6.7900 Fall 2024: Lecture Notes 15

1 Unsupervised Domain Adaptation

1.1 What is domain shift?

Domain shift refers to anytime the the training data is not exactly like the test data. Some examples include:

- Temporal change in financial market
- Demographic changes across different hospital
- Different accents for speech recognition

We will refer to the accessible data as the *source* domain, and the data on which we would like to perform well as the *target* domain.

In general, we may have some labeled or unlabeled examples from the target domain. In this lecture, we will consider the case when we only have unlabeled examples from the target domain (unsupervised domain adaptation). Because we do not have any labeled examples in the target domain, we must assume that the label set is the same between source and target (for instance, if the source domain labels are animal classes like cat, dog, mouse, etc., the target domain labels must also be animal classes).

1.2 Taxonomy of domain shift

We will denote the data we wish to label as x and the desired labels as y. Denote the distribution over x as p_x and the distribution over y given x as $p_{y|x}$. We will also denote the marginal distribution over y as p_y. Finally, we will denote the source distributions with a superscript ^{S} and target distributions with a superscript ^{T}. We categorize the settings we will consider as:

- Label shift only: p_y changes, $p_{x|y}$ is the same
- Co-variate shift only: p_x changes, $p_{y|x}$ is the same
- Co-variate transformation only: p_x and $p_{y|x}$ changes such there is a transformation g satisfying $p_x^T(g(x)) = p_x^S(x)$, $p_{y|x}^T(y|g(x)) = p_{y|x}^S(y|x)$

1.3 Handling domain shift

In practice, how can we account for domain shift? We will assume we have access to sufficient data from the source distribution such that we can closely model or approximate $p_x^S, p_{y|x}^S$. Our goal is to estimate $p_{y|x}^T$.

1.3.1 Label-shift only

In the case of label shift only, assuming we know p_y^T:

$$p_{y|x}^T(y|x) = \frac{p_{y|x}^S(y|x)\frac{p_y^T(y)}{p_y^S(y)}}{\sum_{y'} p_{y|x}^S(y'|x)\frac{p_y^T(y')}{p_y^S(y')}} \tag{1}$$

In other words, we may simply re-weight the predictions on the target points by the ratio of marginal probabilities of y between the source and target ($\frac{p_y^T(y)}{p_y^S(y)}$). In order to practically compute this, we must know the class ratios between the source and target.

Exercise: How can we derive this equation?

1.3.2 Co-variate shift only

We will consider two methods to handle domain shift in the case of co-variate shift: importance sampling and domain alignment.

Importance sampling In the importance sampling approach, we first assume access to a model with parameters θ and loss function L that we would like to minimize in expectation over the target samples:

$$\sum_{x,y} p_x^T(x) p_{y|x}^T(y|x) L(x,y,\theta) \tag{2}$$

If we are able to approximate this loss, we can simply minimize over θ to find a good model on the target domain. However, we do not necessarily know p_x^T or $p_{y|x}^T$, and must find ways to approximate this in terms of p_x^S and $p_{y|x}^S$.

Using the fact that $p_{y|x}^T = p_{y|x}^S$, we have:

$$\sum_{x,y} p_x^T(x) p_{y|x}^T(y|x) L(x,y,\theta) = \sum_x \left(\frac{p_x^T(x)}{p_x^S(x)}\right) \sum_y p_x^S(x) p_{y|x}^S(y|x) L(x,y,\theta) \tag{3}$$

Note that we have simply reweighted the expected loss of each point x under the source domain by the ratio of the marginal probabilities of x between the source and target ($\frac{p_x^T(x)}{p_x^S(x)}$). Intuitively, this works because we are emphasizing points that have high probability under the target domain.

How can we practically estimate this ratio for unseen points x? One approach is to train a classifier that predicts whether a point is drawn from the source distribution or target distribution given that the point is actually drawn from an equal mixture of the two distributions ($\frac{1}{2}p_x^S + \frac{1}{2}p_x^T$). Specifically, we assume a trained classifier Q outputs:

$$Q(T|x) = \frac{p_x^T(x)}{p_x^T(x) + p_x^S(x)} \tag{4}$$

$$Q(S|x) = \frac{p_x^S(x)}{p_x^T(x) + p_x^S(x)} \tag{5}$$

Exercise: (How) do these expressions change if we don't draw from an equal mixture between the source and target?

Then, the ratio of the classifier outputs equals the ratio of marginal probabilities over x:

$$\frac{p_x^T(x)}{p_x^S(x)} = \frac{Q(T|x)}{Q(S|x)} \tag{6}$$

We may finally substitute this into the earlier expression to find:

$$\sum_{x,y} p_x^T(x) p_{y|x}^T(y|x) L(x, y, \theta) = \sum_x \left(\frac{Q(T|x)}{Q(S|x)}\right) \sum_y p_x^S(x) p_{y|x}^S(y|x) L(x, y, \theta) \tag{7}$$

To summarize this approach, we first train a classifier on the mixture $\frac{1}{2}p_x^S + \frac{1}{2}p_x^T$ to distinguish source domain and target domain points. We then compute the expected loss on the source domain, but reweight points x by the likelihood ratio of being in the target domain vs. the source domain. Finally, we minimize θ over this reweighted loss.

Adversarial domain alignment Another approach to deal with co-variate shift is to find a representation of the data that is *shared* between the source and the target. Specifically, we would like some mapping $z = \phi_w(x)$ (with parameters w) such that the distribution of $\phi_w(x)$ when $x \sim p^S$ is the same as the distribution of $\phi_w(x)$ when $x \sim p^T$. In other words, we would like:

$$p_z^T = p_z^S \tag{8}$$

Once we have such a shared representation, we can expect a model trained to predict y from z on the source distribution to also perform well on the target distribution because the distributions over z are shared and $p^S_{y|z} = p^T_{y|z}$ by assumption.

Exercise: Why does $p^S_{y|x} = p^T_{y|x}$ imply $p^S_{y|z} = p^T_{y|z}$?

To do this, we will again use a classifier Q_β (parametrized by β) that tries to separate the source and target examples given a mixture of the two. Now, instead of taking x as input, the classifier will take input z. We will train the mapping ϕ_w to *maximize* the loss of the classifier Q_β: the classifier will try to separate data from the two domains while w will try to make the two domains have the same distribution over z.

Note that one simple way to make $p^S_z = p^T_z$ is to set $z = \phi_w(x) = 0$. However, this z is not at all informative about y and will not prove useful to model the distribution on y in the target (or source) domain. Thus, we also want to set ϕ_w such that y can be adequately predicted from z.

Now, we will synthesize these two goals into a single objective function. We will assume we have a source data model h_θ parameterized by θ that outputs a predicted y given a z. We also assume access to a loss $L(\hat{y}, y)$ where $\hat{y}$ is the predicted y and y is the true y. We would like to solve:

$$\min_{\theta,w} \sum_{x,y} p^S_x(x) p^S_{y|x}(y|x) L(h_\theta(\phi_w(x)), y) + \lambda \max_\beta [\sum_x p^T_x(x) \log Q_\beta(T|\phi_w(x)) + \sum_x p^S_x(x) \log Q_\beta(S|\phi_w(x))] \quad (9)$$

The first term corresponds to maintaining good performance on the source data. The second term corresponds to minimizing the distance between the source and target distributions in the space of $\phi_W(x)$. Note that it simply the log-likelihood of the data being under the source of target domain under the classifier Q_β: thus, intuitively, the classifier would try to maximize it while w would try to minimize it.

1.3.3 Co-variate transformation only

In the case of co-variate transformation only, we will consider an *optimal transport* approach to try to approximate the mapping g between original source points and target points. Optimal transport is a method that, given two sets of points (or distributions), finds a *correspondence* between the points in one and the points in the other such that the distance between each pair of corresponded points is minimized.

The optimal transport distance $W(p^S_x, p^T_x)$ between the source and target distributions p_x^S and p_x^T is defined as:

$$W(p_x^S, p_x^T) = \min_{p_{x,x'}} \sqrt{\sum_{x,x'} p_{x,x'}(x,x') d(x,x')^2} \tag{10}$$

where d is a distance function and the joint distribution $p_{x,x'}$ is constrained to satisfy:

$$\sum_x p_{x,x'}(x,x') = p_x^T(x') \tag{11}$$

$$\sum_{x'} p_{x,x'}(x,x') = p_x^S(x) \tag{12}$$

$$p_{x,x'}(x,x') \geq 0 \tag{13}$$

The resulting minimizing $p_{x,x'}$ defines a correspondence between the points in the source and target domains; given a point x' in the target domain, the distribution over corresponding points x in the source domain is:

$$p_{x|x'}(x|x') = \frac{p_{x,x'}(x,x')}{\sum_{x''} p_{x,x'}(x'',x')} \tag{14}$$

For any given target domain point, once we know the corresponding source domain points, we may simply apply our source domain model on the corresponding source domain points:

$$p_{y|x}^T(y|x') = \sum_x p_{y|x}^S(y|x) p_{x|x'}(x|x') \tag{15}$$

Exercise: What assumption do we need to make about g for this to be valid?