Transfer Learning/Domain Adaptation: Principles and Recent Advances

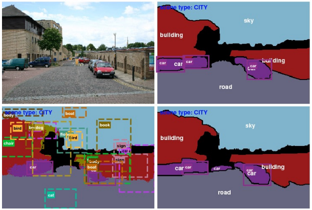
Amaury Habrard

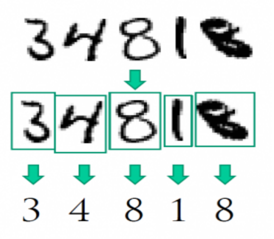
Laboratoire Hubert Curien, UMR CNRS 5516, Université de Saint-Etienne
amaury.habrard@univ-st-etienne.fr

LIMOS
15 novembre 2018

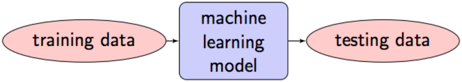
Goals in AI

- **Ultimate goal**: Build systems that can **learn** by exploring the world
  $\rightarrow$ Unfortunately not easy or almost impossible for now
- **Intermediate goal**: Build systems that can classify and recognize well





- **Solution**: Use Machine learning (ML) methods = near-human performance



Issues of Traditional ML

Issues:

- near-human performance is achieved using lots of labeled data
- Some tasks do not have that much labeled data (biology, physics etc) $\rightarrow$ sample bias
- Some data/tasks evolve with time
- There exist too many tasks!

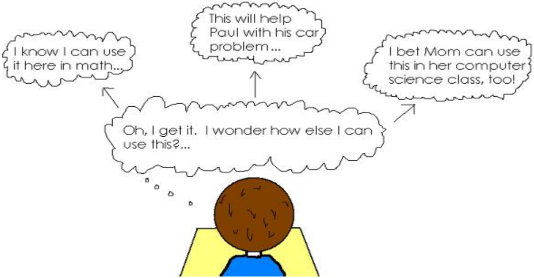
Issues of Traditional ML

Issues:

- near-human performance is achieved using lots of labeled data
- Some tasks do not have that much labeled data (biology, physics etc) $\rightarrow$ sample bias
- Some data/tasks evolve with time
- There exist too many tasks!

Solution: Transfer learning

+ Use systems build for different but related applications



Transfer Learning

Definition [Pan, TL-IJCAI'13 tutorial]

Ability of a system to recognize and apply knowledge and skills learned in previous domains/tasks to novel domains/tasks

An example

- We have **labeled** images from a **Web image** corpus
- Is there a `Person` in **unlabeled** images from a **Video** corpus ?

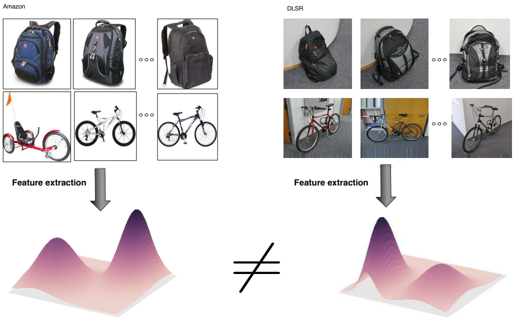
`Person`

`no Person`

?

Is there a `Person`?

Domain Adaptation problem - object detection

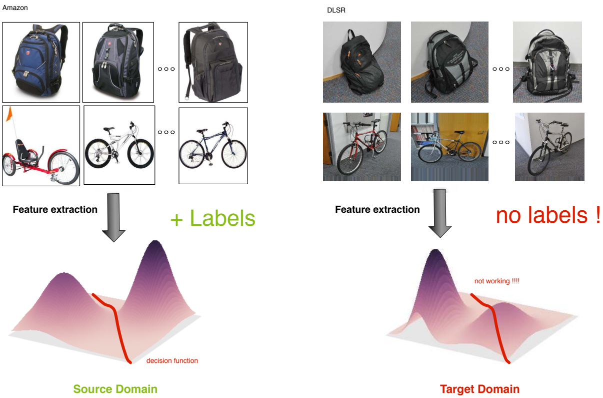


Probability Distribution Functions over the domains

Our context

- Classification problem with data coming from different sources (domains).
- Distributions are different but related.

Domain adaptation problem - object detection



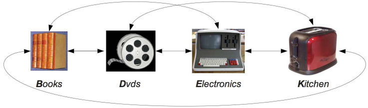
Problems

- Labels only available in the **source domain**, and classification is conducted in the **target domain**.
- Classifier trained on the source domain data performs badly in the target domain: **training and test distributions are different**!

Domain adaptation problem - spam filtering

We aim at learning a spam filter from the mailing box of Bob and deploying the model over the emails received by Alice.

Domain adaptation problem - sentiment analysis



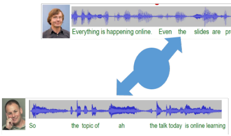
	Electronics	Video games
	(1) Compact; easy to operate; very good picture quality; looks sharp!	(2) A very good game! It is action packed and full of excitement. I am very much hooked on this game.
	(3) I purchased this unit from Circuit City and I was very excited about the quality of the picture. It is really nice and sharp.	(4) Very realistic shooting action and good plots. We played this and were hooked.
	(5) It is also quite blurry in very dark settings. I will never_buy HP again.	(6) It is so boring. I am extremely unhappy and will probably never_buy UbiSoft again.

- Source specific: *compact, sharp, blurry*.
- Target specific: *hooked, realistic, boring*.
- Domain independent: *good, excited, nice*, *never_buy, unhappy*.

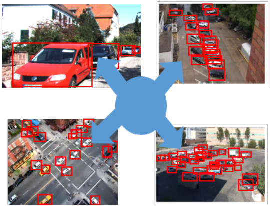
Other examples of applications

- Speech recognition: Adapt to different accents

Speach recognition

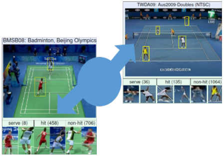


- Object Detection

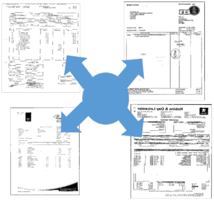


- Action recognition

Action recognition



- Document categorization



- medecine, physics, NLP, ...

Images from [Csurka et al., 2017]

Does it work?

Yes it helps! [Courty el al., 2017]

Domains	**Base**	**SurK**	**SA**	**ARTL**	**OT-IT**	**OT-MM**	**Tloss**
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	–

Outline

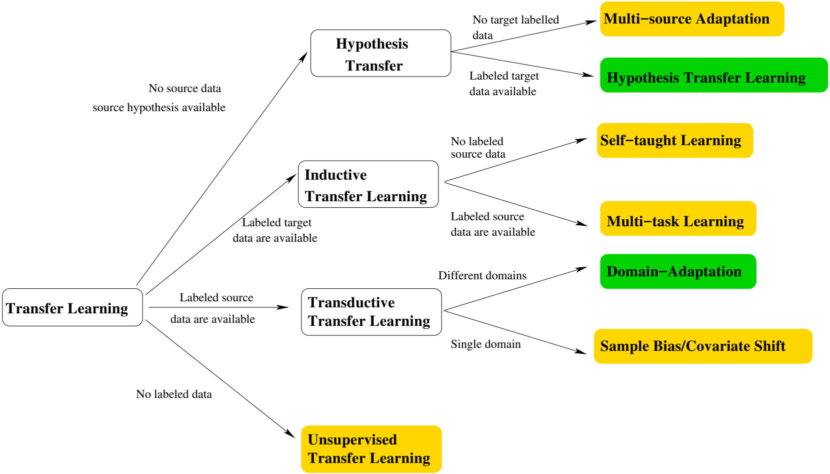
- Definition
- A first approach: co-variate shift
- When domain adaptation can work
- Some Domain Adaptation methods
  - Iterative approaches
  - Optimal Transport
  - Subspace Alignment
  - A quick work on Deep learning
- Hypothesis Transfer Learning

Acknowledgements: Basura Fernando, Nicolas Courty, Rémi Flamary, Pascal Germain, Emilie Morvant, Michaël Perrot, JP Peyrache, Ievgen Redko, Marc Sebban

Transfer Learning/Domain Adaptation

Definition (Pan and Yang 2010)

Given a source domain S and learning task $Y_{\mathcal{S}}$, a target domain T and learning task $Y_{\mathcal{T}}$, **transfer learning** aims to help improve the learning of the target predictive function $f_{\mathcal{T}}$ in $D_{\mathcal{T}}$ using the knowledge in S and T, where $S \neq T$ or $Y_{\mathcal{S}} \neq Y_{\mathcal{T}}$.



Classic setting in domain adaptation

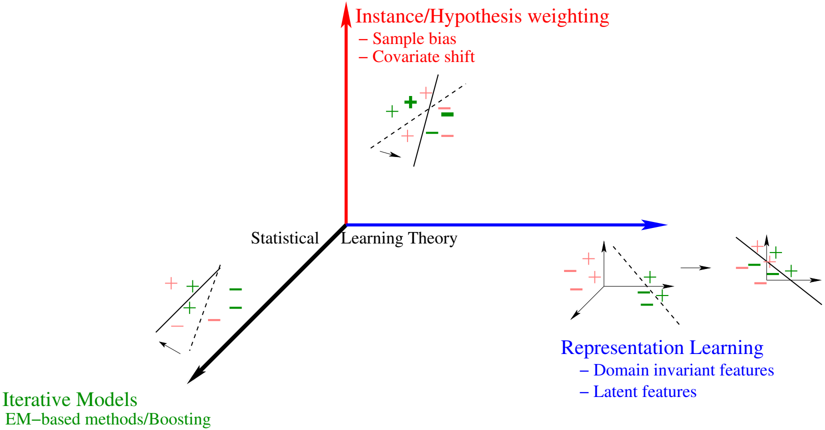
Statistical learning

- A feature space $\mathcal{X}$, label set $\mathcal{Y} = \{-1, 1\}$.
  P_S distribution over $\mathcal{X} \times \mathcal{Y}$, P_T distribution over $\mathcal{X} \times \mathcal{Y}$
- An unknown labeling function $f : \mathcal{X} \to \mathcal{Y}$ that follows $P_T(y|\mathbf{x})$
- A source training set $LS = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m)\} \subset (\mathcal{X} \to \mathcal{Y})^m$ drawn i.i.d. from P_S.
  A target unlabeled set $LT = \{\mathbf{x}_i\}_{i=1}^{n_t}$ drawn i.i.d. from the marginal P_T over $\mathcal{X}$ D_T
- Learn a classifier (or a hypothesis) $h \in \mathcal{H} \subseteq \mathcal{Y}^{\mathcal{X}}$ as close as possible to the unknown function f.
- True source risk: $\epsilon_S(h) = \mathbb{E}_{(\mathbf{x},y)\sim P_S}[h(\mathbf{x}) \neq y]$,
  Empirical source risk over LS: $\hat{\epsilon}_S(h) = \sum_{(\mathbf{x},y)\in LS}[h(\mathbf{x}) \neq y]$.
  True target risk: $\epsilon_T(h) = \mathbb{E}_{(\mathbf{x},y)\sim P_T}[h(\mathbf{x}) \neq y]$

Classic guarantee in supervised ML: $\epsilon_S(h) \leq \hat{\epsilon}_S(h) + \sqrt{\frac{\text{complexity}(h\in\mathcal{H})}{|LS|}}$

$\Rightarrow$ **but we want to be good** on P_T

Main strategies in DA



Reweighting methods

A first analysis

$$
\begin{aligned}
\epsilon_T(h) &= \mathop{\mathbf{E}}_{(\mathbf{x}^t,y^t)\sim P_T} \mathbf{I}\big[h(\mathbf{x}^t) \neq y^t\big] \\
&= \mathop{\mathbf{E}}_{(\mathbf{x}^t,y^t)\sim P_T} \frac{P_S(\mathbf{x}^t,y^t)}{P_S(\mathbf{x}^t,y^t)} \mathbf{I}\big[h(\mathbf{x}^t) \neq y^t\big] \\
&= \sum_{(\mathbf{x}^t,y^t)} P_T(\mathbf{x}^t,y^t) \frac{P_S(\mathbf{x}^t,y^t)}{P_S(\mathbf{x}^t,y^t)} \mathbf{I}\big[h(\mathbf{x}^t) \neq y^t\big] \\
&= \mathop{\mathbf{E}}_{(\mathbf{x}^t,y^t)\sim P_S} \frac{P_T(\mathbf{x}^t,y^t)}{P_S(\mathbf{x}^t,y^t)} \mathbf{I}\big[h(\mathbf{x}^t) \neq y^t\big]
\end{aligned}
$$

Assume similar tasks - covariate shift, $P_S(y|\mathbf{x}) = P_T(y|\mathbf{x})$, then:

$$
\begin{aligned}
&= \mathop{\mathbf{E}}_{(\mathbf{x}^t,y^t)\sim P_S} \frac{D_T(\mathbf{x}^t)P_T(y^t|\mathbf{x}^t)}{D_S(\mathbf{x}^t)P_S(y^t|\mathbf{x}^t)} \mathbf{I}\big[h(\mathbf{x}^t) \neq y^t\big] \\
&= \mathop{\mathbf{E}}_{(\mathbf{x}^t,y^t)\sim P_S} \frac{D_T(\mathbf{x}^t)}{D_S(\mathbf{x}^t)} \mathbf{I}\big[h(\mathbf{x}^t) \neq y^t\big]
\end{aligned}
$$

Covariate shift [Shimodaira,'00]

$\Rightarrow$ With covariate shift, $P_S(y|\mathbf{x}) = P_T(y|\mathbf{x})$, we have:

$$= \mathop{\mathbf{E}}_{(\mathbf{x}^t)\sim D_S} \frac{D_T(\mathbf{x}^t)}{D_S(\mathbf{x}^t)} \mathop{\mathbf{E}}_{y^t\sim P_S(y^t|\mathbf{x}^t)} \mathbf{I}\left[h(\mathbf{x}^t) \neq y^t\right]$$

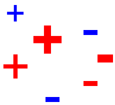
$\Rightarrow$ **weighted error** on the source domain: $\omega(\mathbf{x}^t) = \frac{D_T(\mathbf{x}^t)}{D_S(\mathbf{x}^t)}$

Idea: reweight labeled source data according to an estimate of $\omega(\mathbf{x}^t)$:

$$\mathop{\mathbf{E}}_{(\mathbf{x}^t,y^t)\sim P_S} \omega(\mathbf{x}^t)\mathbf{I}\left[h(\mathbf{x}^t) \neq y^t\right]$$

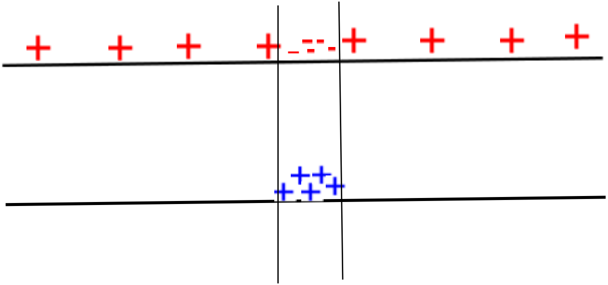
Learn a classifier on a sample reweighted w.r.t. $\hat{\omega}$

$$\sum_{(\mathbf{x}_i^s,y_i^s)\in S} \hat{\omega}(\mathbf{x}_i^s)I[h(\mathbf{x}_i^s) \neq y_i^s]$$



Bad news

- DA is hard, even under covariate shift [Ben-David et al.,ALT'12]
  $\Rightarrow$ To learn a classifier the number of examples depend on $|\mathcal{H}|$ (finite) or exponentially on the dimension of $\mathcal{X}$
- Co-variate shift assumption may fail:
  Tasks are not similar in general $P_S(y|\mathbf{x}) \neq P_T(y|\mathbf{x})$



When domain adaptation can work?

Theoretical guarantees

Theoretical bounds [Ben-david et al., 2007,2010]

The error performed by a given classifier h in the target domain $\epsilon_T(h)$ is upper-bounded by the sum of three terms :

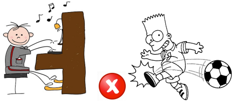
$$\epsilon_T(h) \leq \epsilon_S(h) + Div(\mu_s, \mu_t) + \lambda$$

- Error of the classifier in the source domain $\epsilon_S(h)$
  $\rightarrow$ can be optimized efficiently with supervised ML
- Divergence measure between the two domains $Div(\mu_s, \mu_t)$
  $\rightarrow$ key element to care about
- A third term measuring how much the classification tasks are related to each other.
  $\rightarrow$ Cross fingers and hope that it is small

$\Rightarrow$ **A natural approach is then to move closer the two distributions while ensuring a low-error on the source domain**

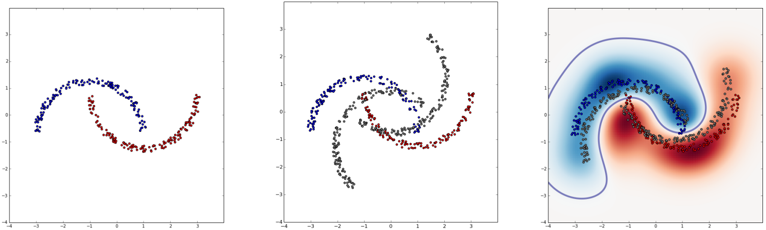
A Strong Assumption!

When can it work? - small λ



- $\lambda = \epsilon_T(h^*) + \epsilon_S(h^*)$
  with $h^* = \text{argmin}_{h\in\mathcal{H}}\epsilon_T(h) + \epsilon_S(h)$ [Ben-David et al., 2007;2010]

$\Rightarrow$ There must exist a good hypothesis on the two domains (relatedness), or two good hypotheses -one on each domain- and close with respect to the target distribution [Mansour et al., 2009]



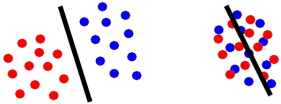
Divergences

H-divergence/Discrepancy

- Related to the hypothesis class $\mathcal{H}$

$$d_{\mathcal{H}\Delta\mathcal{H}}(D_s, D_T) = \sup_{(h,h')\in\mathcal{H}^2} \left|\epsilon_T(h,h') - \epsilon_S(h,h')\right|$$

$$= \sup_{(h,h')\in\mathcal{H}^2} \left| \mathbb{E}_{\mathbf{x}^t\sim D_T}\left[h(\mathbf{x}^t) \neq h'(\mathbf{x}^t)\right] - \mathbb{E}_{\mathbf{x}^s\sim D_S}\left[h(\mathbf{x}^s) \neq h'(\mathbf{x}^s)\right] \right|$$



- $\Rightarrow$ Adaptation is better if domains cannot be distinguished with respect to $\mathcal{H}$
- Allows one to derive uniform convergence-like bounds (VC-dimension) or Rademacher bounds.

Divergences

Weighted average over $\mathcal{H}$

- averaged distance $\text{dis}_\rho(D_S, D_T) = \left| \underset{h,h' \sim \rho^2}{\mathbf{E}} [R_{D_T}(h,h') - R_{D_S}(h,h')] \right|$
- Similar generalization bound
  $\underset{h\sim\rho}{\mathbf{E}} R_{P_T}(h) \leq \underset{h\sim\rho}{\mathbf{E}} R_{P_S}(h) + \text{dis}_\rho(D_S, D_T) + \lambda_{\rho^*}$
  Controlled by PAC-Bayesian theory [Germain et al., 13;16]

Without the Hypothesis class

- Maximum Mean Discrepancy [Huang et al.,06]

$$MMD(D_S, D_T) = |\underset{\mathbf{x}^s \sim D_S}{\mathbb{E}} \phi(\mathbf{x}^s) - \underset{\mathbf{x}^t \sim D_T}{\mathbb{E}} \phi(\mathbf{x}^t)|$$

- Rényi Divergence [Mansour et al., UAI'09]

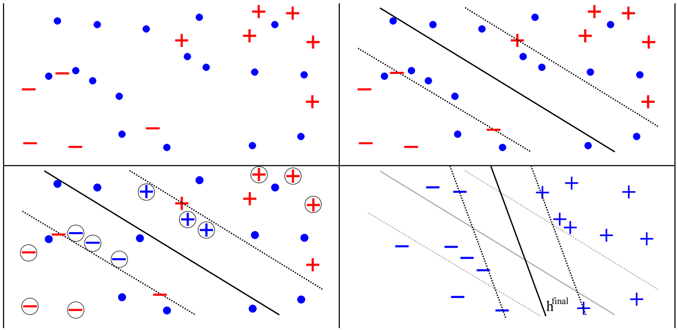
$$D_\alpha(D_S, D_T) = \frac{1}{\alpha - 1} \log \sum_{\mathbf{x}} \frac{D_S(\mathbf{x})^\alpha}{D_T(\mathbf{x})^{\alpha-1}}$$

Adjusting/Iterative methods

Principle

- Integrate some information about the target samples iteratively
  $\Rightarrow$ use of pseudo-labels
- "Move" closer distributions
  $\Rightarrow$ Remove/add some instances $\Rightarrow$ take into account a divergence measure
- Repeat the process until convergence or no remaining instances

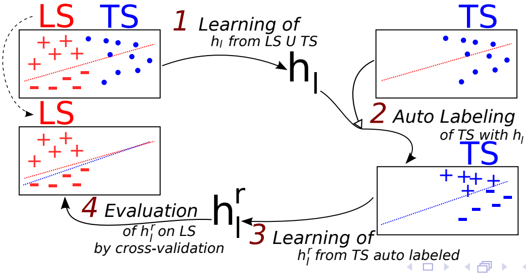
(e.g. DASVM [Bruzzone et al.,'10])



Convergence ?

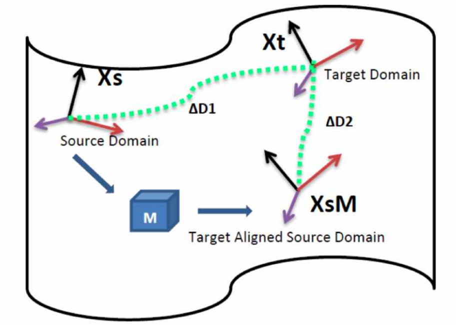
Almost no theoretical guarantees

- Weak classifier assumption: each new classifier must do better than random guessing on the data it has been learned from on both domains
- At least one classifier must do better than no adaptation during iterations
- Control the balance between classes
- Use “soft” labels (limit negative transfer)
- Other idea: reverse validation.



Subspace Alignments

Subspace alignment [Fernando et al.,ICCV'13]



- Extract a source subspace using the first d eigen vectors
- Extract a target subspace using the first d eigen vectors
- Learn a linear function that aligns the source subspace with the linear one
- Totally unsupervised

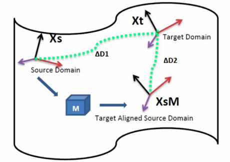
Subspace alignment algorithm

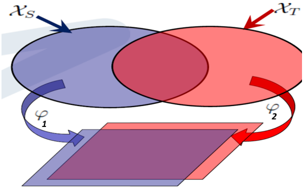
Algorithm 1: Subspace alignment DA algorithm

Data: Source data S, Target data T, Source labels Y_S, Subspace dimension d
Result: Predicted target labels Y_T
$\mathbf{S}_1 \leftarrow PCA(S, d)$ (source subspace defined by the first d eigenvectors) ;
$\mathbf{S}_2 \leftarrow PCA(T, d)$ (target subspace defined by the first d eigenvectors);
$\mathbf{X}_a \leftarrow \mathbf{S}_1 \mathbf{S}_1{}' \mathbf{S}_2$ (operator for aligning the source subspace to the target one);
$\mathbf{S}_a = S\mathbf{X}_a$ (new source data in the aligned space);
$\mathbf{T}_T = T\mathbf{S}_2$ (new target data in the aligned space);
$Y_T \leftarrow Classifier(\mathbf{S}_a, \mathbf{T}_T, Y_S)$;

- $\mathbf{M}^* = \mathbf{S}_1{}'\mathbf{S}_2$ corresponds to the "subspace alignment matrix": closed-form solution of $\mathbf{M}^* = \mathrm{argmin}_{\mathbf{M}} \|\mathbf{S}_1\mathbf{M} - \mathbf{S}_2\|$
- $X_a = \mathbf{S}_1\mathbf{S}_1{}'\mathbf{S}_2 = \mathbf{S}_1\mathbf{M}^*$ projects the source data to the target subspace
- A natural similarity: $\mathrm{Sim}(\mathbf{x}_s, \mathbf{x}_t) = \mathbf{x}_s\mathbf{S}_1\mathbf{M}^*\mathbf{S}_1{}'\mathbf{x}_t' = \mathbf{x}_s\mathbf{A}\mathbf{x}_t'$

A simple approach





Pros

- Very simple and intuitive method
- Totally unsupervised
- Theoretical result on the dimensionality detection

Cons

- Assumes that all source and target instances are relevant
- Cannot be directly kernelizable by using k-PCA
- Can be improved by using landmarks-selection to project data in a non linear space, and by using labels

$\Rightarrow$ Many approaches try to look for latent space moving closer source and target

Domain Adaptation with Optimal Transport

Optimal Transport

Figure: **Monge problem**

$$\inf_{T\#\mu=\nu} \int_\Omega D(x, T(x))\,\mu(dx)$$

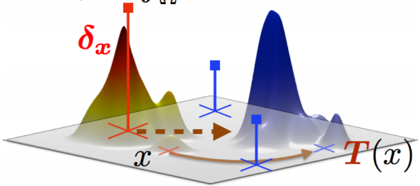
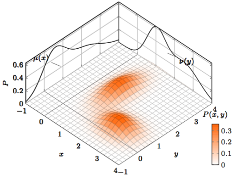
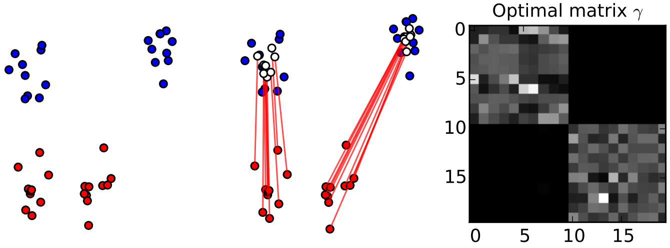


Figure: **Kantorovich relaxation**

$$\Pi(\mu,\nu) \stackrel{\text{def}}{=} \{P \in \mathcal{P}(\Omega\times\Omega) \mid \forall A, B \subset \Omega,\ P(A\times\Omega) = \mu(A),\ P(\Omega\times B) = \nu(B)\}$$



Domain Adaptation with Optimal Transport



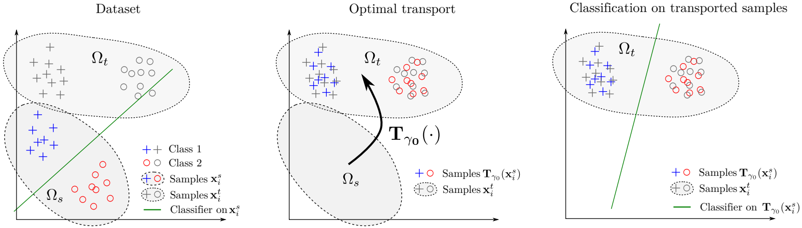
Alignment with optimal transport [Courty et al., '14-'16]

- Find an alignment that minimizes the cost of transportation between source and target
- Optimal transport (Wasserstein distance)

$$W(P_s, P_t) = \min_\gamma \int_{\Omega_s \times \Omega_t} c(x_s, x_t)\gamma(x_s, x_t)dx_s dx_t$$

such that $\int_{\Omega_t} \gamma(x_s, x_t)dx_t = P_s$ and $\int_{\Omega_s} \gamma(x_s, x_t)dx_s = P_t$, where c is a distance/cost function (*i.e.* euclidean distance).

Optimal transport for DA [Courty et al, 2016]



Assumptions

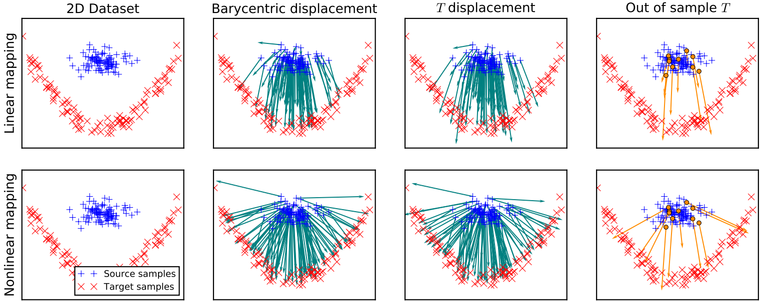
- There exist a transport $\mathbf{T}$ between the source and target domain.
- The transport preserves the conditional distributions (covariate shift):
$$P_s(y|\mathbf{x}_s) = P_t(y|\mathbf{T}(\mathbf{x}_s)).$$

3-step strategy

1. Estimate optimal transport between distributions: $W(\hat{D}_S, \hat{D}_T)$
2. Transport the training samples onto the target distribution $\widehat{\mathbf{x}}_i^S = \mathrm{argmin}_{\mathbf{x}} \quad \sum_j \gamma(i,j)c(\mathbf{x}, \mathbf{x}_j^t)$.
3. Learn a classifier on the transported training samples.

Improvements

- A bound similar to Ben-David et al.'s thm can be obtained
  $\epsilon_T(h) \leq \epsilon_S(h) + W(D_S, D_T) + \lambda$
- We can use regularizers to force examples of the same class to be grouped or to allow efficient optimization scheme
- The transport must be computed for each new sample, one solution is to learn a mapping that estimate the transport [Perrot et al., 2016]



Joint distribution optimal transport

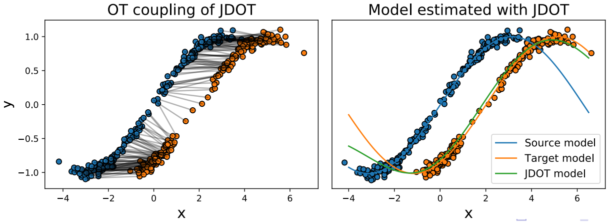
- The model does not include the classifier $\rightarrow$ JDOT [Courty et al., 2017] uses a transport taking into account labels:

  $W(\hat{P}_s, \hat{P}_t^f) = \inf_\gamma \sum_{i,j} c([x_i^s; y_i^s], [x_j^t; f(x_j^t)])\gamma(i,j)$

$$min_{f,\gamma} \sum_{i,j} \left(\alpha d(x_i^s, x_j^t) + \ell(y_i^s, f(x_j^t))\right) \gamma(i,j) + \lambda\|f\|$$

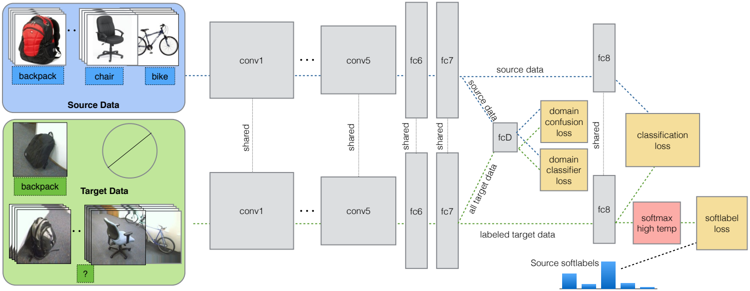
- Theoretical justification under an hypothesis of probabilistic lipschitzness: 2 close examples associated wrt to a joint distribution Π must have similar labels with high proba $1 - \phi(\lambda)$:

$$\epsilon_T(f) \leq W(\hat{P}_s, \hat{P}_t^f) + O(\frac{1}{\sqrt{m_s}} + \frac{1}{\sqrt{m_t}}) + \lambda + M\phi(\lambda)$$



Deep Domain Adaptation

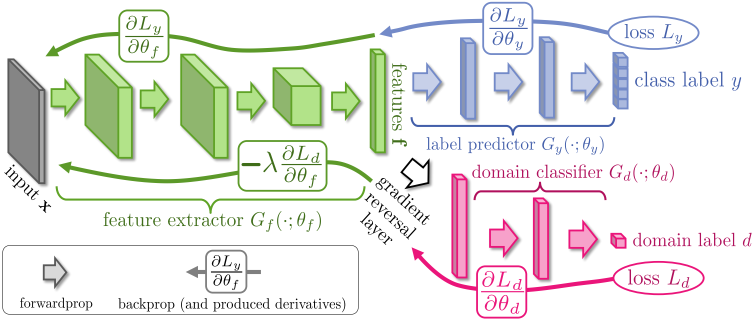
Deep Learning and DA



From [Hoffman et al., 2017]

- Lots of work: achieve state of the art
- Many strategies to find good representations to transfer tasks

Deep Learning - adversarial strategy

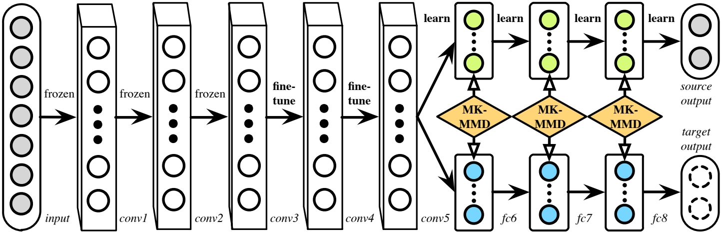


Idea of adversarial Learning [Ganin et al., 2015, 2016]

- Find a representation where source and target cannot be discriminated
- while ensuring a good performance on source.

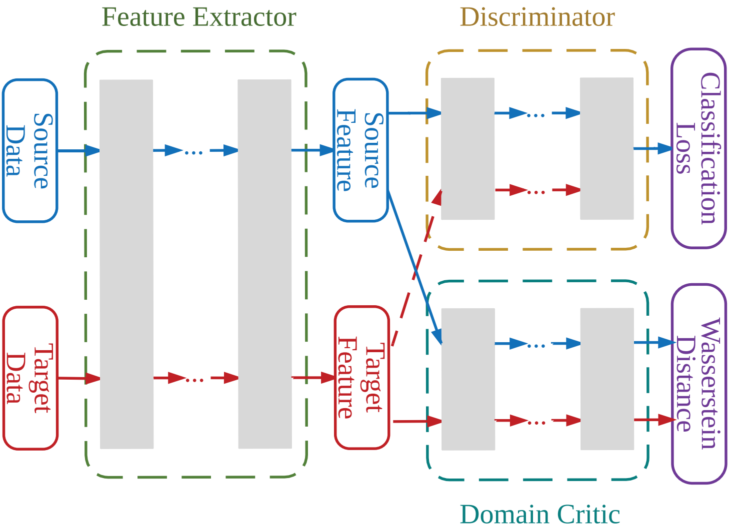
Deep Learning - adversarial strategy

More complex architecture [Long et al., ICML'15]



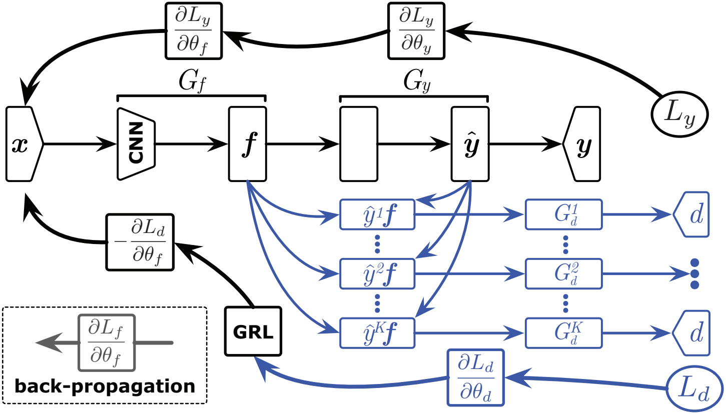
Deep Learning - adversarial strategy

More complex architecture [Shen et al., AAAI'18]



Deep Learning - adversarial strategy

More complex architecture [Pei et al., AAAI'18]



Hypothesis Transfer Learning

Motivation

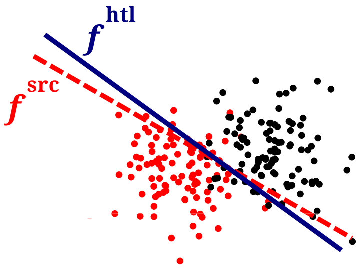
Drawback of classic domain adaptation

- Need to store source data to perform adaptation
- For each new domain, the adaptation process we must retrain with all source data: prohibitive when the number of domains is large
- Need to take into account the distribution shift

Hypothesis Transfer Learning

- We keep only source hypotheses from the source domain
- No explicit access to source domain (data, distribution)
- We require some target labeled data

Motivation



Biased regularized learning

- Given a source hypothesis h_S (or weighted combination of source hypotheses)
- Labeled target training set $LT = \{(\mathbf{x}_i, y_i)\}_{i=1}^m$
- Optimization problem:

$$\mathrm{argmin}_{h \in \mathcal{H}} \sum_{i=1}^{m} \frac{1}{m} \ell(h(\mathbf{x}_i), y_i) + \lambda \| h - h_S \|$$

Some Guarantees

- Strongly convex regularizer $\|\cdot\|$
- Smooth, convex, Lispchitz loss function
- Guarantee (simplified) obtained with Algorithmic Stability framework [Kuzborskij et al., 2013, 2017]:

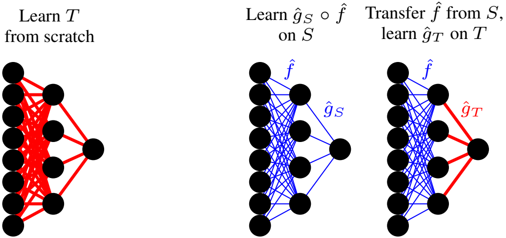
$$\epsilon_T(h) \leq \hat{\epsilon}_{LT}(h) + O\left(\frac{\sqrt{H \times \epsilon_T(h_S)}}{m\lambda}\right) + O\left(\frac{1}{m}\right)$$

with $H \leq m\lambda$

Implications

- If h_S is a bad fit, bound is similar to standard bounds
- If $\epsilon_T(h_S)$ is small enough, the bound is better - less examples required- and can even tend to a fast rate $O(1/m)$

Representation Transfer from NN



"Result" of McNamara and Balcan, ICML'17

$$\epsilon_T(\hat{g}_T \cdot \hat{f}) \leq \omega \left(\hat{\epsilon_S}\left(\hat{g}_S \cdot \hat{f}\right) + 2O\left(\sqrt{\frac{VCdim(\mathcal{H})}{m_S}}\right)\right) + O\left(\sqrt{\frac{VCdim(g)}{m_T}}\right)$$

ω: measure of transferability

- Justify representation transfer
- Better guarantee than learning from scratch is $VCdim(g)$ is small

Other perspective: Transferability through SGD ? [kuzborskij, arxiv 2017] [Hardt et al., 2016]

Conclusion

Conclusion

- Transfer Learning is a key problem for a wide applicability of machine learning methods
- Many methods, good empirical results on some tasks
- The theoretical foundations are still insufficient to explain/justify transferability
  - Guarantees specific to the data/method?
  - What to optimize/transfer
- Parameter tuning
- The control of negative transfer
- Other areas: lifelong learning, concept drift, knowledge distillation, distributed models, reinforcement learning, ...

Still a lot to do in an important topic!