* when there exists an algorithm A that takes a sample S as input to select a predictor $h_S = A(S)$ from $\mathcal{H}$ which has high chances ("Probably") to have low risk $R[h]$ ("Approximately Correct"), provided the sample size $m := |S|$ is large enough.