Joint distribution optimal transportation for domain adaptation

Nicolas Courty, Rémi Flamary, Amaury Habrard, Alain Rakotomamonjy
Neurips 2017

Presented By: Enyi Jiang, Carl Norbert, Akul Goyal

Presentation Overview

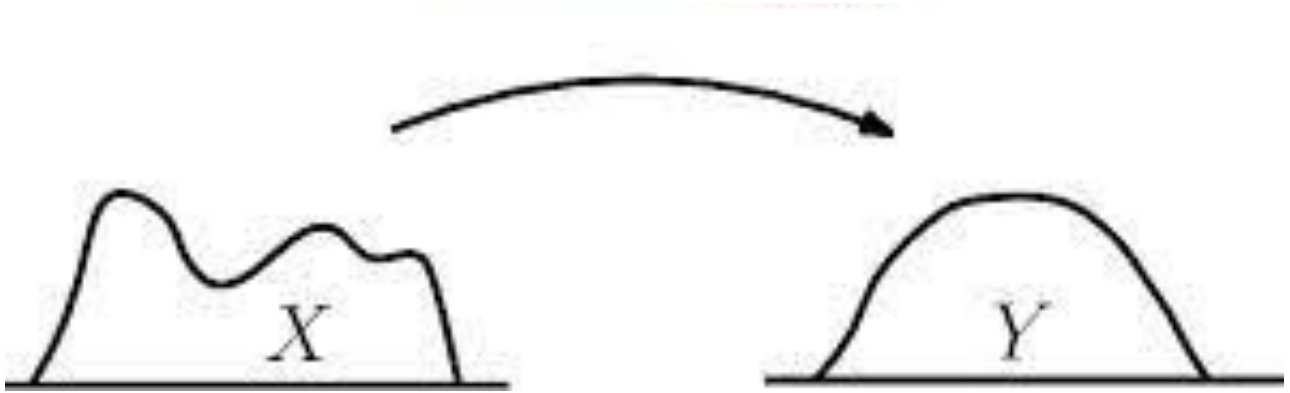
- Background
- Related Works
- Methodology
- Theoretical Analysis
- Results
- Conclusion

Presentation Overview

- Background
- Related Works
- Methodology
- Theoretical Analysis
- Results
- Conclusion

Background - Optimal Transport Distance (Wasserstein Distance)

- Measure Distance Between Two Distributions
- Common Form: Earth Mover's Distance



Background - Optimal Transport Distance (Wasserstein Distance)

- Measure Distance Between Two Distributions
- Solve this optimization problem:

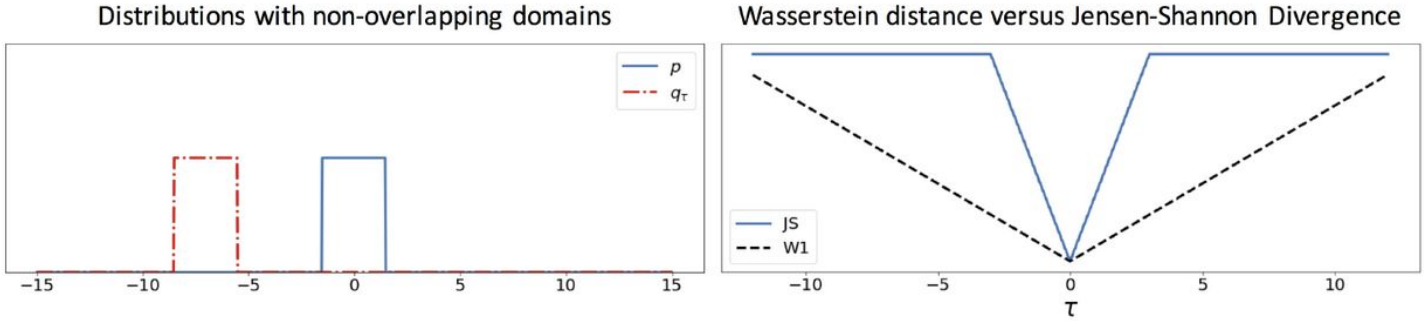
$$W_p(\mu,\nu) := \left(\inf_{\gamma \in \Gamma(\mu,\nu)} \int_{M\times M} d(x,y)^p \, \mathrm{d}\gamma(x,y) \right)^{1/p}$$

$\Gamma(\mu,\nu)$ =

	X_{i1}	X_{i1}	...	X_{in}	Total
X_{j1}					$P(X_{j1})$
X_{j2}					...
...					...
X_{jm}					$P(X_{jm})$
Total	$P(X_{i1})$	...	...	$P(X_{in})$	

Background - Optimal Transport Distance (Wasserstein Distance)

- Metric for Distance Between Two Distributions
- Different from JS-Divergence - Ability to handle non-overlapping distributions



P = uniform distribution
q(x) = (p(x-τ))

Presentation Overview

- Background
- Related Works
- Methodology
- Theoretical Analysis
- Results
- Conclusion

Related Works

Invariant Components **$P_s(T(X)) = P_t(T(X))$**: looking for a **transformation *T*** makes new representations of input data are **matching**.

- Class of transformation:
  - Projections [1]
  - Affine transform [2]
  - Non-linear transformation - neural networks [3]
- Types of divergences to compare these two
  - Kullback Leibler / Maximum Mean Discrepancy [4, 5]
  - Optimal Transport - Wasserstein Distance [6]

Presentation Overview

- Background
- Related Works
- Methodology
- Theoretical Analysis
- Results
- Conclusion

Methodology

Using a fixed *f* we solve for Γ:

Solve: $W_p(\mu,\nu) := \left(\inf_{\gamma \in \Gamma(\mu,\nu)} \int_{M\times M} d(x,y)^p \, \mathrm{d}\gamma(x,y) \right)^{1/p}$

$\mathcal{D}(\mathbf{x}_1, y_1; \mathbf{x}_2, y_2) = \alpha \mathrm{d}(\mathbf{x}_1, \mathbf{x}_2) + \mathcal{L}(y_1, y_2)$

After finding γ, we now find *f* using the following loss:

$$\min_{f \in \mathcal{H}} \sum_{i,j} \gamma_{i,j} \mathcal{L}(y_i^s, f(\mathbf{x}_j^t)) + \lambda \Omega(f)$$

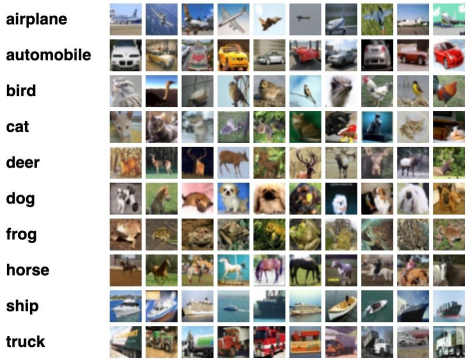
Algorithmically Solving for Γ, *f*

Setting: Source Dataset $P_s = (X_s, Y_s)$, Target Dataset $P_t = (X_t)$

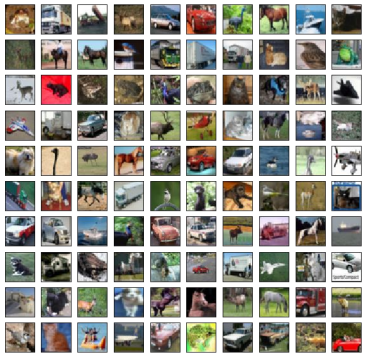
Algorithm:

1. Train Classifier (f) on P_S
2. Pseudo-Label P_t st. $P_t = (X_t, f(X_t))$
3. Calculate $W_p(P_s, P_t)$
4. Use Coupling Matrix (Upper-Case Gamma) to create soft label for $(x_t \in X_t)$
5. Retrain (f) on target data with soft labels
6. Repeat steps 2-5

Algorithmically Solving for Γ, *f*



Source Dataset



$y_{11}...y_{1m}$
$y_{21}...y_{2m}$
.
.
.

Target Dataset

Target Dataset

Source Dataset

	X_{i1}	X_{i1}	...	X_{in}	Total
X_{j1}					$P(X_{j1})$
X_{j2}					...
...					...
X_{jm}					$P(X_{jm})$
Total	$P(X_{i1})$	...	...	$P(X_{in})$	

Pseudo
y_{labels}
y_{1m}
y_{m}

train

Classifier *f*

train

$y_{i1} = (z_{j1} * y_{j1}) + (z_{j2} * y_{j2}) + ... + (z_{jm} * y_{jm})$

$\tilde{y}_{i1} = \{p(c_1), ..., p(c_k)\}$

$x_{i1}, \tilde{y}_{i1}$
...
$x_{in}, \tilde{y}_{in}$

X_{i1}
z_{j1}
z_{j2}
...
z_{jm}
$P(X_{i1})$

Presentation Overview

- Background
- Related Works
- Methodology
- Theoretical Analysis
- Results
- Conclusion

Theoretical Analysis

Probabilistic Transfer Lipschitzness (PTL):

- ***f***: labeling function, ***Π***: coupling, $\boldsymbol{\mu_s}$, $\boldsymbol{\mu_t}$***:*** source and target distributions
- ***φ(λ)***: φ-Lipschitz transferable

$$Pr_{(\mathbf{x}_1,\mathbf{x}_2)\sim\Pi(\mu_s,\mu_t)}\left[|f(\mathbf{x}_1)-f(\mathbf{x}_2)|>\lambda d(\mathbf{x}_1,\mathbf{x}_2)\right]\leq\phi(\lambda).$$

- given a deterministic labeling function and coupling, it bounds the probability of finding pairs of source-target instances labelled differently in a (1/λ)-ball with respect to Π

Bounds on Target Error

Theorem (Ben-David et al.' 07):

$$\varepsilon_T(h) \leq \varepsilon_S(h) + d_{\mathcal{H}}(S,T) + \lambda^*$$

- $\varepsilon_T(h)/\varepsilon_S(h)$: true target/source errors
- $d_{\mathcal{H}}(S,T)$: $\mathcal{H}$-div
- $\lambda^* := \min_{h' \in \mathcal{H}} \varepsilon_S(h') + \varepsilon_T(h')$: optimal joint error

Theorem: *f** is an optimal labeling function which verifies PTL

$$err_T(f) \leq W_1(\hat{\mathcal{P}}_s, \hat{\mathcal{P}}_t^f) + \sqrt{\frac{2}{c'}\log(\frac{2}{\delta})}\left(\frac{1}{\sqrt{N_S}} + \frac{1}{\sqrt{N_T}}\right) + err_S(f^*) + err_T(f^*) + kM\phi(\lambda).$$

Wasserstein Distance

Estimation Error

Joint error

$$|f^*(\mathbf{x}_1) - f^*(\mathbf{x}_2)| \leq M \text{ for all } \mathbf{x}_1, \mathbf{x}_2$$

$$|err_{T^f}(f^*) - err_S(f^*)| \leq W_1(\mathcal{P}_s, \mathcal{P}_t^f) + k * M * \phi(\lambda)$$

M: Essentially maximum possible distance in labels under constraints

φ(λ) : probability under which the probabilistic Lipschitzness does not hold

Presentation Overview

- Background
- Related Works
- Methodology
- Theoretical Analysis
- Results
- Conclusion

Office-Caltech Dataset - SVM + Linear Kernel

Figure 4: Some example images from Office-Caltech dataset. This dataset consists of four visual domains, namely images collected from Amazon merchant website, images collected from a high resolution DSLR camera, images collected from a web camera and images collected from Caltech-101 dataset.

Experimental Results - Classification

Table 1: Accuracy on the Caltech-Office Dataset. Best value in bold.

Domains	**Base**	**SurK**	**SA**	**ARTL**	**OT-IT**	**OT-MM**	**JDOT**
caltech→amazon	92.07	91.65	90.50	92.17	89.98	**92.59**	91.54
caltech→webcam	76.27	77.97	81.02	80.00	80.34	78.98	**88.81**
caltech→dslr	84.08	82.80	85.99	88.54	78.34	76.43	**89.81**
amazon→caltech	84.77	84.95	85.13	85.04	85.93	**87.36**	85.22
amazon→webcam	79.32	81.36	**85.42**	79.32	74.24	85.08	84.75
amazon→dslr	86.62	87.26	**89.17**	85.99	77.71	79.62	87.90
webcam→caltech	71.77	71.86	75.78	72.75	**84.06**	82.99	82.64
webcam→amazon	79.44	78.18	81.42	79.85	89.56	90.50	**90.71**
webcam→dslr	96.18	95.54	94.90	**100.00**	99.36	99.36	98.09
dslr→caltech	77.03	76.94	81.75	78.45	**85.57**	83.35	84.33
dslr→amazon	83.19	82.15	83.19	83.82	**90.50**	**90.50**	88.10
dslr→webcam	96.27	92.88	88.47	**98.98**	96.61	96.61	96.61
Mean	83.92	83.63	85.23	85.41	86.02	86.95	**89.04**
Mean rank	5.33	5.58	4.00	3.75	3.50	2.83	2.50
p-value	< 0.01	< 0.01	0.01	0.04	0.25	0.86	–

JDOT consistently outperforms the baseline (2 points in average)

Amazon Review Dataset - Neural Network

- Reviews are encoded as bag of words unigram and bigrams.
- Goal: Predict review sentiment as a binary class (>3 stars or <= 3 stars)
- Four product types are considered:
  - Books, DVDs, electronics, kitchens
  - 12 possible adaptation tasks
- Per domain:
  - 2,000 labelled samples
  - 4,000 unlabelled samples
  - Unlabelled samples used for transfer, labelled used for testing

Experimental Results - Classification

Table 2: Accuracy on the Amazon review experiment. Maximum value in bold font.

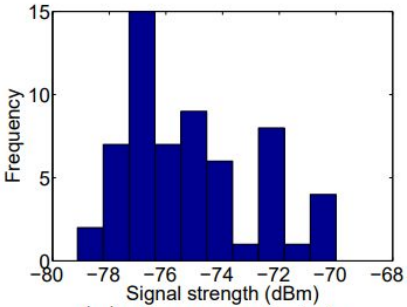
Domains	NN	DANN	JDOT (mse)	JDOT (Hinge)
books→dvd	0.805	**0.806**	0.794	0.795
books→kitchen	0.768	0.767	0.791	**0.794**
books→electronics	0.746	0.747	0.778	**0.781**
dvd→books	0.725	0.747	0.761	**0.763**
dvd→kitchen	0.760	0.765	0.811	**0.821**
dvd→electronics	0.732	0.738	0.778	**0.788**
kitchen→books	0.704	0.718	**0.732**	0.728
kitchen→dvd	0.723	0.730	0.764	**0.765**
kitchen→electronics	**0.847**	0.846	0.844	0.845
electronics→books	0.713	0.718	0.740	**0.749**
electronics→dvd	0.726	0.726	**0.738**	0.737
electronics→kitchen	0.855	0.850	0.868	**0.872**
Mean	0.759	0.763	0.783	**0.787**
p-value	0.004	0.006	0.025	–

JDOT surpasses DANN in 11 out of 12 tasks

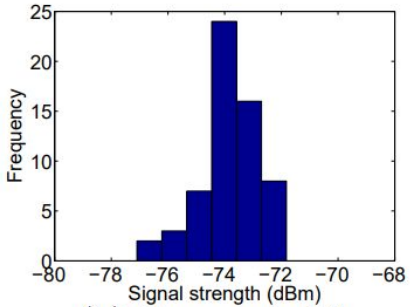
The Hinge loss is better in than MSE in 10 out of 12 case

Wifi localization dataset

- Goal: Given a signal from several access points, predict the location of a device in a hallway
- Two adaptation types considered:
  - Transfer across time periods
  - Transfer across devices used for collection (hallway 1, 2, and 3)
- For experiment, randomly sample 60% of source and target domain to use.
- Prediction considered correct within three meters for period and 6 meters for device



(a) time period 1.



(b) time period 2.

Experimental Results - Regression

Table 3: Comparison of different methods on the Wifi localization dataset. Maximum value in bold.

Domains	KRR	SurK	DIP	DIP-CC	GeTarS	CTC	CTC-TIP	JDOT
Transfer across periods								
t1 → t2	80.84±1.14	90.36±1.22	87.98±2.33	91.30±3.24	86.76 ± 1.91	89.36±1.78	89.22±1.66	**93.03 ± 1.24**
t1 → t3	76.44±2.66	**94.97±1.29**	84.20±4.29	84.32±4.57	90.62±2.25	94.80±0.87	92.60 ± 4.50	90.06 ± 2.01
t2 → t3	67.12±1.28	85.83 ± 1.31	80.58 ± 2.10	81.22 ± 4.31	82.68 ± 3.71	87.92 ± 1.87	**89.52 ± 1.14**	86.76 ± 1.72
Transfer across devices								
hallway1	60.02 ±2.60	76.36 ± 2.44	77.48 ± 2.68	76.24± 5.14	84.38 ± 1.98	86.98 ± 2.02	86.78 ± 2.31	**98.83±0.58**
hallway2	49.38 ± 2.30	64.69 ±0.77	78.54 ± 1.66	77.8± 2.70	77.38 ± 2.09	87.74 ± 1.89	87.94 ± 2.07	**98.45±0.67**
hallway3	48.42 ±1.32	65.73 ± 1.57	75.10± 3.39	73.40± 4.06	80.64 ± 1.76	82.02± 2.34	81.72 ± 2.25	**99.27±0.41**

JDOT achieve almost perfect accuracy on Transfer across Devices

Presentation Overview

- Background
- Related Works
- Methodology
- Theoretical Analysis
- Results
- Conclusion

Conclusion

- Prior work generally only considered the optimal transport plan between the source and target feature distribution.
  - This work proposes using the joint distribution of target and labels.
- How do we use the optimal transport between labels if we don’t have them?
  - We learn a function *f* to guess them: Guessing the labels is the task anyway.
- *f* is hypothesis agonistic:
  - Works on neural networks, linear classifiers, kernel methods
- Results show this method is competitive with/outperforms previous work

Critiques and Praise

Pros:

- First metric to consider the labels when aligning with wasserstein distance
- Is model agnostic: can work with any kind of architecture

Cons (Future work):

- The bounds introduced by the paper are loose
- **Semi-supervised** extension (using unlabelled examples in source domain)
- Better stochastic techniques for solving **efficiently** the adaptation

Questions?

References

1. M. Gong, K. Zhang, T. Liu, D. Tao, C. Glymour, and B. Schölkopf. Domain adaptation with conditional transferable components. In ICML, volume 48, pages 2839–2848, 2016.
2. B. Gong, Y. Shi, F. Sha, and K. Grauman. Geodesic flow kernel for unsupervised domain adaptation. In CVPR, 2012.
3. Y. Ganin and V. Lempitsky. Unsupervised domain adaptation by backpropagation. In ICML, pages 1180–1189, 2015.
4. M. Long, J. Wang, G. Ding, J. Sun, and P. Yu. Transfer joint matching for unsupervised domain adaptation. In CVPR, pages 1410–1417, 2014.
5. M. Gong, K. Zhang, T. Liu, D. Tao, C. Glymour, and B. Schölkopf. Domain adaptation with conditional transferable components. In ICML, volume 48, pages 2839–2848, 2016.
6. N.Courty,R.Flamary,andD.Tuia.Domainadaptationwithregularizedoptimaltransport.In ECML/PKDD, 2014.
7. Zhang, Kai, et al. "Covariate shift in hilbert space: A solution via sorrogate kernels." International Conference on Machine Learning. PMLR, 2013.