Robust Transduction for Unsupervised Adaptation

Abstract

Abstract.

1. Introduction

Intro

2. Related Work

Related work

3. Method

3.1. Problem Definition

We are trying to learn the relationships between two domains of data. In our setting, one of the domains is fully supervised and has both data points as well as class labels $\{\hat{\mathbf{x}}_i, \hat{y}_i\}_{i \in [N^s]}$ such that \mathbf{x}_i is the data point i and y_i is the corresponding label. Whereas, the other domain is unsupervised and only have data points as $\{\mathbf{x}_i\}_{i \in [N^u]}$. We further assume that there is a feature function $\Phi(\cdot)$ which is used in both of the domains.

We formulate our problem as a metric learning problem and consider an an asymmetric similarity metric;

$$s(\mathbf{x_i}, \mathbf{\hat{x}_i}) = \Phi(\mathbf{x_i})^\mathsf{T} \mathbf{W} \Phi(\mathbf{\hat{x}_i}) \tag{1}$$

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

We model our learning setting in a fully transductive setting; 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} .

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

$$s.t. \qquad i^+ = \arg\max_{j|y_j = \hat{y}_i} s(\hat{\mathbf{x}}_i, \mathbf{x}_j)$$

$$i^- = \arg\max_{j|y_j \neq \hat{y}_i} s(\hat{\mathbf{x}}_i, \mathbf{x}_j)$$

$$(2)$$

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

3.2. Labeling Unsupervised Points

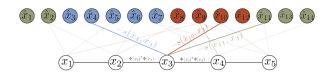


Figure 1. Visualization of the Label Propogation. Consider the unsupervised point x_3 , the resulting label would be: $\arg\min_{y_3} - s(\hat{\mathbf{x}}_4,\mathbf{x}_3)\mathbb{1}(\mathbf{y}_3=1) - s(\hat{\mathbf{x}}_{10},\mathbf{x}_3)\mathbb{1}(y_3=2) - s(\hat{\mathbf{x}}_{11},\mathbf{x}_3)\mathbb{1}(y_3=0) + \Phi(\mathbf{x}_2)^{\mathsf{T}}\phi(\mathbf{x}_3)\mathbb{1}(y_3\neq y_2) + \Phi(\mathbf{x}_3)^{\mathsf{T}}\phi(\mathbf{x}_4)\mathbb{1}(y_3\neq y_4)$ assuming green is 0, blue is 1 and red is 2

In order to label the unsupervised data-points, we use nearest-neighbor(NN) rule as computing the NN supervised datapoint using the learned metric $s(\cdot,\cdot)$. Moreover, the NN rule will be accurate only if the metric is accurate enough. Since our algorithm is iterative alternating optimization, we still need rather accurate labeling even in the case of sub-optimal similarity metric. Hence, we introduce an additional robustness measure through label propagation. We first explain our NN-rule, then extend it to label propagation.

Given a similarity metric $s(\cdot,\cdot)$, the nearest neighbor rule is:

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

The major issue for using the nearest neighbor rule is computationally efficiency. However, our implementation method and choice of parameters makes the NN rule tractable. We explain how we use stochastic gradient descent, select the batch size and efficiently implement the NN rule using OpenBLAS in Section??.

In order to make our transduction stage robust, we use label propagation. Main idea behind our label propagation approach is enforcing consistency of the labels over unsupervised data points. In order to enforce this consistency, we create a k-NN graph over the unsupervised data points such that neighbor set $\mathcal{N}(\mathbf{x_i})$ for \mathbf{x}_i is the k-unsupervised data

point nearest to $\mathbf{x_i}$ using the l_2 distance in feature space. After the k-NN graph is created, we solve the following optimization problem for labeling unsupervised data points;

 $\arg \min_{y_i} \sum_{i \in N^u} - \max_{\hat{y}_j = y_i} s(\hat{\mathbf{x}}_j, \mathbf{x}_i) + \lambda \sum_{i \in N^u} \sum_{j \in \mathcal{N}(\mathbf{x}_i)} \mathbf{x}_i^T \mathbf{x}_j \mathbb{1}(y_i \neq y_j)$ (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 (?). In order to further explain the label propagation, we visualize a sample example with k=2 and 3-class classification problem in Figure 1. Since it is rather out-of-scope of this paper, we explain the details of the α - β swapping algorithm to the appendix.

3.3. Learning Similarity Metric

Given the predicted labels y_i for unsupervised data points $\mathbf{x_i}$, we need to learn the asymmetric metric in order to optimize the loss function defined in (2). We extend the LMNN(Large Margin Nearest Neighbour)(?) construction to the multi-domain case in order to define our metric learning objective.

Main intuition behind our formulation is searching for a metric which will label the supervised data points correctly using the unsupervised data points and their predicted labels. Since at this stage we already have a predicted label for each unsupervised data points, we can estimate a label for the supervised data points using these predicted labels. We also have ground truth labels for the supervised data points and we can look for a metric which will maximize the accuracy. In other words, the over all metric learning is combination of;

- ullet predicting \hat{y}_{j}^{pred} using $\mathbf{x_i}, \mathbf{\hat{x}_j}, \hat{y}_{j}$
- learning $s(\cdot, \cdot)$ by penalizing $(\hat{y}_i)^{pred} \neq \hat{y}_i$

Fortunately, this can be jointly solved by minimizing the triplet loss as we define through nearest same-class and different-class examples of each supervised datapoint from unsupervised data points. In other words, we find the nearest positive and negative examples through;

$$i^{+} = \arg \max_{j|y_{j} = \hat{y}_{i}} s(\hat{\mathbf{x}_{i}}, \mathbf{x_{j}})$$

$$i^{-} = \arg \max_{j|y_{j} \neq \hat{y}_{i}} s(\hat{\mathbf{x}_{i}}, \mathbf{x_{j}})$$
(5)

Then, we construct the triplet loss function with regularizer as;

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

Algorithm 1 Robust Transduction with Metric Learning

Input: unsupervised \mathbf{x}_i , supervised $\hat{\mathbf{x}}_i$, y_i , batch size B repeat

Sample $(\mathbf{x}_1^b, \dots \mathbf{x}_B^b)$, $(\hat{\mathbf{x}}_1^b, \dots \hat{\mathbf{x}}_B^b)$, $(\hat{y}_1^b, \dots \hat{x}_B^b)$ Solve (4) using $(\mathbf{x}_1^b, \dots \mathbf{x}_B^b)$ and $(\hat{\mathbf{x}}_1, \dots \hat{\mathbf{x}}_{N^s}^b)$ for i = 1 to B do

if \hat{y}_i in $y_1 \dots y_B$ then

Compute (i^+, i^-) via $(\mathbf{x}_1^b, \dots \mathbf{x}_B^b)$, $(\hat{\mathbf{x}}_1^b, \dots \hat{\mathbf{x}}_B^b)$ Update $\frac{\partial loss(y_i, \mathbf{W})}{\partial \mathbf{\Theta}}$ and $\frac{\partial loss(y_i, \mathbf{W})}{\partial \mathbf{W}}$ using (7,8)end if

end for $\mathbf{W} \leftarrow \mathbf{W} + \alpha \frac{\partial loss(y_i, \mathbf{W})}{\partial \mathbf{W}}$ $\mathbf{\Theta} \leftarrow \mathbf{\Theta} + \alpha \frac{\partial loss(y_i, \mathbf{W})}{\partial \mathbf{\Theta}}$ until CONVERGENCE or $MAX \ JTER$

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

$$\sum_{i \in [N^s]} \mathbb{1}(s(\hat{\mathbf{x_i}}, \mathbf{x_{i^-}}) - s(\hat{\mathbf{x_i}}, \mathbf{x_{i^+}}) > \alpha) (\hat{\mathbf{x_i}} \mathbf{x_{i^-}}^\intercal - \hat{\mathbf{x_i}} \mathbf{x_{i^+}}^\intercal) + \frac{\partial r(\mathbf{W}) 85}{\partial \mathbf{W}^{186}}$$

As a regularizer we are using the Frobenius norm of the similarity matrix as $r(\mathbf{W}) = \frac{1}{2} ||\mathbf{W}||_F^2$. We explain the details of this optimization routine and how we implement in the Section ??.

3.4. Learning Features

In Section 3.2 and 3.3, we developed our transductive labeling method with propogation and the way to learn the metric $s(\cdot,\cdot)$ through learning \mathbf{W} . During this formulation, we used a pre-defined feature function Φ . However, the current trends in machine learning suggests that learning this feature function Φ from the data directly is a promising direction especially for visual data points. Hence, we consider the case Φ_{Θ} is a parametrized feature function with parameter set Θ . A typical example is CNNs(convolutional neural networks) with Θ as concatenation of weights and biases. We update the feature weights as part of the metric learning. Hence, the gradient update for Θ becomes; $\frac{\partial loss(y_i, \mathbf{W})}{\partial \mathbf{W}} =$

$$\sum_{i \in [N^s]} \mathbb{1}(s(\hat{\mathbf{x_i}}, \mathbf{x_{i^-}}) - s(\hat{\mathbf{x_i}}, \mathbf{x_{i^+}})) \left(\frac{\partial s(\hat{\mathbf{x_i}}, \mathbf{x_{i^-}})}{\partial \mathbf{\Theta}} - \frac{\partial s(\hat{\mathbf{x_i}}, \mathbf{x_{i^+}}))}{\partial \mathbf{\Theta}}\right) \frac{209}{210}$$
(8)

3.5. Implementation Details

3.6. Weakly-Supervised Case

4. Convergence Analysis

It is monotonically decreasing after the each iteration

224

225

226

230

231

235

236

242

247

248

253

259

260

261

262

263

264

265

266

267

268

269

270

271

272

273

274

319

275

276

278

279

281

282

283

284

285

286

287

289

290

291

293

295

296

297

299

300

301

302

303

304

305

306

307

308

309

312

313

314

324

329

Figure 2. Sample images from MNIST, MNIST-M and SVHN datasets.

Figure 3. Sample images from Amazon, Webcam and D-SLR domains of the Office dataset.

5. Experimental Results

We evaluate our algorithm on various unsupervised domain adaptation tasks. We perform two sets of experiments one on MNIST and its house-made artificial variant. As well as the standard domain adaptation dataset Office(?). Main motivation behind performing two different sets of experiments is the effect of dataset size; although it is standard, Office dataset is quite small having maximum number of 2478 data points per domain over 31 object classes. We discuss the datasets and their characteristics as well as the baselines in detail in the following sections.

5.1. Datasets

In order to evaluate our algorithm, we are using two sets of experiments one on digit classification and the other one on object recognition.

For digit classification, we are using three different domains as; MNIST(?): MNIST is a digit classification dataset of 60k images. MNIST-M: As a controlled target domain, we generated a series of digit images by using the original MNIST dataset and the color images of BSDS500(?) following the method explained in (?). Since the dataset is not distributed directly by the authors, we further confirmed that the performance is similar to experimented in (?). SVHN(?): Street view house numbers dataset is a collection of house numbers collected directly from Google street view images. It is quite interesting because of dataset size as well as the difficulty of the task. There are 600k images in the SVHN(?) dataset. Moreover, the adaptation task of MNIST \rightarrow SVHN is proven to be extremely hard and we show the first successful adaptation results for it.

For object recognition experiment, we are using the OF-FICE(?) dataset and it has three domains as Amazon: Images of objects with white background directly taken from Amazon. Webcam: Images of the same objects taken with a webcam on a complicated background. **D-SLR:** Images of the same objects taken with a high resultion D-SLR.

5.2. Baselines

5.3. Implementation Details

5.3.1. FEATURE EXTRACTION - CNN ARCHITECTURES

In all of our experiments, we are using 128 dimensional features extracted using convolutional neural networks. For each experiment, we are using a different architecture mostly because of the dataset size.

For digit classification experiments, we are using 5-layer networks of; We learn all parameters directly from the data



Figure 4. Feature Extraction Network - Modified AlexNet

using AdaGrad(?) and initialize with truncated normals having unit variance. We use the learning rate $2.5x10^{-3}$.

Moreover; for object recognition experiment, we are using the AlexNet architecture(?); Due to the scarcity of the



Figure 5. Feature Extraction Network - Modified AlexNet

data, we use the parameters pre-trained on ImageNet for all layers except last fully connected layer. We further extend the network with fully connected layer of dimensionality reduction to 128D. We learn the fc-7 and fc-8 using Ada-Grad(?) and initialize with truncated normals having unit variance. We use the learning rate $2.5x10^{-4}$.

We further share our learned models as well as the source code using TensorFlow(?) on anonymous.xyz

5.4. Evaluation Procedure

6. Conclusion

Acknowledgements

References

Chopra, Sumit, Balakrishnan, Suhrid, and Gopalan, Raghuraman. Dlid: Deep learning for domain adaptation by interpolating between domains. In ICML workshop on challenges in representation learning, volume 2, pp. 5, 2013.

Fernando, Basura, Habrard, Amaury, Sebban, Marc, and Tuytelaars, Tinne. Unsupervised visual domain adaptation using subspace alignment. In Computer Vision

Robust Transduction for Unsupervised Adaptation

Source Target	Amazon Webcam	D-SLR Webcam	Webcam D-SLR	Webcam Amazon	Amazon D-SLR	D-SLR Amazon	MNIST-M MNIST	MNIST MNIST-M
GFK (Gong et al., 2013)	.214	.691	.650					
SA* (Fernando et al., 2013)	.450	.648	.699					.567
DLID (Chopra et al., 2013)	.519	.782	.899					
DDC (Tzeng et al., 2014)	.605	.948	.985					
DAN (Long & Wang, 2015)	.645	.952	.986					
Backprop (Ganin & Lempitsky, 2015)	.730	.964	.992					.771
Source Only	.642	.961	.978					.524
Our Method	.781	.930	.987	.625	.839	.567	.835	.855

(ICCV), 2013 IEEE International Conference on, pp. 2960–2967. IEEE, 2013.

Ganin, Yaroslav and Lempitsky, Victor. Unsupervised domain adaptation by backpropagation. In Blei, David and Bach, Francis (eds.), *Proceedings of the 32nd International Conference on Machine Learning (ICML-15)*, pp. 1180–1189. JMLR Workshop and Conference Proceedings, 2015. URL http://jmlr.org/proceedings/papers/v37/ganin15.pdf.

Gong, Boqing, Grauman, Kristen, and Sha, Fei. Connecting the dots with landmarks: Discriminatively learning domain-invariant features for unsupervised domain adaptation. In *Proceedings of The 30th International Conference on Machine Learning*, pp. 222–230, 2013.

Long, Mingsheng and Wang, Jianmin. Learning transferable features with deep adaptation networks. *arXiv* preprint arXiv:1502.02791, 2015.

Tzeng, Eric, Hoffman, Judy, Zhang, Ning, Saenko, Kate, and Darrell, Trevor. Deep domain confusion: Maximizing for domain invariance. *arXiv preprint arXiv:1412.3474*, 2014.