Unsupervised Domain Adaptation by Transduction

Abstract

Abstract.

1. Introduction

Intro

2. Related Work

Related work

3. Method

3.1. Problem Definition

We are interested in the problem in which we have an unsupervised domain $\{\mathbf{x_i}\}_{i\in[N^u]}$ and a supervised domain $\{\hat{\mathbf{x}}_\mathbf{i}, \hat{y}_i\}_{i\in[N^s]}$ such that $\mathbf{x_i}$ is the extracted feature for point i and y_i is the corresponding label. We also assume that unsupervised and supervised features have the same dimension i.e. $\mathbf{x}, \hat{\mathbf{x}} \in \mathcal{R}^d$

We consider an asymmetric similarity metric;

$$s(\mathbf{x_i}, \hat{\mathbf{x}}_\mathbf{j}) = \mathbf{x_i}^\intercal \mathbf{W} \hat{\mathbf{x}}_\mathbf{j} \tag{1}$$

such that it is high if two points from supervised and unsupervised domains are similar to each other.

We approach to the problem from a transduction perspective; in other words, the main purpose of the method is recovering the labels y_i for each unsupervised example $\mathbf{x_i}$. We consider the following objective function in order to compute y_i as well as the similarity metric $\mathbf{W}$.

$$\begin{aligned} \min_{\mathbf{W}, y_i} \sum_{i\in[N^s]} \quad & [s(\hat{\mathbf{x}}_\mathbf{i}, \mathbf{x_{i^-}}) - s(\hat{\mathbf{x}}_\mathbf{i}, \mathbf{x_{i^+}}) + \alpha]_+ \\ s.t. \quad & i^+ = \arg\max_{j|y_j=\hat{y}_i} s(\hat{\mathbf{x}}_\mathbf{i}, \mathbf{x_j}) \\ & i^- = \arg\max_{j|y_j\neq\hat{y}_i} s(\hat{\mathbf{x}}_\mathbf{i}, \mathbf{x_j}) \end{aligned} \tag{2}$$

We solve this optimization problem via alternating minimization through iterating over solving for unsupervised labels y_i and the similarity metric $\mathbf{W}$. We explain these two steps the following sections.

Preliminary work. Under review by the International Conference on Machine Learning (ICML). Do not distribute.

3.2. Labeling Unsupervised Points

We are using nearest neighbor rule in our labelling with an additional robustness metric. We first explain the nearest neighbor formulation then explain our extension.

Given a similarity metric $\mathbf{W}$, the labeling with nearest neighbor rule is;

$$(y_i)^{pred} = \hat{y}_{\arg\max_j s(\mathbf{x_i}, \hat{\mathbf{x}}_\mathbf{j})} \tag{3}$$

Although the nearest neighbor rule is computationally inefficient, we solve the efficiency issues through usage of batches with stochastic gradient descent as well as an efficient implementation though OpenBLAS routines. We discuss these details in the implementation section of the paper.

Although the nearest neighbor rule is expected to work with a good metric, at the initial stage of the algorithm the metric will not be accurate enough. Hence, we need a robustness measure to handle the initial stage of the algorithm. Our robustness measure comes from the consistency of labels over the unsupervised data graph.

We create a k-NN graph over the unsupervised data points such that $\mathcal{N}(\mathbf{x_i})$ is the k unsupervised data point nearest to $\mathbf{x_i}$ via the l_2 distance. After the k-NN graph is created, we solve the following optimization problem for labeling unsupervised data points;

$$\begin{aligned} \arg\min_{y_i} \sum_{i\in N^u} & - \max_{y_j=y_i} s(\hat{\mathbf{x}}_\mathbf{j}, \mathbf{x_i}) \\ & + \lambda \sum_{i\in N^u} \sum_{j\in\mathcal{N}(\mathbf{x_i})} \mathbf{x_i}^T \mathbf{x_j} \mathbb{1}(y_i \neq y_j) \end{aligned} \tag{4}$$

This problem is sub-modular and can easily be optimized through many methods like α-β swapping, quadratic pseudo-boolean optimization (QPBO), linear programming through roof-duality etc. We use α-β swapping algorithm from (**?**).

3.3. Learning Similarity Metric

Given the predicted labels y_i for unsupervised data points $\mathbf{x_i}$, we learn the asymmetric metric following the LMNN(Large Margin Nearest Neighbour)(**?**) construction. We start with finding the nearest positive and negative ex-

amples through;

$$\begin{aligned} i^+ &= \arg\max_{j|y_j=\hat{y}_i} s(\hat{\mathbf{x}}_\mathbf{i}, \mathbf{x}_\mathbf{j}) \\ i^- &= \arg\max_{j|y_j\neq\hat{y}_i} s(\hat{\mathbf{x}}_\mathbf{i}, \mathbf{x}_\mathbf{j}) \end{aligned} \tag{5}$$

Then, we construct the loss function with regularizer;

$$\min_{\mathbf{W}, y_i} \sum_{i\in[N^s]} [s(\hat{\mathbf{x}}_\mathbf{i}, \mathbf{x}_{\mathbf{i}^-}) - s(\hat{\mathbf{x}}_\mathbf{i}, \mathbf{x}_{\mathbf{i}^+}) + \alpha]_+ + r(\mathbf{W}) \tag{6}$$

which is convex in terms of the similarity metric $\mathbf{W}$ if the regularizer is convex; and we optimize it by using Stochastic gradient descent through the subgradient $\frac{\partial loss(y_i, \mathbf{W})}{\partial \mathbf{W}} =$

$$\sum_{i\in[N^s]} \mathbb{1}[s(\hat{\mathbf{x}}_\mathbf{i}, \mathbf{x}_{\mathbf{i}^-}) - s(\hat{\mathbf{x}}_\mathbf{i}, \mathbf{x}_{\mathbf{i}^+}) > \alpha] \left(\hat{\mathbf{x}}_\mathbf{i}\mathbf{x}_{\mathbf{i}^-}{}^\intercal - \hat{\mathbf{x}}_\mathbf{i}\mathbf{x}_{\mathbf{i}^+}{}^\intercal\right) + \frac{\partial r(\mathbf{W})}{\partial \mathbf{W}} \tag{7}$$

As a regularizer we are using the Frobenius norm of the similarity matrix as $r(\mathbf{W}) = \frac{1}{2}\|\mathbf{W}\|_F^2$

3.4. Learning Features

3.5. Efficient Concerns

ADD THE FINAL ALGO HERE IN ALGORITHMIC FORMAT

4. Convergence Analysis

It is monotonically decreasing after the each iteration

5. Experimental Results

6. Conclusion

Acknowledgements

a