ESANN 2015 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 22-24 April 2015, i6doc.com publ., ISBN 978-287587014-8.
Available from http://www.i6doc.com/en/.

Optimal transport for semi-supervised domain adaptation

Denis Rousselle[1,2] and Stéphane Canu[1]

1- Normandie Université, INSA de Rouen - LITIS EA 4108
Avenue de l'Université, BP 8, 76801 Saint-Etienne-du-Rouvray- FRANCE
2- Thales Underwater Systems
Route de sainte Anne du Portzic, CS 43814, 29238 Brest cedex 3- FRANCE

Abstract. Domain adaption for semi-supervised learning is still a challenging task. Indeed, available solutions are often slow and fail to provide relevant interpretations. Here we propose a new algorithm to solve this problem of semi-supervised domain adaptation efficiently, by using an adapted combination of transportation algorithms. Our empirical evidence supports our initial intuition, showing the interest of the proposed method.

1 Motivation

Statistical learning usually makes the assumption that training and testing data are drawn from the same distribution. Dataset shift occurs when this assumption is violated [for a detailed presentation see 1]. In that case, the problem amounts to finding a transformation that transports data and learned structures from a training domain, toward a test domain. This is the problem of domain adaptation (DA). Our work focuses on a particular case of domain adaptation for supervised learning, when a subset of the target labels is known. This problem is referred as semi-supervised domain adaptation.

An interesting solution to semi-supervised domain adaptation has been introduced by Gong *et al.* [2] providing state-of-the-art results. Their solution is a kernel-based method that takes advantage of directly exploiting data low-dimensional structures. This approach is also interesting because it facilitates comparisons. Indeed, it focuses only on data domain adaptation, regardless the classifier used to perform the supervised classification task. However, in their approach, dimension reduction used is computationally expensive and leads to the loss of data interpretability. Inspired by [3], we propose to address this domain adaptation issue by using an efficient and easy to parallelize optimal transport algorithm, adapted for semi-supervised domain adaptation. Furthermore, our method can be physically interpreted since links between source and target data are explicit.

Before presenting our method, some notations have to be introduced. Observed input in the source domain is X^s of size n_s observations $\times\, d$ variables associated with the probability distribution μ_s while $Y^s \in L^{n_s}$ denotes the associated labels, L being the discrete set of all possible labels. Analogously, (X^t, Y^t) denote data in the target domain of size n_t associated with the probability distribution μ_t.

ESANN 2015 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 22-24 April 2015, i6doc.com publ., ISBN 978-287587014-8.
Available from http://www.i6doc.com/en/.

In semi supervised learning, the target domain is assumed to be split into two subsets depending on whether or not target label is known. The known part is (X^k, Y^k) while (X^u, Y^u) denotes the unknown (with Y^u unobserved), so that the training set is $(X^a, Y^a) = (X^s \cup X^k, Y^s \cup Y^k)$ of size n_a. In [3], *unsupervised* domain adaptation is seen as the optimal transport of the underlying distribution of sources μ^s at X^s towards target μ^t distribution at X^t. The associated (Monge-Kantorovitch) optimal transport problem can be written as:

$$P^\star = \underset{P \in U(\mu_s, \mu_t)}{\arg\min} \ \langle P, M \rangle_F, \tag{1}$$

with $U(\mu_s, \mu_t) := \left\{ P \in \mathbb{R}^{n_s \times n_t} \mid P\mathbf{1} = \mu^s;\ P^T\mathbf{1} = \mu^t \right\}$ the set of possible transportation between μ_s and μ_t, $\mathbf{1}$ the all-ones vector, $\langle \bullet, \bullet \rangle_F$ being the Frobenius inner product and M a cost matrix encoding the distance between samples of source and target domains (typically $M_{ij} = \|X^s_{i,\bullet} - X^t_{j,\bullet}\|_2^2$). This article propose to extend this idea to the *semi-supervised* domain adaptation problem.

2 Algorithm

2.1 Background

In [4], Cuturi proposes to solve the optimal transportation between two histograms (μ_s, μ_t) adding an entropic penalization, that is, for a given $\lambda > 0$:

$$P^\lambda = \underset{P \in U(\mu_s, \mu_t)}{\arg\min} \ \langle P, M \rangle_F \ - \ \lambda \, h(P), \tag{2}$$

with $h(P) = -\sum_{i=1}^{n_s} \sum_{j=1}^{n_t} P_{ij} log(P_{ij})$ the element wise entropy function. This formulation has two advantages: its solution is smoother than the one of (1) and it can be solved in a fast and easy to parallelize way (with the Sinkhorn algorithm see [4] for details). Since, this formulation does not take into consideration the labels of the source domain, [3] introduces a new term penalizing the mixing of source labels by promoting group sparcity:

$$P^{\lambda,\nu} = \underset{P \in U(\mu_s, \mu_t)}{\arg\min} \ \langle P, M \rangle_F \ - \ \tfrac{1}{\lambda} \, h(P) + \nu \sum_{j=1}^{n_t} \sum_{c \in L} \| \ P(I_c, j) \ \|_q^p, \tag{3}$$

where I_c is the index of the source points of label $c \in L$, and (λ, ν) a couple of given positive hyper-parameters.

2.2 Semi-supervised domain adaptation

Algo 1. A first idea to solve the semi-supervised domain adaptation problem is to apply (3) to transfer the underlying distribution of X^s onto the one of X^t, using the source labels Y^s. We will refer to this method as *algo 1.* But this global transfer doesn't make use of Y^k, the known labels in the target domain.

ESANN 2015 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 22-24 April 2015, i6doc.com publ., ISBN 978-287587014-8.
Available from http://www.i6doc.com/en/.

Algo 2. A way to take advantage of our knowledge of label is to, instead of transferring the whole distribution, transfer all the conditional distributions with repeat to the classes of X^s onto X^k, class by class. This can be done by applying (2) as many times as there are classes. We call this method *algo 2*.

Algo 3. However, especially when the number of classes is important and the number of points in each class is small, this can lead to wrong solutions. In that case we assume that it is easier to estimate the overall transfer of X^s only onto X^k than the one of the conditional distribution. Yet the information contained in the labels should be used. So we propose to start with transferring globally X^s only onto X^k using (2), and then to use the information provided by the labels in a post treatment. The idea is to force the transfer class by class by removing parts of the transfer matrix P (solution of 2) associated with two different classes and by distributing related information. As we know that optimal transport only takes into account the proportionality between cost coefficients, we allocate the sum of all P_{ij} that should be equal to zero, proportionally to the magnitude of the relevant one. Given matrix P, the post treatment provides a matrix P^0 such that:

$$P^0(i,j) = \begin{cases} 0 & \text{if } Y_i^s \neq Y_j^k \\ s_j P(i,j) & \text{else} \end{cases}$$

where s_j is set such that $\sum_{j=1}^{n_s} P^0(i,j) = \sum_{j=1}^{n_s} P(i,j)$. This method is *algo 3*.

Algo 2b and 3b. To get better estimates of the underlying density distribution, we had the idea to adapt also the source data X^s to X^u the target where the labels are unknown. We propose to do so by computing the optimal transport of X^k the target data with known labels towards X^u the target where the labels are unknown, and apply this transport to the projection of X^s onto X^k. This amounts to solving two independent transportation problems:

- first, transport from the source to the target where the labels are known using P^0 (note this can be done by either *algo 2* or *3*),
- second, transport from the target where the labels are known to the target where the labels are unknown with P^1.

On the second part, since X^k labels are known, we can use the algorithm proposed in (3). Errors due to this transportation should be insignificant (the two domains are in fact the same). Only the class balance may eventually change. Projection of source to the target domain can be written as a weighted barycenter of data X^u:

$$X^{s\to t} = diag((P^0\mathbf{1})^{-1})P^0 diag((P^1\mathbf{1})^{-1})P^1 X^u.$$

This two step method is referred as respectively *algo 2b* and *3b* when resp. *algo 2* or *3* is used in the first phase.

ESANN 2015 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 22-24 April 2015, i6doc.com publ., ISBN 978-287587014-8.
Available from http://www.i6doc.com/en/.

2.3 Implementation considerations

The use of these methods require the knowledge of the distance matrix M and histograms μ_s and μ_t. The euclidean distance has been used for M. Regarding the histograms, as μ_s represent the distribution of source points, one can simply use $\mu_s(i) = \frac{1}{n_s}$. But the Parzen-Rozenblatt estimator is recommended by [3]:

$$\mu_s(i) = \frac{\sum_{j=1}^{n_s} k_\sigma(X^s_{i,\bullet}, X^s_{j,\bullet})}{\sum_{i=1}^{n_s}\sum_{j=1}^{n_s} k_\sigma(X^s_{i,\bullet}, X^s_{j,\bullet})},$$

where $k_\sigma(\bullet,\bullet)$ is the gaussian kernel. Albeit, it is said that σ have little incidence on the result, so we set σ to 1.

3 Experiment

3.1 Protocol and dataset

To test our method, we use the CaltechOffice dataset for semi-supervised learning[1]. This image dataset has been designed for benchmarking domain adaptation algorithm [2]. This dataset is composed of four domains (Amazon, Caltech, Webcam and DSLR), each divided into 10 classes. Images are represented by normalize surf [5] histograms of $d = 800$ bins, domains are subsequently independently zscored (zero mean and unitary variance). To facilitate comparisons, we follow the same protocol as in [2]:

1. the source domain has $n_s = 20$ examples except for DSLR where $n_s = 8$;
2. the target domain is divided into 2 parts:
   - label data with $n_k = 3$ labeled examples per target category ;
   - the remaining data is unlabeled. It is the *real* testing set ;
3. first, source domain is adapted to target domain by one of the proposed algorithm and then decision is made by a 1-Nearest Neighbor classifier ;
4. each experiment is repeated 20 times.

We also set $\lambda = \frac{1}{100}$ and $\nu = 1$ for all experiment.

3.2 Results

All domain adaptation set-ups possible were done that is $4 \times 3 = 12$ DA tasks. For instance, A→C denotes the problem of adapting Amazon data to solve the Caltech problem. In figure 1 are plotted the mean accuracy of different methods on all these adaptation tasks whose numerical values are reported table 1. Table 1 also compare our results to state-of-the-art GFK method [2]. Figure 1 reveals

[1] Available at `http://www-scf.usc.edu/~boqinggo` and `https://github.com/jhoffman/MaxMarginDomainTransforms/tree/master/DataSplits-OfficeCaltech`

ESANN 2015 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 22-24 April 2015, i6doc.com publ., ISBN 978-287587014-8.
Available from http://www.i6doc.com/en/.

that the average gain due to basic DA is around 5% (*algo 1*, green line), while the gain due to semi supervised DA is about 10% (*algo 2*, black line) as well as the gain of the DA post treatment which is also of the order of 10% (*algo 2b*, red line)[2]. Our best result (*algo 2b*) improves the state of the art of the order of 5%. We remark also that our approach introduce a significant increase of standard deviation.

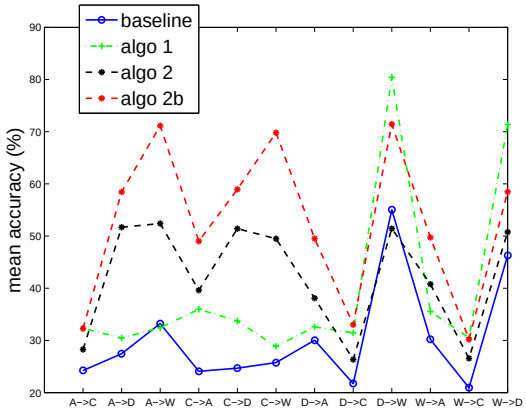

Fig. 1: Mean accuracy for 12 domains adaptations

	baseline	*algo 1*	*algo 2b*	GFK(S,A)
A → C	24.2±0.3	32.3±1.2	32.3±4.1	**39.6**±0.4
A → D	27.4±0.5	30.5±3.0	**58.4**±4.1	50.9±0.9
A → W	33.2±0.6	32.4±3.5	**71.2**±5.4	56.9±1.0
C → A	24.1±0.4	36.0±2.6	**49.0**±3.7	46.1±0.6
C → D	24.7±0.7	33.7±3.8	**58.9**±4.5	55.0±0.9
C → W	25.8±0.6	28.8±5.1	**69.8**±5.0	57.0±0.9
D → A	30.0±0.7	32.6 ±1.2	**49.5**±3.4	46.2±0.6
D → C	21.7±0.7	31.4±1.2	**33.0**±3.0	**33.9**±0.6
D → W	55.0±0.7	**80.4**±2.3	71.4±3.3	**80.2**±0.4
W → A	30.2±0.6	35.5±1.7	**49.7**±3.2	46.2±0.7
W → C	20.9±0.5	30.5±2.5	30.2 ±4.4	**32.3**±0.6
W → D	46.3±0.7	71.4±3.0	58.5±5.4	**74.1**±0.9
Median	26.6	32.5	**54.1**	48.6

Table 1: Mean accuracies for a baseline (kNN with no adaptation) and different DA algorithms including state of the art (GFK) and the best method proposed here (*algo 2b*). Best results are in bold.

[2]We choose not to report results for *algo 3* and *algo 3b* because they are very close to those of *algo 2* and *algo 2b*.

ESANN 2015 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 22-24 April 2015, i6doc.com publ., ISBN 978-287587014-8. Available from http://www.i6doc.com/en/.

4 Speed up

The method providing the best results and therefore the one we recommend, *algo 2b*, is quite slow and take 74 s for running all the experiments versus 28 s with *algo 1*[3]. This time complexity is due to our second stage. In [3], $\sum_{j=1}^{n_t} \sum_{c \in L} \| P(I_c, j) \|_{q=1}^{p=0.5}$ is solved by a sub-gradient since $(\bullet)^{0.5}$ is non-differentiable. We can smooth this with $p = 2$, leading to a convex and differentiable regularization term. The drawback is a group sparsity is less held in account, but our intuition is that this is not so important from the moment that histograms are drawn from the same domain. As expected, the running time reduces significantly (29 s) at the price of a very small decreasing of the mean accuracy.

5 Conclusion

In this paper was presented a new method for semi-supervised domain adaptation using a combination of slightly modified optimal transport algorithms. It performed competitive results on domain adaptation datasets. Nevertheless, high variance is observed on our experiments, thus possible improvements are the explanation and reduction of this variance. Our study focuses on the transport of source domain to target domain, it will be then interesting to study the other way, this may allow to learn once a complex classifier and then adapt testing data.

References

[1] Joaquin Quionero-Candela, Masashi Sugiyama, Anton Schwaighofer, and Neil D Lawrence. *Dataset shift in machine learning*. 2009.

[2] Boqing Gong, Yuan Shi, Fei Sha, and Kristen Grauman. Geodesic flow kernel for unsupervised domain adaptation. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 2066–2073. IEEE, 2012.

[3] N. Courty, R. Flamary, and D. Tuia. Domain adaptation with regularized optimal transport. In *Proceedings of ECML/PKDD 2014*, LNCS, pages 1–16, Nancy, France, September 2014.

[4] Marco Cuturi. Sinkhorn distances: Lightspeed computation of optimal transport. In *Advances in Neural Information Processing Systems*, pages 2292–2300, 2013.

[5] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. Surf: Speeded up robust features. In *Computer Vision–ECCV 2006*, pages 404–417. Springer, 2006.

[3] On a one core 2.5 GHz cadenced processor.