Multi-tasks learning and transfer and domain adaptation

Antoine Cornuéjols

AgroParisTech – INRA MIA 518

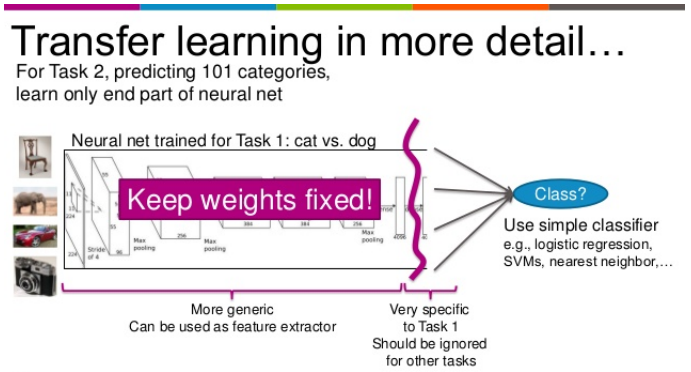
antoine.cornuejols@agroparistech.fr

Introduction

Motivation

Deep Neural Networks …

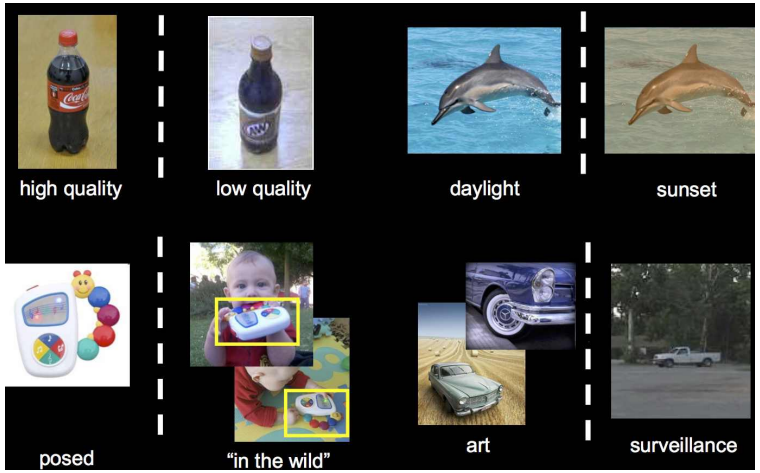
… are often trained from a version learnt on another task



Domain adaptation

- 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**
- Example
  - We have **labeled images** from a **web corpus**
  - Novel task: **is there a person** in unlabeled images from a **video corpus**?

Examples: transfer learning in vision



[Xu, Saenko, Tsang "Domain Transfer" tutorial – CVPR'12]

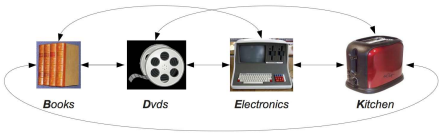
Natural Language Processing

Texts are represented by "words" (e.g. Bag of words)

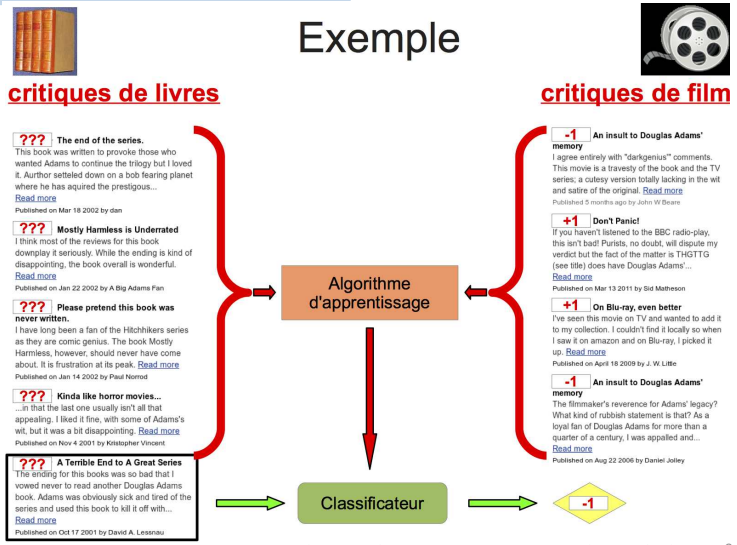
- Tasks
  - **Part of Speech Tagging**: Adapt a tagger learned from medical papers to a journal (Wall Street Journal) – Newsgroup

Biomedical	WSJ
the **signal** *required* to	**investment** *required*
stimulatory **signal** *from*	**buyouts** *from* buyers
essential **signal** *for*	to **jail** *for* violating

  - **Spam detection**: Adapt a classifier from one user to another
  - **Sentiment analysis**



Domain adaptation for sentiment analysis [Pan, TL-IJCAI'13 tutorial]



Domain adaptation for 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*.

[Pan, TL-IJCAI'13 tutorial]

Domain adaptation

- **Objective**
  - Improve a **target prediction function** in the target domain using knowledge from the **source domain**
- The **training** and **test set** can be from the **same domain**, but with different probability distributions
  - Co-variate shift
  - Concept drift
- Or they can be from **different domains**
  - Transfer

Notations

1. **Source** domain *S*
   - Source **training data** S_S
   - Source data **distribution** D_S
   - Source **hypothesis** h_S
2. **Target** domain *T*
   - Target **training data** S_T (|S_T| << |S_S|)
   - Target **data distribution** D_T
   - Target **hypothesis** h_T

Formalisation

- Domaine **cible** : $\mathcal{X}_\mathcal{T} \times \mathcal{Y}_\mathcal{T}$
  - Échantillon d'apprentissage $S_\mathcal{T} = \{(\mathbf{x}_i^\mathcal{T}, y_i^\mathcal{T})\}_{1 \leq i \leq m}$
  - Distribution $\mathbf{P}^T_{\mathcal{X}\mathcal{Y}}$
- Domaine **source** : $\mathcal{X}_\mathcal{S} \times \mathcal{Y}_\mathcal{S}$
  - Échantillon d'apprentissage $S_\mathcal{S} = \{(\mathbf{x}_i^\mathcal{S}, y_i^\mathcal{S})\}_{1 \leq i \leq m}$
  - Hypothèse source $h_\mathcal{S}$
- On cherche : $H_\mathcal{T} : \mathcal{X}_\mathcal{T} \rightarrow \mathcal{Y}_\mathcal{T}$
- **Algorithme** $A^{\text{htl}} : (\mathcal{X}_\mathcal{T} \times \mathcal{Y}_\mathcal{T})^m \times \mathcal{H}^\mathcal{S} \rightarrow \mathcal{H}^\mathcal{T} \subseteq \mathcal{Y}^\mathcal{X}$ d'apprentissage par transfert

Questions

Questions

1. **What** to transfer?
2. **How** to transfer?
3. What kind of **guarantees** on transfer learning can we get?
4. What are the **factors** affecting the success of transfer learning?
   - Quality of H_S?
   - Number of target training examples?
   - Similarity between S and T?

Transfer learning: questions

- **Sur quoi fonder** le transfert ?

  Comment traduire :

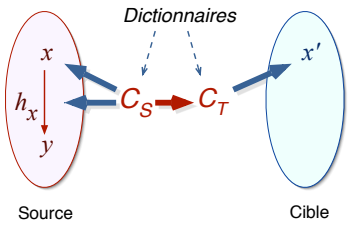
  « *le domaine cible est comme le domaine source* » ?

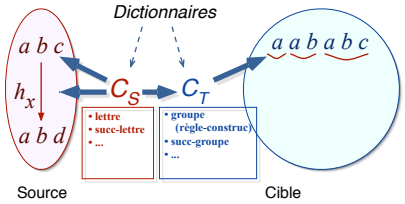
- Qu'est-ce qui **détermine sa pertinence** ?
  - Une « bonne source » ?
  - Une « proximité » étroite ?
- Comment mesurer la **valeur d'un apprentissage par transfert** ?
- Quelles **garanties** peut-on avoir sur l'hypothèse transférée ?

Le raisonnement par analogie

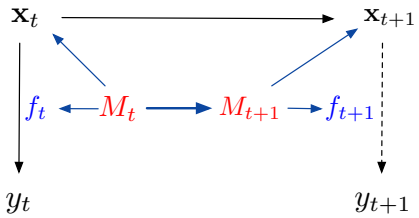
Adaptation de domaine & analogie

- Apprendre à la fois :
  - Une bonne **représentation**
    - Du domaine source
    - Du domaine cible
  - Une bonne règle de **transformation**





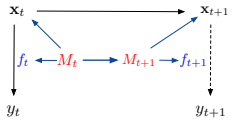
An approach to analogy: using Kolmogorov complexity



$$K(M_t) + K(\mathbf{x}_t|M_t) + K(f_t|M_t) + \underbrace{K(M_{t+1}|M_t)}_{\text{Change of system of reference}} + K(\mathbf{x}_{t+1}|M_{t+1}) + K(f_{t+1}|M_{t+1})$$

[Cornuéjols, 1996, 1997, 1998, 2016]

An approach to analogy: using Kolmogorov complexity

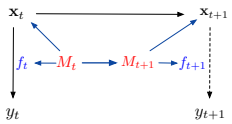


- Descripteurs utilisés dans la définition des structures :
  - orientation (-> / <-) — *1 bit*
  - cardinalité ou nombre d'éléments : n — *$\log_2(n)$ + 1 bits*
  - type d'éléments — (voir en-dessous)
  - longueur : l — *$\log_2(l)$ + 1 bits*
  - commençant ou se terminant par l'élément = x — *L(x) bits*
- **Lettre** — (1/2) -> *1 bit*
  Une lettre particulière (e.g. 'd') — (1/2.26) -> *6 bits*
- **Chaine** (orientation,éléments) — (1/8) -> *3 bits*
  L = 3 + L(orientation) + $\sum$ L(éléments)
  e.g. L('a3bd' avec orientation = ->) = 3 + 1 + $\log_2((1/2.26)^3)$ + L(3)
  = 3 + 1 + 18 + 3 = 25 bits
- **Ensemble** (type d'éléments, cardinalité, éléments) — (1/8) -> *3 bits*
  L = 3 + L(type) + L(cardinalité) + $\sum$ L(éléments)
- **Groupe** (type d'éléments, nombre d'éléments, éléments) — (1/8) -> *3 bits*
  L = 3 + L(type) + L(nb él.) + $\sum$ L(éléments)
- **Séquence** (orientation, type d'éléments, loi de succession ou nombre d'éléments, longueur, commençant ou se terminant par) — (1/8)
  L = 3+ L(orient.) + L(type) + L(loi) or L(nb él.) + L(long) + L(début/fin)
- Description et longueur d'une loi de **succession**
  succ(type-of-el.,n,x) = le nième successeur de l'élément x du type type-of-el.
  L = L(type) + L(n (voir ci-dessous)) + L(x)
  L(n) = L(1/6) si n=1 ou -1 (1er successeur ou prédécesseur)
  L(1/3) si n=0 (même élément)
  L((1/3).(1/2)^p) sinon (avec p=n si n≥0, p=-n sinon)
- Premier / Dernier (par rapport à l'orientation définie) — *1 bit*
- nième — *n bits*

$$K(M_t) + K(\mathbf{x}_t|M_t) + K(f_t|M_t) + \underbrace{K(M_{t+1}|M_t)}_{\text{Change of system of reference}} + K(\mathbf{x}_{t+1}|M_{t+1}) + K(f_{t+1}|M_{t+1})$$

An approach to analogy: using Kolmogorov complexity

[Cornuéjols, 1996, 1997, 1998, 2016]



'abc' = **Chaîne** (1/8)
orientation : -> (1/2)
1er='A', 2ème='B', 3ème='C' $(1/4.26)^3$
TOTAL (longueur) : **21** bits

'abc' = **Ensemble** (1/8)
{'A', 'B', 'C'} $(1/4.26)^3$
TOTAL : **20** bits

'abc' = **Séquence** (1/8)
orientation : -> (1/2)
type d'éléments = lettres (1/2)
loi de succession :
successeur(élt(lettre=x)) = élt(succ(lettre,1,x))
L(lettre) + L(1er succ) + L(x) = L(1/2 . 1/6 . 1)
= l(1/12) = 4 bits
longueur = 3 3 bits
commençant avec l'élément(lettre='A') (1/26)
TOTAL : **17** bits

An approach to analogy: using Kolmogorov complexity

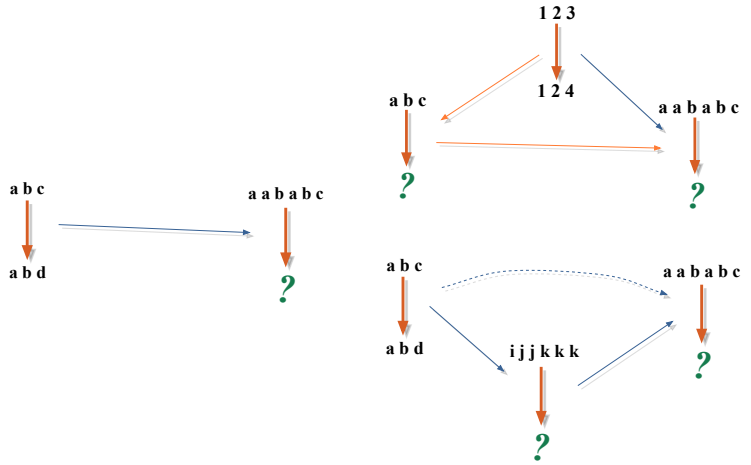
[Cornuéjols, 1996, 1997, 1998, 2016]

Problème 1 : **abc => abd ; iijjkk => ?**

Solution 1 : *"Remplacer groupe de droite par son successeur"* **iijjkk => iijjll**
Solution 2 : *"Remplacer lettre de droite par son successeur"* **iijjkk => iijjkl**
Solution 3 : *"Remplacer lettre de droite par D"* **iijjkk => iijjkd**
Solution 4 : *"Remplacer 3ème lettre par son successeur"* **iijjkk => iikjkk**
Solution 5 : *"Remplacer les C par D"* **iijjkk => iijjkk**
Solution 6 : *"Remplacer groupe de droite par la lettre D"* **iijjkk => iijjd**

	P1:S1	P1:S2	P1:S3	P1:S4	P1:S5	P1:S6
$L(M_S)$	10	9	11	11	12	11
$L(S_S \mid M_S)$	8	18	18	18	22	15
$L(\beta_S \mid M_S)$	4	4	3	7	8	3
$L(M_C \mid M_S)$	5	0	0	0	0	17
$L(S_C \mid M_C)$	8	36	36	36	42	15
$L(\beta_C \mid M_C)$	6	4	3	7	8	3
Total-1 (bits)	**41**	**71**	**71**	**79**	**93**	**65**
Total-2 (bits)	**35**	**67**	**68**	**72**	**85**	**62**
Rang	1	3	4	4	6	2

Transfer and sequence effects

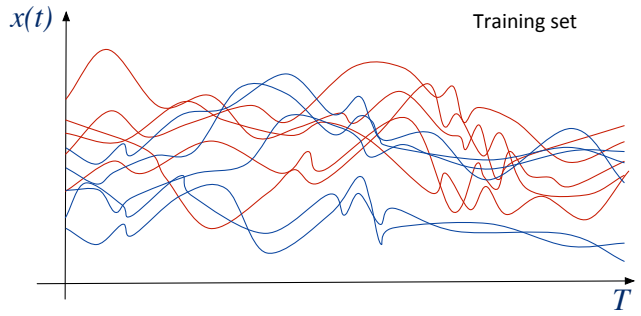


- toto

Une illustration et une approche de l'apprentissage par transfert

La **classification précoce** de séries temporelles

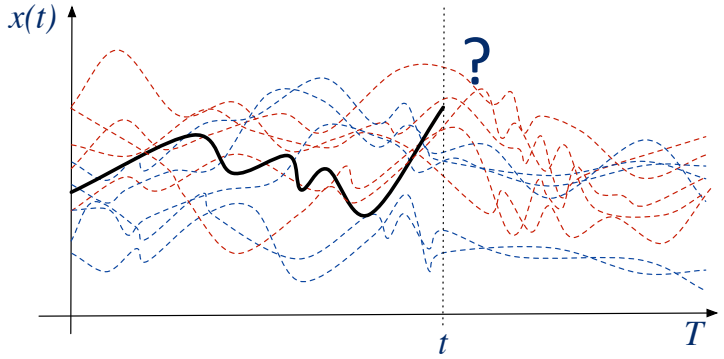
(Early) classification of time series



- Early prediction of daily ***electrical consumption***: high or low

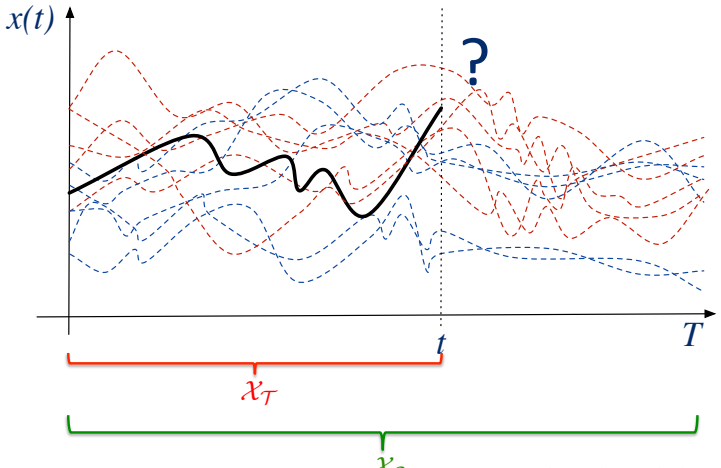
Early classification of time series

- What is the class of the new **incomplete** time series x_t?



Early classification of time series

- What is the class of the new **incomplete** time series x_t?



Principe

- Apprendre un classifieur sur les **données plus complètes**

$$S_\mathcal{S} = \{(\mathbf{x}_i^\mathcal{S}, y_i^\mathcal{S})\}_{1 \leq i \leq m} \rightarrow h_\mathcal{S}$$

- Chercher à l'utiliser pour classer des **séries incomplètes**

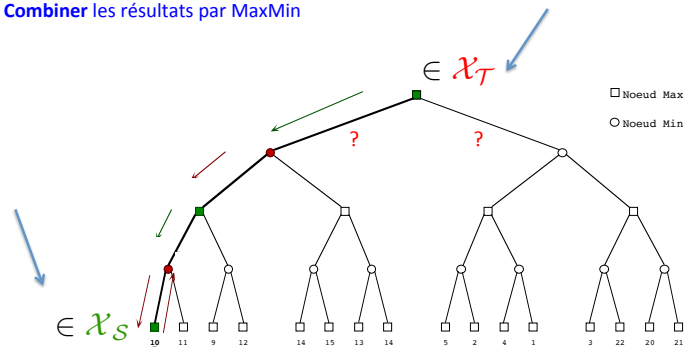
$$h_\mathcal{T} = \texttt{Fonction utilisant } h_\mathcal{S}$$

TransBoost

Transfert et algorithmes de jeux

Que jouer ?

- Fonction d'évaluation imparfaite dans $\mathcal{X}_S$
- L'utiliser dans $\mathcal{X}_T$
- **Combiner** les résultats par MaxMin



TransBoost

- Idée :
  - Apprendre des « *projections faibles* » : $\pi_i : \mathcal{X}_S \rightarrow \mathcal{X}_T$
    - À partir de : $S_S = \{(\mathbf{x}_i^S, y_i^S)\}_{1 \leq i \leq m}$
  - Par une méthode de boosting
    - Projection π_n telle que : $\varepsilon_n \doteq \mathbf{P}_{i \sim D_n}[h_S(\pi_n(\mathbf{x}_i)) \neq y_i] < 0.5$
    - Re-pondération des séries temporelles d'apprentissage
  - Résultat $H_T(\mathbf{x}^T) = \text{sign}\left\{\sum_{n=1}^{N} \alpha_n h_S(\pi_n(\mathbf{x}^T))\right\}$

TransBoost

Algorithm 1: Transfer learning by boosting

Input: $h_S : \mathcal{X}_S \rightarrow \mathcal{Y}_S$ the source hypothesis
$\mathcal{S}_T = \{(\mathbf{x}_i^T, y_i^T\}_{1 \leq i \leq m}$: the target training set

Initialization of the distribution on the training set: $D_1(i) = 1/m$ for $i = 1, \ldots, m$;

for $n = 1, \ldots, N$ **do**

Find a projection $\pi_i : \mathcal{X}_T \rightarrow \mathcal{X}_S$ st. $h_S(\pi_i(\cdot))$ performs better than random on $D_n(\mathcal{S}_T)$;
Let ε_n be the error rate of $h_S(\pi_i(\cdot))$ on $D_n(\mathcal{S}_T)$: $\varepsilon_n \doteq \mathbf{P}_{i \sim D_n}[h_S(\pi_n(\mathbf{x}_i)) \neq y_i]$ (with $\varepsilon_n < 0.5$) ;
Computes $\alpha_i = \frac{1}{2} \log_2\left(\frac{1-\varepsilon_i}{\varepsilon_i}\right)$;
Update, for $i = 1 \ldots, m$:

$$D_{n+1}(i) = \frac{D_n(i)}{Z_n} \times \begin{cases} e^{-\alpha_n} & \text{if } h_S(\pi_n(\mathbf{x}_i^T)) = y_i^T \\ e^{\alpha_n} & \text{if } h_S(\pi_n(\mathbf{x}_i^T)) \neq y_i^T \end{cases}$$
$$= \frac{D_n(i) \exp\left(-\alpha_n y_i^{(T)} h_S(\pi_n(\mathbf{x}_i^{(T)}))\right)}{Z_n}$$

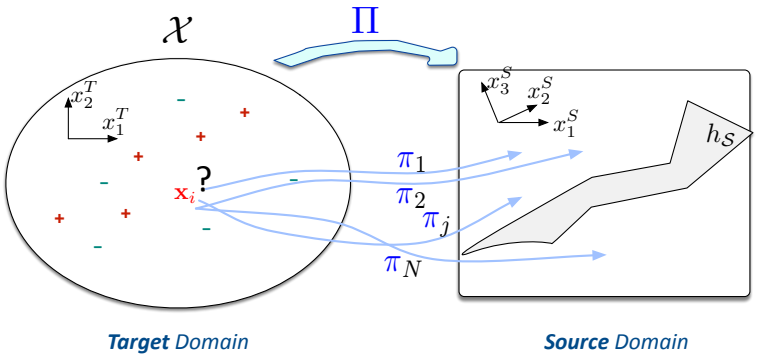
where Z_n is a normalization factor chosen so that D_{n+1} be a distribution on $\mathcal{S}_T$;

end

Output: the final target hypothesis $H_T : \mathcal{X}_T \rightarrow \mathcal{Y}_T$:

$$H_T(\mathbf{x}^T) = \text{sign}\left\{\sum_{n=1}^{N} \alpha_n h_S(\pi_n(\mathbf{x}^T))\right\} \quad (2)$$

TransBoost



$$H_{\mathcal{T}}(\mathbf{x}^{\mathcal{T}}) = \text{sign}\left\{\sum_{n=1}^{N} \alpha_n \, h_{\mathcal{S}}\left(\pi_n(\mathbf{x}^{\mathcal{T}})\right)\right\}$$

2ème question : (bons) ingrédients du transfert

La **qualité** d'un apprentissage par transfert

- **ne dépend pas** de la qualité de l'hypothèse source
- mais **dépend** de l'ensemble des projections

Experiments

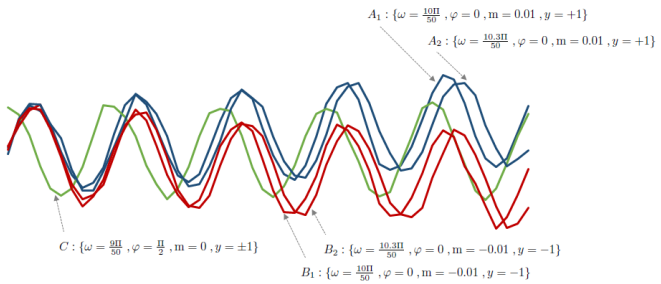
Objectifs des expériences

1. Tester la **validité** de la méthode proposée
2. Mesurer le rôle de la « **distance** » entre source et cible
3. Mesurer l'**importance de l'ensemble** Π de projections faibles

Données artificielles

- Control of
  - The time-dependent **information** provided to distinguish between **classes**
  - The shapes of **time series** within each class
  - The **noise level**

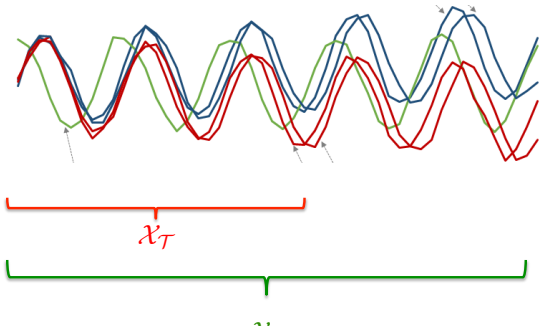
$$\mathbf{x}_t = \underbrace{t \times \text{slope} \times \text{class}}_{\text{information gain}} + \underbrace{\mathrm{x}_{max}\ \sin(\omega_i \times t + \varphi_j)}_{\text{sub shape within class}} + \underbrace{\eta(t)}_{\text{noise factor}}$$



Données artificielles

Un apprentissage par transfert

$$\mathbf{x}_t = \underbrace{t \times \text{slope} \times \text{class}}_{\text{information gain}} + \underbrace{\mathrm{x}_{max}\ \sin(\omega_i \times t + \varphi_j)}_{\text{sub shape within class}} + \underbrace{\eta(t)}_{\text{noise factor}}$$



Protocole expérimental

- Les **données** (chaque expérience)
  - 900 séries temporelles de **150** pas de temps (source)
    - 450 ('+') et 450 ('-')
    - 300 pour l'apprentissage
    - 600 en test
  - Cible : 20, 50 ou 70 pas de temps
  - Pente = 0.01 ; 0.002
  - Taux de bruit : 0.001 ; 0.01 ; 0.1 ; 05

L'espace des projections

- Ensemble de projections

Fonctions coude (5 paramètres)

- **Abscisse** du coude
- **Angles** avant et après
- **Fenêtre** prise en compte

Results

Learning from target data only | TransBoost | On the source domain | Naïve transfert

slope, noise, $t_\mathcal{T}$	$h_\mathcal{T}$ (train)	$h_\mathcal{T}$ (test)	$H_\mathcal{T}$ (train)	$H_\mathcal{T}$ (test)	$h_\mathcal{S}$ (test)	$H'_\mathcal{T}$ (test)
0.001, 0.001, 20	0.46 ± 0.02	0.50 ± 0.08	0.08 ± 0.03	$\mathbf{0.08} \pm 0.02$	0.05	0.49 ± 0.01
0.005, 0.001, 20	0.46 ± 0.02	0.49 ± 0.01	0.01 ± 0.01	$\mathbf{0.01} \pm 0.01$	0.01	0.45 ± 0.01
0.005, 0.002, 20	0.46 ± 0.02	0.49 ± 0.03	0.03 ± 0.02	$\mathbf{0.04} \pm 0.02$	0.02	0.43 ± 0.01
0.005, 0.02, 20	0.44 ± 0.02	0.48 ± 0.03	0.09 ± 0.01	$\mathbf{0.10} \pm 0.01$	0.01	0.47 ± 0.01
0.001, 0.2, 20	0.46 ± 0.02	0.50 ± 0.01	0.46 ± 0.02	0.51 ± 0.02	0.11	0.49 ± 0.01
0.01, 0.2, 20	0.42 ± 0.03	0.47 ± 0.03	0.34 ± 0.02	0.35 ± 0.02	0.02	0.35 ± 0.01
0.001, 0.001, 50	0.46 ± 0.02	0.50 ± 0.01	0.08 ± 0.03	$\mathbf{0.08} \pm 0.02$	0.06	0.41 ± 0.01
0.005, 0.001, 50	0.25 ± 0.07	0.28 ± 0.09	0.01 ± 0.01	$\mathbf{0.01} \pm 0.01$	0.01	0.28 ± 0.01
0.005, 0.002, 50	0.27 ± 0.07	0.30 ± 0.08	0.02 ± 0.01	$\mathbf{0.02} \pm 0.01$	0.02	0.28 ± 0.01
0.005, 0.02, 50	0.26 ± 0.07	0.30 ± 0.08	0.04 ± 0.01	$\mathbf{0.04} \pm 0.01$	0.01	0.31 ± 0.01
0.001, 0.2, 50	0.44 ± 0.02	0.50 ± 0.01	0.38 ± 0.03	0.44 ± 0.02	0.15	0.43 ± 0.01
0.01, 0.2, 50	0.10 ± 0.03	0.12 ± 0.04	0.10 ± 0.02	0.11 ± 0.02	0.03	0.15 ± 0.02
0.001, 0.001, 100	0.43 ± 0.03	0.47 ± 0.03	0.07 ± 0.02	$\mathbf{0.07} \pm 0.02$	0.02	0.23 ± 0.01
0.005, 0.001, 100	0.06 ± 0.03	0.07 ± 0.03	0.01 ± 0.01	$\mathbf{0.01} \pm 0.01$	0.01	0.07 ± 0.02
0.005, 0.002, 100	0.08 ± 0.03	0.10 ± 0.04	0.02 ± 0.01	$\mathbf{0.02} \pm 0.01$	0.02	0.07 ± 0.01
0.005, 0.02, 100	0.08 ± 0.03	0.09 ± 0.03	0.02 ± 0.01	$\mathbf{0.03} \pm 0.01$	0.01	0.07 ± 0.01
0.001, 0.2, 100	0.04 ± 0.03	0.46 ± 0.02	0.28 ± 0.02	0.31 ± 0.01	0.16	0.31 ± 0.01
0.01, 0.2, 100	0.03 ± 0.01	0.05 ± 0.02	0.04 ± 0.01	0.05 ± 0.01	0.02	0.05 ± 0.01

Table 1: Comparison of learning directly in the target domain (columns $h_\mathcal{T}$ (train) and $h_\mathcal{T}$ (test)), using `TransBoost` (columns $H_\mathcal{T}$ (train) and $H_\mathcal{T}$ (test)), learning in the source domain (column $h_\mathcal{S}$ (test)) and, finally, completing the time series with a SVR regression and using $h_\mathcal{S}$ (naïve transfer). Test errors are highlighted in the orange columns. Bold numbers indicates where `TransBoost` significantly dominates both learning without transfer and learning with naïve transfer.

Results

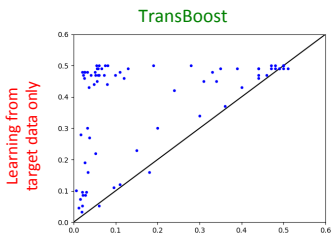


Figure 3: Comparison of error rates. y-axis: test error of the SVM classifier (without transfer). x-axis : test error of the TransBoost classifier with 10 boosting steps. The results of 75 experiments (each one repeated 100 times) are summed up in this graph.

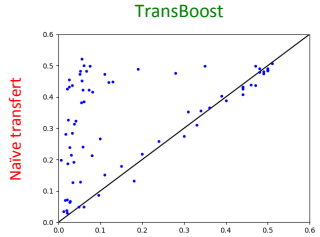


Figure 4: Comparison of error rates. y-axis: test error of the "naïve" transfer method. x-axis : test error of the TransBoost classifier with 10 boosting steps. The results of 75 experiments (each one repeated 100 times) are summed up in this graph.

Results

- Hypothèse source a priori **sans lien** avec la tâche cible !!??

Learning from target data only | TransBoost with any source hypothesis

slope, noise, $t_\mathcal{T}$	$h_\mathcal{T}$ (train)	$h_\mathcal{T}$ (test)	$H_\mathcal{T}$ (train)	$H_\mathcal{T}$ (test)
0.001, 0.001, 70	0.44 ± 0.02	0.48 ± 0.02	0.06 ± 0.02	$\mathbf{0.06} \pm 0.02$
0.005, 0.005, 70	0.11 ± 0.04	0.13 ± 0.05	0.02 ± 0.01	$\mathbf{0.02} \pm 0.02$
0.005, 0.005, 70	0.10 ± 0.04	0.11 ± 0.05	0.01 ± 0.01	$\mathbf{0.01} \pm 0.01$
0.005, 0.05, 70	0.11 ± 0.04	0.12 ± 0.05	0.04 ± 0.02	$\mathbf{0.03} \pm 0.01$
0.001, 0.001, 70	0.42 ± 0.03	0.48 ± 0.02	0.33 ± 0.02	$\mathbf{0.37} \pm 0.02$
0.01, 0.1, 70	0.06 ± 0.03	0.08 ± 0.03	0.08 ± 0.02	0.08 ± 0.02

Table 2: Learning without transfer and with transfer using an apriori irrelevant source hypothesis.

Résultats

Effet du nombre de projections (étapes de boosting)

- $T_\mathcal{T}$ = **20**

Table 1: TransBoost test error for time series of length 20

dataset	SVM	5-TransBoost	10-TransBoost	20-TransBoost
1	0.03	0.06	0.06	0.06
2	0.49	0.5	0.5	0.5
3	0.23	0.02	0.02	0.02
4	0.04	0.11	0.09	0.06
5	0.32	0.25	0.25	0.24
6	0.48	0.32	0.29	0.29
7	0.35	0.31	0.30	0.30
8	0.50	0.44	0.38	0.34

Résultats

Effet du nombre de projections (étapes de boosting)

- T_T = 50

Table 2: TransBoost test error for time series of length 50

dataset	SVM	5-TransBoost	10-TransBoost	20-TransBoost
1	0.08	0.02	0.02	0.02
2	0.08	0.06	0.05	0.04
3	0.02	0.06	0.05	0.03
4	0.12	0.16	0.13	0.13
5	0.08	0.02	0.02	0.02
6	0.02	0.06	0.04	0.04
7	0.45	0.45	0.45	0.45
8	0.12	0.12	0.12	0.12

Plan

1. **Qu'est-ce que le transfert ?**
   - **Questions** qui se posent
   - Un peu de **formalisation**
   - Travaux **antérieurs**
2. La **classification précoce** de séries temporelles
3. **L'algorithme** proposé
   - Analyse **théorique**
   - **Expériences**
4. Conclusions : **comprendre**

Conclusions … et suites

Conclusions

- Le transfert est un type d'**apprentissage intéressant**
  - Apprentissage au long cours
  - « Curriculum learning »
  - Nouvelles questions sur les conditions de l'induction

- **Nouvelle perspective** sur l'apprentissage par transfert
  - **Boosting de projections** (pas recherche de points communs)
  - Change le point de vue sur les **conditions d'un bon apprentissage par transfert**
    - Ensemble de projections **adéquat**
    - Et de **faible capacité** Pas une bonne source !!

Conclusions

- Un **algorithme original**
  - **Simple**
  - **Un** seul paramètre
  - Hérite des **bonnes propriétés** du boosting
    - Contrôle de l'**erreur en apprentissage**
    - Contrôle de l'**erreur en test**
  - Le problème de l'apprentissage est maintenant déplacé vers le choix d'un bon espace de projections

Mais … !?

=> *Pas de condition sur la source !??*

Pourtant des problèmes d'apprentissage par transfert nous paraissent **plus faciles que d'autres** ???

Transfert facile vs. pas facile

- Facile
  - Quand il est **facile de trouver** une ensemble de projections
    - Avec la **propriété faible de transfert**
    - De **faible capacité**
- Sinon : Pas facile

E.g. classification précoce de séries temporelles en utilisant une hypothèse source quelconque

Interprétation

- Le transfert agit comme un terme de **régularisation**
- On **oblige les hypothèses cibles** à être de la forme

$$h_{\mathcal{S}} \circ \pi \qquad \text{avec} \qquad \pi : \mathcal{X}_{\mathcal{T}} \rightarrow \mathcal{X}_{\mathcal{S}}$$

  - Si l'**hypothèse cible** est **bien choisie** : ça aide
  - **Sinon** : **pas d'apprentissage** possible

    ou **sur-apprentissage** si Π de trop grande capacité

sé

Which theoretical guarantees

for transfer learning?

Rappel

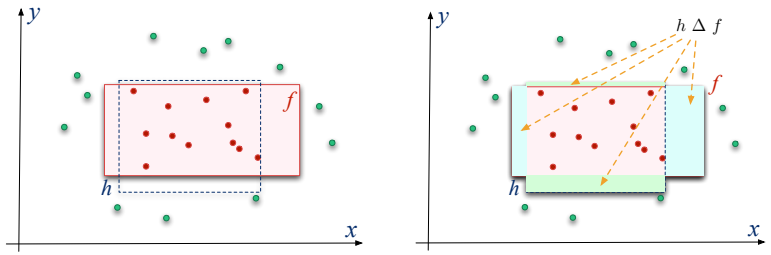
Théorie statistique de l'apprentissage

Le 1er temps

Un individu

Étude statistique pour UNE hypothèse

- choix d'une hypothèse de risque empirique nul (pas d'erreur sur l'échantillon d'apprentissage *S*)
- Quelle performance attendue pour *h* ?
- Quel est le risque d'avoir une erreur $R(h) > \varepsilon$?



Étude statistique pour UNE hypothèse

- Supposons *h* tq. $R(h) \geq \varepsilon$ (*h* « mauvaise »)
- Quelle est la probabilité que pourtant *h* ait été sélectionnée ?

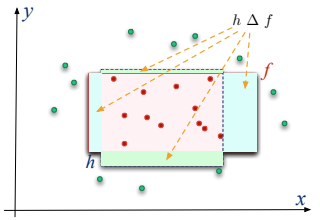
$$R(h) = \mathbf{p}_{\mathcal{X}}(h \Delta f)$$

Après un exemple : $p(\hat{R}(h) = 0) \leq 1 - \varepsilon$

« tombe » en dehors de $h \Delta f$

Après *m* exemple (**i.i.d.**) :

$$p^m(\hat{R}(h) = 0) \leq (1 - \varepsilon)^m$$



On veut : $\forall\, \varepsilon, \delta \in [0,1] : \quad p^m(R(h) \geq \varepsilon) \leq \delta$

Étude statistique pour UNE hypothèse

- On cherche : $\forall\, \varepsilon, \delta \in [0,1] : \quad p^m\big(R(h) \geq \varepsilon\big) \leq \delta$

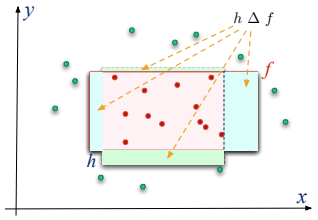
Soit :

$$(1-\varepsilon)^m \leq \delta$$

$$e^{-\varepsilon m} \leq \delta$$

$$-\varepsilon m \leq \ln(\delta)$$

D'où : $$m \geq \frac{\ln(1/\delta)}{\varepsilon}$$



Théorie statistique de l'apprentissage

Le 2ème temps

Quel individu dans la Foule

Étude statistique pour $|\mathcal{H}|$ hypothèses

- Quelle est la probabilité que je choisisse une hypothèse h_{err} de risque réel > ε **et que je ne m'en aperçoive pas** après l'observation de *m* exemples ?
- Probabilité de survie de h_{err} **après 1 exemple** : $(1-\varepsilon)$
- Probabilité de survie de h_{err} **après *m* exemples** : $(1-\varepsilon)^m$
- Probabilité de survie d'**au moins une hypothèse dans $\mathcal{H}$** : $|\mathcal{H}|\,(1-\varepsilon)^m$
  - On utilise la probabilité de l'union

$$\mathbf{P}(A \cup B) \leq \mathbf{P}(A) + \mathbf{P}(B)$$

- On veut que la probabilité qu'il reste au moins une hypothèse de risque réel > ε dans l'espace des versions soit bornée par δ :

$$|\mathcal{H}|\,(1-\varepsilon)^m < |\mathcal{H}|e^{(-\varepsilon m)} < \delta$$

$$\log|\mathcal{H}| - \varepsilon m < \log\delta$$

$$m > \frac{1}{\varepsilon}\log\frac{|\mathcal{H}|}{\delta}$$

L'analyse « PAC learning »

- On arrive à :

$$\forall h \in \mathcal{H}, \forall \delta \leq 1 : \quad \mathbf{P}^m\left[R_{\text{Réel}}(h) \leq R_{\text{Emp}}(h) + \overbrace{\frac{\log|\mathcal{H}| + \log\frac{1}{\delta}}{m}}^{\varepsilon}\right] > 1-\delta$$

$R_{\text{Réel}}(h) - R_{\text{Emp}}(h) = 0$

Cas réalisable : il existe au moins une fonction *h* de risque 0

Le principe de minimisation du risque empirique
n'est sain que si il y a des contraintes sur l'espace des hypothèses

Bounding the true risk with the empirical risk + …

- $\mathcal{H}$ finite, realizable case

$$\forall h \in \mathcal{H}, \forall \delta \leq 1 : \quad P^m \left[R_{\text{Réel}}(h) \leq R_{\text{Emp}}(h) + \frac{\log |\mathcal{H}| + \log \frac{1}{\delta}}{m} \right] > 1 - \delta$$

- $\mathcal{H}$ finite, **non** realizable case

$$\forall h \in \mathcal{H}, \forall \delta \leq 1 : \quad P^m \left[R_{\text{Réel}}(h) \leq R_{\text{Emp}}(h) + \sqrt{\frac{\log |\mathcal{H}| + \log \frac{1}{\delta}}{2\,m}} \right] > 1 - \delta$$

$|\mathcal{H}|$ infinite !!

- Effective dimension of $\mathcal{H}$ = *the **Vapnik-Chervonenkis** dimension*
  - **Combinatorial criterion**
  - Size of the largest set of points (in general configuration) that can be labeled in any way by hypotheses drawn from $\mathcal{H}$

$$d_{VC}(\mathcal{H}) = \max\{m : \Pi_{\mathcal{H}}(m) = 2^m\}$$

Bound on the true risk

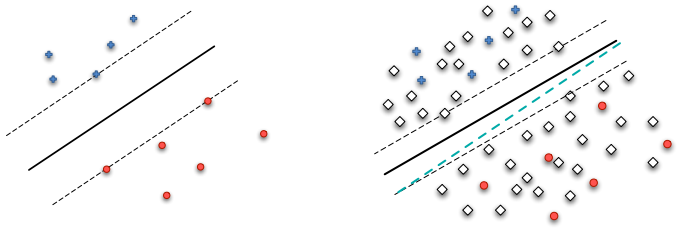
$$\forall h \in \mathcal{H}, \forall \delta \leq 1 : \quad P^m \left[R_{\text{Réel}}(h) \leq R_{\text{Emp}}(h) + \sqrt{\frac{8\, d_{VC}(\mathcal{H}) \log \frac{2\,e\,m}{d_{VC}(\mathcal{H})} + 8 \log \frac{4}{\delta}}{m}} \right] > 1 - \delta$$

A détour:

Looking at a theory of semi-supervised learning

Semi supervised learning

- General principle
  - The decision function does not cut through high density regions of *X*
    - P_X is related to $P_{Y|X}$
  - The S3VM algorithm



Semi supervised learning: how to approach it theoretically?

The **ability of unlabeled data to help** depends on two quantities:

1. The extent to which
   the target function indeed **satisfies the given assumptions**

2. The extent to which
   the distribution allows this assumption to **rule out alternative hypotheses**

How to derive guarantees for semi-supervised learning?

[Balcan & Blum (2006). *"An augmented PAC model for semi-supervised learning"*]

- Let's assume that **it is reasonable that the frontier between two classes does not cut through high density regions of the input space** *X*
  - Then the unlabeled data points bring constraints on the possible decision functions -> gain of information
- Formally: let's define a compatibility function $\chi : \mathcal{H} \times X \to [0,1]$
  - E.g. $\chi(h,x)$ could be an increasing function of the distance of x to the decision function (separator) h

$$\chi(h,\mathcal{D}) = \mathbb{E}_{\mathbf{x}\sim\mathcal{D}}[\chi(h,\mathbf{x})]$$ Compatibility between *h* and *D*

$$\chi(h,S) = \frac{1}{m}\sum_{i=1}^{m}\chi(h,\mathbf{x}_i)$$ Empirical compatibility measured on *S*

How to derive guarantees for semi-supervised learning?

- Incompatibility $er_{\text{unl}}(h) = 1 - \chi(h,\mathcal{D})$

  $\widehat{err}_{\text{unl}}(h) = 1 - \chi(h,S)$

- Let's define the **set of hypotheses whose incompatibility is** at most some given value τ

$$\mathcal{H}_{\mathcal{D},\mathcal{X}}(\tau) = \{h \in \mathcal{H} : err_{\text{un}}(h) \leq \tau\}$$

$$\mathcal{H}_{S,\mathcal{X}}(\tau) = \{h \in \mathcal{H} : \widehat{err}_{\text{un}}(h) \leq \tau\}$$

How to derive guarantees for semi-supervised learning?

- Theorem (realizable case and $\mathcal{H}$ finite)

If we see m_u unlabeled examples and m_l labeled examples, where

$$m_u \geq \frac{1}{\varepsilon}\left[\ln|\mathcal{H}| + \ln\frac{2}{\delta}\right] \quad \text{and} \quad m_l \geq \frac{1}{\varepsilon}\left[\ln|\mathcal{H}_{\mathcal{D},\mathcal{X}}(\varepsilon)| + \ln\frac{2}{\delta}\right]$$

then, with probability $\geq 1-\delta$, any $h \in \mathcal{H}$ with $\widehat{err}(h) = 0$ and $\widehat{err}_{\text{unl}}(h) = 0$ has $err(h) \leq \varepsilon$

How to derive guarantees for semi-supervised learning?

- Proof:

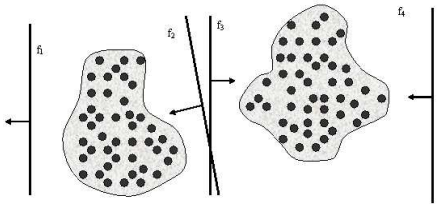
The probability that a given hypothesis h with $err_{\text{unl}}(h) > \varepsilon$ has $\widehat{err}_{\text{unl}}(h) = 0$ is at most $(1-\varepsilon)^{m_u} < \dfrac{\delta}{2|\mathcal{H}|}$ for the given value of m_u.

Therefore, by the union bound, the number of unlabeled examples is sufficient to ensure that, with probability $1 - \delta/2$, only hypotheses in $\mathcal{H}_{\mathcal{D},\mathcal{X}}(\varepsilon)$ have $\widehat{err}_{\text{unl}}(h) = 0$.

Similarly, the number of labeled examples ensures that with probability $1 - \delta/2$, none of those hypotheses whose true error is $\geq \varepsilon$ have an empirical error of 0, yielding the theorem.

How to derive guarantees for semi-supervised learning?

Note: Notice that for the setting of Example 1, in the worst case (over distributions D) this will essentially recover the standard margin sample-complexity bounds for the number of labeled examples. In particular, $C_{S,\chi}(0)$ contains only those separators that split S with margin $\geq \gamma$, and therefore, $s = \left|C_{S,\chi}(0)[2m_l, \overline{S}]\right|$ is no greater than the maximum number of ways of splitting $2m_l$ points with margin γ. However, if the distribution is helpful, then the bounds can be much better because there may be many fewer ways of splitting S with margin γ. For instance, in the case of two well-separated "blobs" illustrated in Figure 2.1, if S is large enough, we would have just $s = 4$.



How to derive guarantees for semi-supervised learning?

- The theorem **assumes**
  - The data is **i.i.d.**
    - Probability of each hypothesis to obey the criteria and still be in error
    - Union bound
  - The true target functions **obey the compatibility criterion**

What if Nature does not obey these assumptions?

E.g. the interesting decision functions cut through high density regions of *X*

What kind of theory for transfer learning?

Multi-task learning

Assumption behind MTL

- The **combined learning** of multiple related tasks **can outperform learning each task in isolation**
- MTL allows for **common information** shared between the tasks to be used in the learning process, which leads to better generalization if the tasks are related
- E.g. Learning to **predict the ratings** for several different critics (in different countries) can lead to better performances for **each separate task** (predict the restaurant ratings for a specific critic)
- Learning to **recognize a face** and the **expression** (fear, disgust, anger, …)
- **Multi modality learning**: e.g. vision and proprioception

Possible relations between tasks

- All functions to be learn are **close** to each other **in some norm**
  - E.g. functions capturing preferences in users' modeling problems
- Tasks that share a **common underlying representation**
  - E.g. in *human vision*, all tasks use the **same set of features** learnt in the first stages of the visual system (e.g. local filters similar to wavelets)
  - Users may also *prefer* different types of things (e.g. books, movies, music) based on the **same set of features** or **score** functions

Question

How do we chose to **model the shared information** between the tasks?

- Some shared underlying constraint
  - E.g. a **low dimensional representation** shared across multiple related tasks
    - By way of a shared hidden layer in a neural network
    - By explicitly constraining the dimensionality of a shared representation

Approche par régularisation : apprentissage multi-tâches

- *T* tâches de classification binaire définies sur $\mathcal{X}$ x $\mathcal{Y}$

$$\mathcal{S} = \{\{(\mathbf{x}_{11}, y_{11}), (\mathbf{x}_{21}, y_{21}), \ldots, (\mathbf{x}_{m1}, y_{m1})\}, \ldots, \{(\mathbf{x}_{1T}, y_{1T}), (\mathbf{x}_{2T}, y_{2T}), \ldots, (\mathbf{x}_{mT}, y_{mT})\}\}$$

$h_j(\mathbf{x}) = \mathbf{w}_j \cdot \mathbf{x}$ Hypothèses linéaires

Partage entre tâches $\mathbf{w}_j = \mathbf{w}_0 + \mathbf{v}_j$

$$h_1^\star, \ldots, h_T^\star = \underset{\mathbf{w}_0, \mathbf{v}_j, \xi_{ij}}{\text{Argmin}} \left\{ \sum_{j=1}^{T} \sum_{i=1}^{m} \xi_{ij} + \frac{\lambda_1}{T} \sum_{j=1}^{T} ||\mathbf{v}_j||^2 + \lambda_2 ||\mathbf{w}_0||^2 \right\}$$

Multi-task feature learning

- Suppose that each regression function h_t is linear in feature functions f_t

$$h_t(\mathbf{x}) = \sum_{i=1}^{d} a_{it} f_i(\mathbf{x})$$

- The feature functions are supposed to be linear. And the $\boldsymbol{u}_i$ are supposed to be orthonormal

$$f_i(\mathbf{x}) = \langle \mathbf{u}_i, \mathbf{x} \rangle$$

$$\begin{aligned} h_t(\mathbf{x}) &= \langle \mathbf{w}_t, \mathbf{x} \rangle \\ &= \langle \sum_i a_{it} \mathbf{u}_i, \mathbf{x} \rangle \end{aligned}$$

A. Argyriou & T. Evgeniou (2006). "Multi-task feature learning". NIPS-2006.

Multi-task feature learning (2)

- Suppose **only one task** with **features $\boldsymbol{u}_i$ fixed** a priori
- Learning task: learn the parameter vector a_t in R^d from data set $\{(\mathbf{x}_{ti}, y_{ti})\}_{i=1,\ldots,m}$
- We want to bound the number of non zero components of $\boldsymbol{a}_t$

$$\underset{\mathbf{a}_t \in \mathbb{R}^d}{\text{ArgMin}} \left\{ \sum_{i=1}^{m} \ell(\underbrace{\langle \mathbf{a}_t, U^\top \mathbf{x}_{ti} \rangle}_{h_t(\mathbf{x}_{ti})}, y_{ti}) + \gamma ||\mathbf{a}_t||_1^2 \right\}$$

Multi-task feature learning (3)

- Suppose **multiple tasks** with **features $\boldsymbol{u}_i$ to be learned**
- Learning task: learn the parameter vector a_t in R^d and the features $\boldsymbol{u}_i$ from data set $\{(\mathbf{x}_{ti}, y_{ti})\}_{i=1,\ldots,m}^{t=1,\ldots,T}$
- We want to bound the number of non zero components of $\boldsymbol{a}_t$ and have **W** to be a low rank matrix

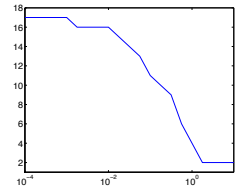
$$\underset{\mathbf{a}_t \in \mathbb{R}^{d \times T}, U \in \mathbf{O}^d}{\text{ArgMin}} \left\{ \sum_{t=1}^{T} \sum_{i=1}^{m} \ell(\underbrace{\langle \mathbf{a}_t, U^\top \mathbf{x}_{ti} \rangle}_{h_t(\mathbf{x}_{ti})}, y_{ti}) + \gamma ||\mathbf{A}||_{2,1}^2 \right\}$$

But this is a **non-convex problem**, and the norm ||A||2,1 is **non smooth**.

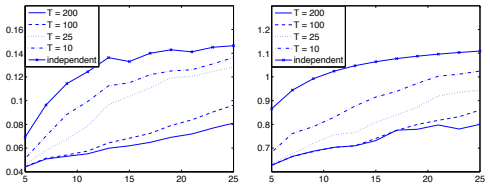
=> Alternate minimization of loss wrt. **A** and U, and the computation of the w_t

Multi-task feature learning (4)

- **Experiments** (synthetic data) 200 tasks
  - with w_t from a 5 dimensional Gaussian (mean and covariance)+ 20 irrelevant dimensions
  - Each task: 5 or 10 examples

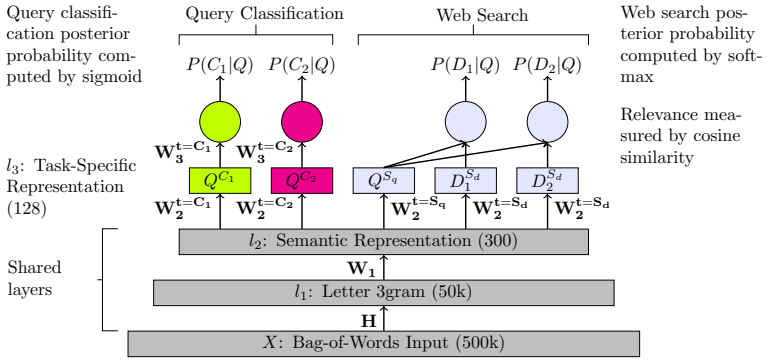


Number of features learned versus the regularization parameter γ



Test error (left) and residual of learned features (right) vs. dimensionality of the input

Multi-task learning with deep neural networks



[X. Liu, J. Gao, X.g He, L. Deng, K. Duh and Ye-Yi Wang (2015). « *Representation Learning Using Multi-Task Deep Neural Networks for Semantic Classification and Information Retrieval* ». Proc. NAACL, May 2015]

Multi-task learning with deep neural networks

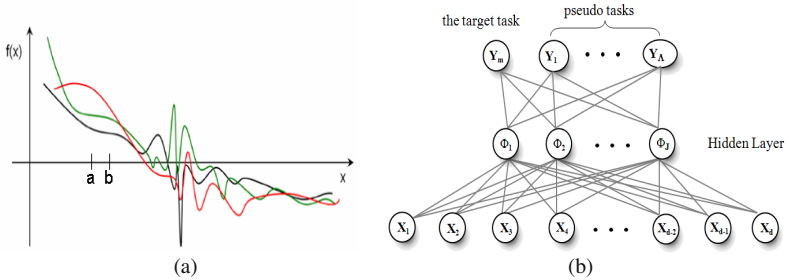
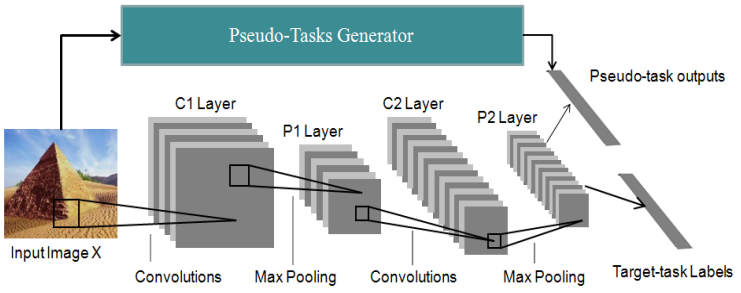


Fig. 1. Illustrating the mechanism of transfer learning. (a) Functional view: tasks represented as functional mapping share stochastic characteristics. (b) Transfer learning in neural networks, the hidden layer represents the level of sharing between all the task.

[A. Ahmed, K. Yu, W. XU, Y. Gong and E. Xing (2008). « *Training hierarchical feed-forward visual recognition models using transfer learning from pseudo-tasks*». Proc. ECCV-2008]

Multi-task learning with deep neural networks



[A. Ahmed, K. Yu, W. XU, Y. Gong and E. Xing (2008). « *Training hierarchical feed-forward visual recognition models using transfer learning from pseudo-tasks*». Proc. ECCV-2008]

Current approaches to transfer learning

Three main approaches

- **Reweighting / Instance-based methods**

  Correct a sample bias by reweighting source labeled data:
  source instances "close" to target instances are more important

- **Feature-based methods / Find new representation spaces**

  Find a common space where source and target are close
  (projection, new features, etc.)

- **Adjustment / Iterative methods**

  Modify the model by incorporating pseudo-labeled information

Reweighting methods

Co-variate shift

Environnement non stationnaire

- **Co-variate shift** $\mathbf{p}_{\mathcal{X}}$
  - Dérive virtuelle
  - Non i.i.d.

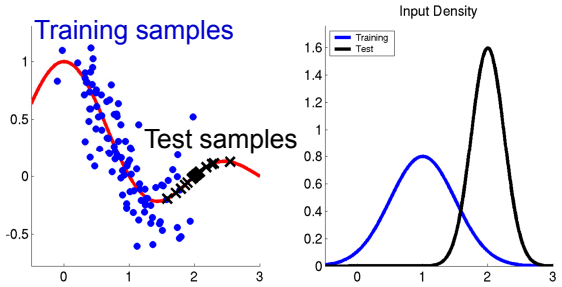
E.g. Robot se déplaçant

- **Changement de concept** $\mathbf{p}_{\mathcal{Y}|\mathcal{X}}$
  - *Concept drift*
  - Non i.i.d. + non stationnaire

E.g. goût des utilisateurs

Covariate shift

- Input distribution changes $P_{train}(\mathbf{x}) \neq P_{test}(\mathbf{x})$
- Functional relation remains unchanged $P_{train}(y|\mathbf{x}) = P_{test}(y|\mathbf{x})$



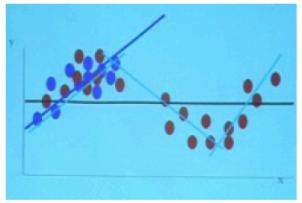
Co-variate shift

No longer a "direct" link between empirical risk and real risk

Modify the inductive criterion

The performance for the target distribution $\mathbf{P}'_{\mathcal{X}}$ (*generalization*) depends on :

- The performance for $\mathbf{P}_{\mathcal{X}}$ (*learning*)
- The similarity between $\mathbf{P}_{\mathcal{X}}$ and $\mathbf{P}'_{\mathcal{X}}$



Principle

$$\hat{h}_{\mathrm{ERM}}(\mathcal{S}) = \underset{h\in\mathcal{H}}{\mathrm{ArgMin}}\left\{\frac{1}{m}\sum_{i=1}^{m}\ell(h(\mathbf{x}_i), y_i)\right\}$$

- We know that ERM is **consistent**

$$R(\hat{h}_{\mathrm{ERM}}(\mathcal{S})) \xrightarrow[\text{converge}]{\mathbf{x}_i \overset{i.i.d.}{\sim} \mathbf{p}(\mathbf{x})} R(h^\star)$$

If the test distribution is the same as the train distribution $p_{train}(x)$

$$R(h) = \int_{\mathcal{X}\times\mathcal{Y}} \ell(h(\mathbf{x}), y)\ \mathbf{p}(\mathcal{X}, \mathcal{Y})\ d\mathbf{x}\, dy$$

First analysis

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

First analysis

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

First analysis

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

First analysis

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

First analysis

Covariate shift [Shimodaira,'00]

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

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

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

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

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

Principle

- Law of large numbers
  - Sample averages converge to the population mean

$$\frac{1}{n}\sum_{i=1}^{n} A(x_i) \xrightarrow[n\to\infty]{x_i \overset{i.i.d.}{\sim} \mathbf{p}_{train}(x)} \int A(x)\,\mathbf{p}_{train}(x)\,dx$$

$$\frac{1}{n}\sum_{i=1}^{n} \frac{\mathbf{p}_{test}(x)}{\mathbf{p}_{train}(x)} A(x_i) \xrightarrow[n\to\infty]{x_i \overset{i.i.d.}{\sim} \mathbf{p}_{train}(x)} \int \frac{\mathbf{p}_{test}(x)}{\mathbf{p}_{train}(x)} A(x)\,\mathbf{p}_{train}(x)\,dx$$

$$\xrightarrow[n\to\infty]{x_i \overset{i.i.d.}{\sim} \mathbf{p}_{train}(x)} \int A(x)\,\mathbf{p}_{test}(x)\,dx$$

  - But how to estimate $\dfrac{\mathbf{p}_{test}(x)}{\mathbf{p}_{train}(x)}$?

Importance weighting

- A naïve estimation of $\dfrac{\mathbf{p}_{test}(x)}{\mathbf{p}_{train}(x)}$ does not work
  - Estimation density is too crude in high dimension space (and with few known testing instances)

- Idea of Sugiyama:
  - **Learn a parametric model** of $w(\mathbf{x}) = \dfrac{\mathbf{p}_{test}(x)}{\mathbf{p}_{train}(x)}$

$$\hat{w}(\mathbf{x}) = \sum_{j=1}^{J} \theta_j \phi_j(\mathbf{x}) \quad \text{and} \quad \hat{\mathbf{p}}_{test}(\mathbf{x}) = \hat{w}(\mathbf{x})\,\mathbf{p}_{train}(\mathbf{x})$$

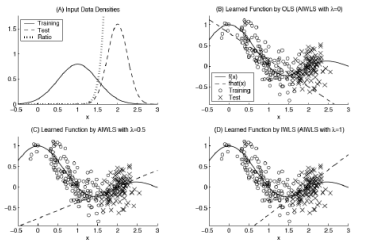
Covariate shift in regression

"Importance weighted" inductive criterion

Principle : weighting the classical ERM

$$R_{Cov}(h) = \frac{1}{m}\sum_{i=1}^{m} \left(\frac{\mathbf{P}_{\mathcal{X}'}(\boldsymbol{x}_i)}{\mathbf{P}_{\mathcal{X}}(\boldsymbol{x}_i)}\right)^{\lambda} (h(\boldsymbol{x}_i - y_i)^2$$

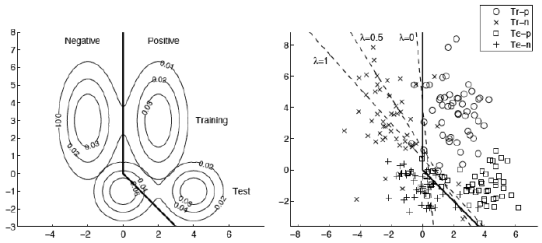
λ controls the stability / consistency (absence of bias)



SKM07 M. Sugiyama and M. Kraudelat and K.-R. Müller (2007) *"Covariate Shift Adaptation by Importance Weighted Cross Validation"* Journal of Machine Learning Research, vol.8: 985-1005.

Covariate shift in classification

"Importance weighted" inductive criterion (classification task)



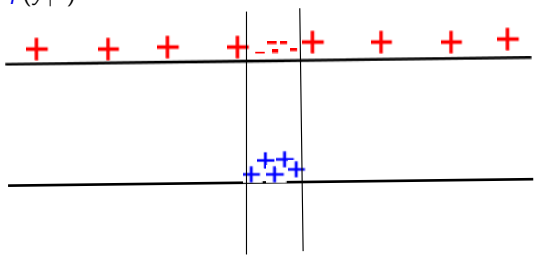
(a) Contours of training and test input densities.

(b) Optimal decision boundary (solid line) and learned boundaries (dashed lines). 'o' and '×' denote the positive and negative training samples, while '□' and '+' denote the positive and negative test samples. Note that the test samples are not given in the training phase; they are plotted in the figure for illustration purposes.

SKM07 M. Sugiyama and M. Kraudelat and K.-R. Müller (2007) *"Covariate Shift Adaptation by Importance Weighted Cross Validation"* Journal of Machine Learning Research, vol.8: 985-1005.

But

- 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 X
- Covariate shift assumption may fail: Tasks are not similar in general $P_S(y|\mathbf{x}) \neq P_T(y|\mathbf{x})$



- We did not consider the hypothesis space.
- Can define a general theory about DA?

Adjusting / Iterative methods

Principle

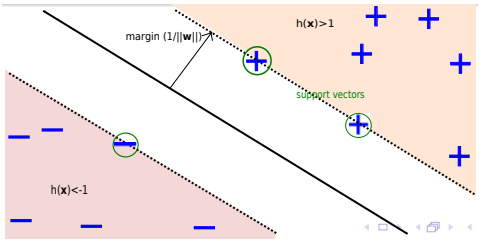
- Integrate some **information about the target** samples iteratively
  => use ***pseudo-labels***
- **Remove / add** some **instances** so as to *move the source distribution towards the target distribution*
- **Repeat** the process **until** convergence or no remaining instances

DASVM [Bruzone et al., 2010]

A brief recap on SVM

- Learning sample $LS = \{(\mathbf{x}_i, y_i)\}_{i=1}^n$
- Learn a classifier $h(\mathbf{x}) = \langle \mathbf{w}, \mathbf{x} \rangle + b$

$$\text{Formulation: } \min_{\mathbf{w},b,\boldsymbol{\xi}} \quad \tfrac{1}{2}\|\mathbf{w}\|_2^2 + \ C \sum_{i=1}^n \xi_i$$
$$\text{subject to} \quad \ell_i(\langle \mathbf{w}, \mathbf{x}_i \rangle + b) \geq 1 - \ \xi_i, \quad 1 \leq i \leq n$$
$$\boldsymbol{\xi} \succeq 0$$

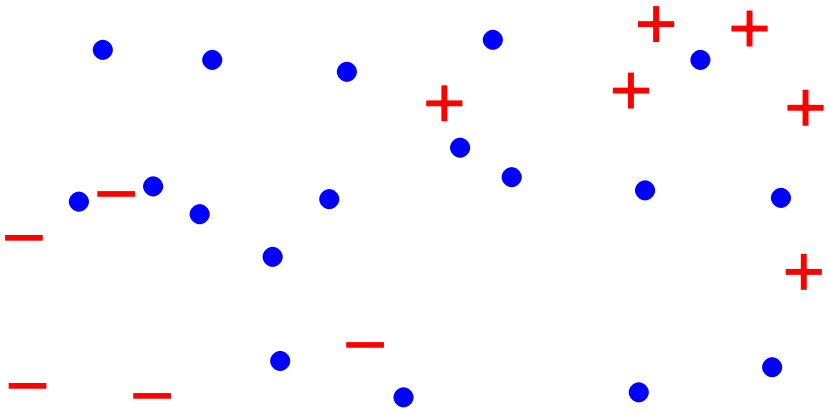


DASVM: algorithm

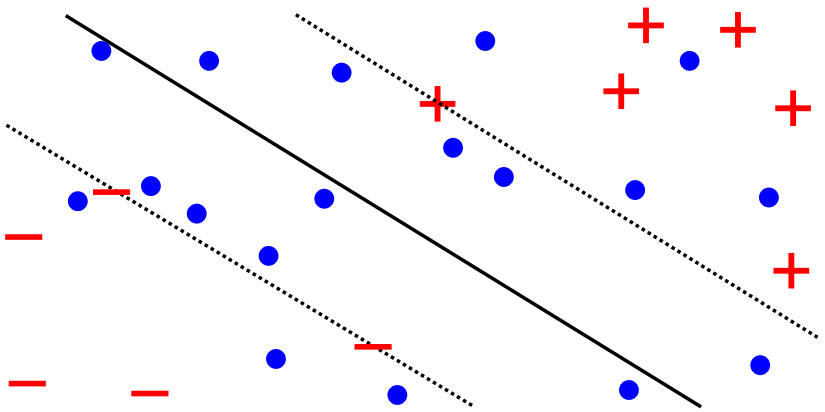
1. *LS* = *S*
2. Learn a classifier h^0 from the learning sample *LS*
3. **Repeat until** stopping criterion
   - **Select** the first *k* **target examples** $\boldsymbol{x}^t$ s.t. $0 < h(\boldsymbol{x}^t) < 1$ with highest margin (inside margin) and affect the pseudo-label -1
   - **Select** the first *k* **target examples** $\boldsymbol{x}^t$ s.t. $-1 < h(\boldsymbol{x}^t) < 0$ with highest margin (inside margin) and affect them the pseudo-label +1
   - **Add** these 2*k* examples (pseudo-labeled) to ***LS***
   - **Remove** from ***LS*** the first *k* positive and *k* negative source instances with highest margin
4. **Output** the last classifier

Algorithm stops when the number of selected instances at each step falls down below a threshold.

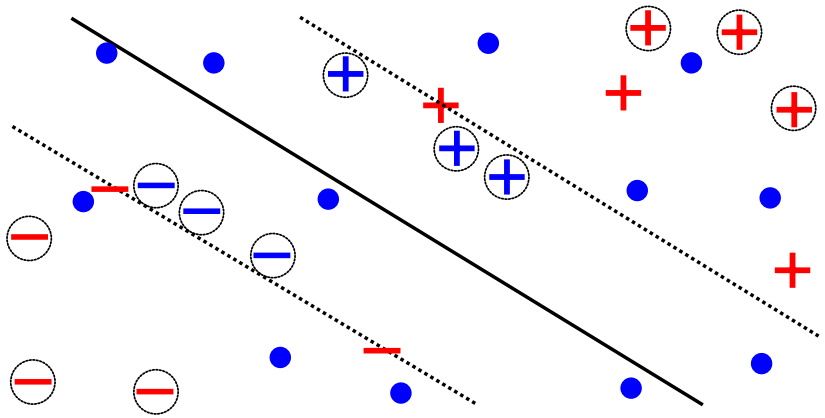
DASVM: illustration



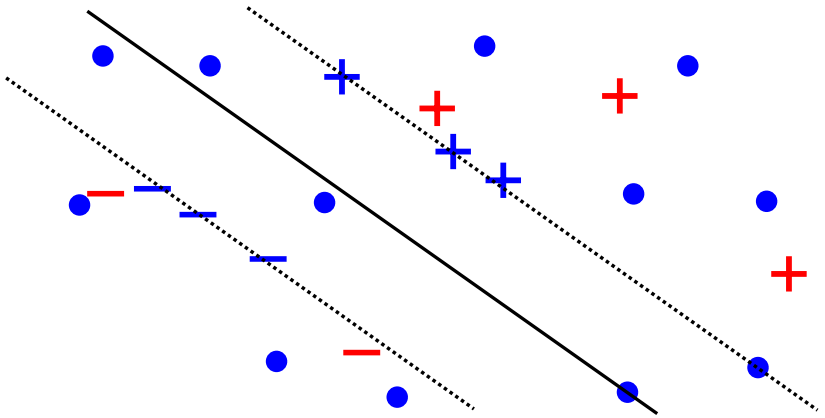
DASVM: illustration



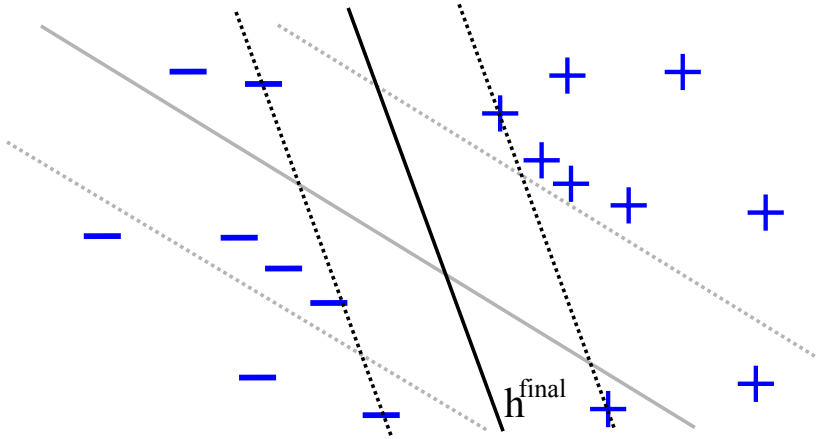
DASVM: illustration



DASVM: illustration



DASVM: illustration



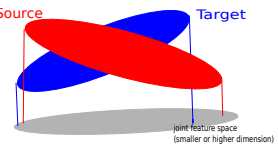
DASVM

- There are **theoretical studies**
  - Based on the notion of weak learners
  - In order to determine the conditions and guarantees of DASVM
- There are **applications**
  - E.g. in the domain of character recognition

Feature / Projection based approaches

Idea

- **Change the feature representation** X to better represent shared characteristics between the two domains
  - some features are domain-specic,
  - others are generalizable
  - or there exist mappings from the original space



=> Make source and target domain explicitly **similar**

=> Learn a **new feature space** by embedding or projection

Illustration: Find latent spaces – Structural Correspondence Learning [Blitzer et al., 2007]

Identify shared features

Domains	Negative	Positive
Books	plot <num>_pages predictable reading_this page_<num>	reader grisham engaging must_read fascinating
Kitchen	the_plastic poorly_designed leaking awkward_to defective	excellent_product espresso are_perfect years_now a_breeze
Pivot features	weak don't_waste awful	and_easy loved_it a_wonderful a_must highly_recommended

- Sentiment analysis - Bag of words (bigrams)
- Choose K pivot features (frequent words in both domains, highly correlated with labels)
- Learn K classifiers to predict pivot features from remaining features
- For each feature add K new features
- Represents source and target data with these features

Illustration: Find latent spaces – Structural Correspondence Learning [Blitzer et al., 2007]

- Apply PCA source+target new features to get a low rank latent representation
- Learn a classifier in the new projection space defined by PCA

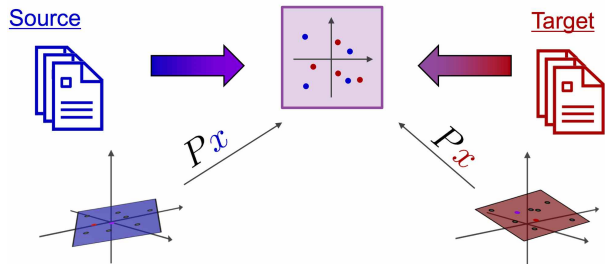
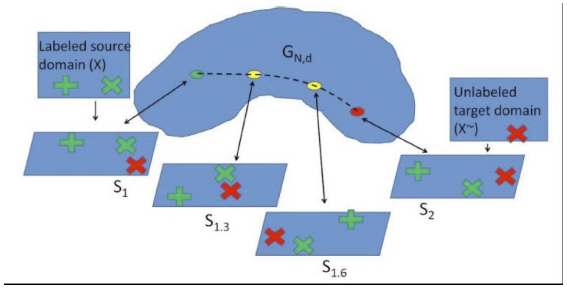
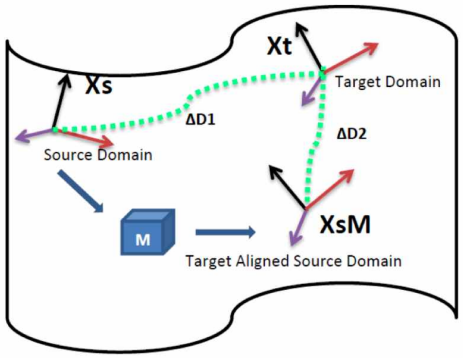


Illustration: Manifold-based methods



- Assume $X \subseteq \mathbb{R}^N$
- Apply PCA on source data $\Rightarrow$ matrix $\mathbf{S}_1$ of rank d
- Apply PCA on target data $\Rightarrow$ matrix $\mathbf{S}_2$ of rank d
- Geodesic path on the Grassman manifold $\mathbb{G}_{N,d}$ (d-dimensional vector subspaces $\subset \mathbb{R}^N$) between $\mathbf{S}_1$ and $\mathbf{S}_2$

A simpler approach – **Subspace alignment** [Fernando et al., ICCV-03]



- Move closer to PCA-based representations
- 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": $\mathbf{M}^* = \text{argmin}_{\mathbf{M}} \|\mathbf{S}_1\mathbf{M} - \mathbf{S}_2\|$
- $X_a = \mathbf{S}_1\mathbf{S}_1'\mathbf{S}_2 = \mathbf{S}_1\mathbf{M}^*$ projects the source data to the target subspace
- A natural similarity: $\text{Sim}(\mathbf{x}_s, \mathbf{x}_t) = \mathbf{x}_s\mathbf{S}_1\mathbf{M}^*\mathbf{S}_1'\mathbf{x}_t' = \mathbf{x}_s\mathbf{A}\mathbf{x}_t'$

Feature-based methods

- … are very popular
- Hot topic right now
- One central question:
  - Define a similarity map

References

References on Domain Adaptation and Transfer

- List of transfer learning papers
  http://www1.i2r.a-star.edu.sg/~jspan/conferenceTL.html
- List of available softwares
  http://www.cse.ust.hk/TL/index.html
- Surveys
  - Patel, Gopalan, Chellappa. Visual Domain Adaptation: An Overview ofRecent Advances. Tech report, 2014.
  - Qi Li. Literature Survey: Domain Adaptation Algorithms for Natural Language Processing, Tech report, 2012
  - Margolis. A Literature Review of Domain Adaptation with Unlabeled Data. Tech report 2011.
  - Pan and Yang. A survey on Transfer Learning', TKDE 2010.
- J. Quionero-Candela and M. Sugiyama and A. Schwaighofer and N.D. Lawrence (Eds)
  Dataset Shift in Machine Learning
  MIT Press, 2009

Additional References

- S. Ben-David
  Towards theoretical understanding of domain adaptation learning
  Workshop LNIID at ECML-09
- J. Blitzer and H Daumé III
  Domain Adaptation
  Tutorial ICML 2010
- K. Graumann
  Adaptation for objects and attributes
  Workshop VisDA at ICCV'13
- A. Habrard, J-P Peyrache and M. Sebban
  Iterative self-labeling Domain Adaptation for Linear Structured Image Classification
  IJAIT-2013
- A. Habrard, J-P Peyrache and M. Sebban
  Boosting for unsupervised domain adaptation
  ECML-2013
- A. Habrard
  An Introduction to Transfer Learning and Domain Adaptation
  Ecole d'été EPAT-2014
- S. Pan, Q. Yang and W. Fan
  Tutorial: Transfer Learning with Applications
  IJCAI'13
- F. Sha and B. Kingsbury
  Domain Adaptation in Machine Learning and Speech Recognition
  Tutorial – Interspeech 2012
- D. Xu, K. Saenko and I. Tang
  Tutorial on Domain Transfer Learning for Vision Applications
  CVPR'12

Lessons

Lessons

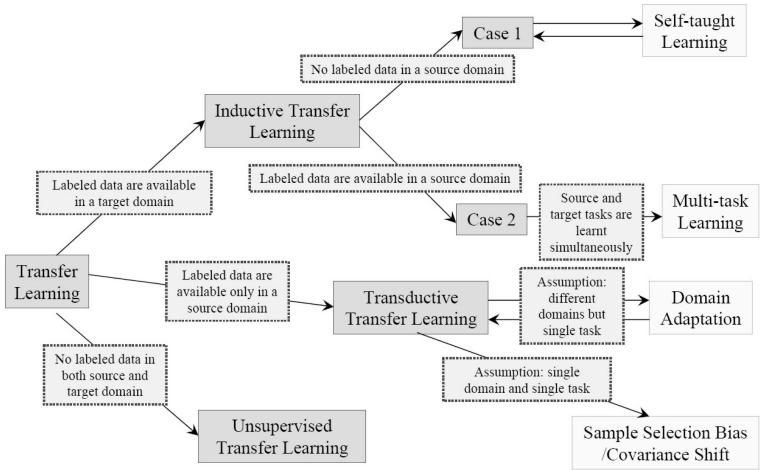
- The **big questions**
  1. **What** to transfer?
  2. **How** to transfer?
  3. How to decide that transfer learning should be profitable?
     How to obtain **theoretical guarantees**?
- What should **intervene** in the **guarantees**?
  1. "**distance**" between source and target
  2. Size of the **target** training **data**
  3. (performance of the source hypothesis?)

Lessons

- The approaches
  1. Hypothesis Transfer Learning (HTL)
  2. Change the distributions such that they become indistinguishable

Still a **very open** scientific question

A taxonomy



From "A survey on Transfer Learning" [Pan & Yang, TKDE, 2010]

Feature / Projection based **approaches**

A first attempt at a theoretical framework

for **domain adaptation**

Shai Ben-David and colleagues

The $\mathcal{H}\Delta\mathcal{H}$ divergence [Ben-David et al., NIPS-2006, MLj'10]

Definition

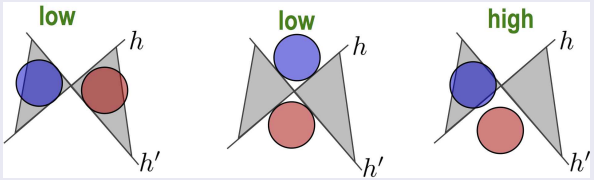
$$d_{\mathcal{H}\Delta\mathcal{H}}(D_S, D_T) = \sup_{(h,h')\in\mathcal{H}^2} \left| R_{D_T}(h,h') - R_{D_S}(h,h') \right|$$
$$= \sup_{(h,h')\in\mathcal{H}^2} \left| \mathop{\mathbf{E}}_{\mathbf{x}^t\sim D_T} \mathbf{I}\left[h(\mathbf{x}^t) \neq h'(\mathbf{x}^t)\right] - \mathop{\mathbf{E}}_{\mathbf{x}^s\sim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

The $\mathcal{H}\Delta\mathcal{H}$ divergence [Ben-David et al., NIPS-2006, MLj'10]

Definition

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

Illustration with only 2 hypothesis in $\mathcal{H}$ h and h'



Note: With a larger $\mathcal{H}$, the distance will be **high** since we can easily find two hypothesis able to **distinguish** the two domains

The $\mathcal{H}\Delta\mathcal{H}$ divergence [Ben-David et al., NIPS-2006, MLj'10]

Consider two samples S, T of size m from D_S and D_T

$$d_{\mathcal{H}\Delta\mathcal{H}}(D_S, D_T) \leq d_{\mathcal{H}\Delta\mathcal{H}}(S,T) + O(\text{complexity}(\mathcal{H})\sqrt{\tfrac{\log(m)}{m}})$$

complexity($\mathcal{H}$): VC-dimension [Ben-david et al.,06;'10], Rademacher [Mansour et al.,'09]

Empirical estimation

$$\hat{d}_{\mathcal{H}\Delta\mathcal{H}}(S,T) = 2\left(1 - min_{h\in\mathcal{H}}\left[\frac{1}{m}\sum_{\mathbf{x}:h(\mathbf{x})=-1} I[\mathbf{x}\in S] + \frac{1}{m}\sum_{\mathbf{x}:h(\mathbf{x})=1} I[\mathbf{x}\in T]\right]\right)$$

$\Rightarrow$ Already seen: label source examples as -1, target ones as +1 and try to learn a classifier in $\mathcal{H}$ minimizing the associated empirical error

Going to a generalization bound

Preliminaries

- $R_{P_T}(h, h') = \mathop{\mathbf{E}}_{(\mathbf{x},y)\sim P_S} I[h(\mathbf{x}) \neq h'(\mathbf{x})] = \mathop{\mathbf{E}}_{\mathbf{x}\sim D_T} I[h(\mathbf{x}) \neq h'(\mathbf{x})]$
  R_{P_T} (R_{P_S}) fulfills the triangle inequality
- $|R_{P_T}(h, h') - R_{P_S}(h, h')| \leq \frac{1}{2} d_{\mathcal{H}\Delta\mathcal{H}}(D_S, D_T)$
  since $d_{\mathcal{H}\Delta\mathcal{H}}(D_S, D_T) = 2 \sup_{(h,h')\in\mathcal{H}^2} \left| R_{D_T}(h, h') - R_{D_S}(h, h') \right|$
- $h^*_S = \text{argmin}_{h\in\mathcal{H}} R_{P_S}(h)$: best on source
- $h^*_T = \text{argmin}_{h\in\mathcal{H}} R_{P_T}(h)$: best on target

Ideal joint hypothesis

$h^* = \text{argmin}_{h\in\mathcal{H}} R_{P_S}(h) + R_{P_T}(h)$; $\lambda = R_{P_S}(h^*) + R_{P_T}(h^*)$

Going to a generalization bound

$$
\begin{aligned}
R_{P_T}(h) &\leq R_{P_T}(h^*) + R_{P_T}(h, h^*) \\
&\leq R_{P_T}(h^*) + R_{P_S}(h, h^*) + R_{P_T}(h, h^*) - R_{P_S}(h, h^*) \\
&\leq R_{P_T}(h^*) + R_{P_S}(h, h^*) + |R_{P_T}(h, h^*) - R_{P_S}(h, h^*)| \\
&\leq R_{P_T}(h^*) + R_{P_S}(h, h^*) + \frac{1}{2} d_{\mathcal{H}\Delta\mathcal{H}}(D_S, D_T) \\
&\leq R_{P_T}(h^*) + R_{P_S}(h) + R_{P_S}(h^*) + \frac{1}{2} d_{\mathcal{H}\Delta\mathcal{H}}(D_S, D_T) \\
&\leq R_{P_S}(h) + \frac{1}{2} d_{\mathcal{H}\Delta\mathcal{H}}(D_S, D_T) + \lambda \\
&\leq R_S(h) + \frac{1}{2} d_{\mathcal{H}\Delta\mathcal{H}}(S, T) + O(\text{complexity}(\mathcal{H}) \sqrt{\frac{\log(m)}{m}}) + \lambda
\end{aligned}
$$

Main theoretical bound

Theorem [Ben-David et al.,MLJ'10,NIPS'06]

Let $\mathcal{H}$ a symmetric hypothesis space. If D_S and D_T are respectively the marginal distributions of source and target instances, then for all $\delta \in (0, 1]$, with probability at least $1 - \delta$:

$$\forall h \in \mathcal{H}, \quad R_{P_T}(h) \leq R_{P_S}(h) + \frac{1}{2} d_{\mathcal{H}\Delta\mathcal{H}}(D_S, D_T) + \lambda$$

Formalizes a natural approach: Move closer the two distributions while ensuring a low error on the source domain.
Justifies many algorithms:

- reweighting methods,
- feature-based methods,
- adjusting/iterative methods.

L'adaptation de domaine

- Une mise en théorie pionnière [Ben-David et al., 2010]

Théorème classique [Ben-David et al., 2010, Mansour et al., 2009a]
Soit $\mathcal{H}$ un espace d'hypothèses. Si D_S et D_T sont deux distributions sur X, alors :

$$\forall h \in \mathcal{H}, \quad \overbrace{R_{P_T}(h)}^{\text{erreur cible}} \leq \underbrace{R_{P_S}(h)}_{\text{erreur source}} + \underbrace{\tfrac{1}{2} d_{\mathcal{H}}(D_S, D_T)}_{\text{divergences}} + \nu$$

$R_{P_S}(h)$: erreur classique sur le domaine source
Minimisable via une méthode de classification supervisée sans adaptation

$\frac{1}{2} d_{\mathcal{H}}(D_S, D_T)$: la $\mathcal{H}$-divergence entre D_S et D_T

$$\frac{1}{2} d_{\mathcal{H}}(D_S, D_T) = \sup_{(h,h')\in\mathcal{H}^2} \left| R_{D_T}(h, h') - R_{D_S}(h, h') \right| = \sup_{(h,h')\in\mathcal{H}^2} \left| \mathop{\mathbf{E}}_{\mathbf{x}^t\sim D_T} \mathbf{I}\left[h(\mathbf{x}^t) \neq h'(\mathbf{x}^t)\right] - \mathop{\mathbf{E}}_{\mathbf{x}^s\sim D_S} \mathbf{I}\left[h(\mathbf{x}^s) \neq h'(\mathbf{x}^s)\right] \right|$$

ν : divergence entre les étiquetages

$\nu = \inf_{h'\in\mathcal{H}} (R_{P_S}(h') + R_{P_T}(h'))$,
erreur jointe optimale [Ben-David et al., 2010]

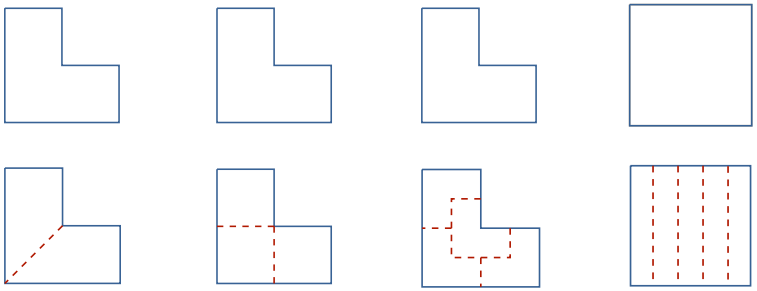
ou $\nu = R_{P_T}(h^*_T) + R_{P_T}(h^*_T, h^*_S)$,
h^*_X est la meilleure hypothèse sur le domaine X [Mansour et al., 2009a]

Idée : **construire** un nouvel espace de projection dans laquelle les deux distributions sont proches, tout en gardant une bonne performance sur le domaine source

Effets de séquences

- *Consigne* : découper la figure suivante en *n* parties superposables

En 2 : En 3 : En 4 : En 5 :



Analogy, transfer and ***sequence effects***

[Cornuéjols & Murena, 2016]