# Transfer Learning

## Covariant Learning and Parallel Transport

Antoine Cornuéjols

***AgroParisTech* – INRAe** MIA Paris-Saclay

EKINOCS research group

Induction is about using information

**from** some source data

**to** expected queries

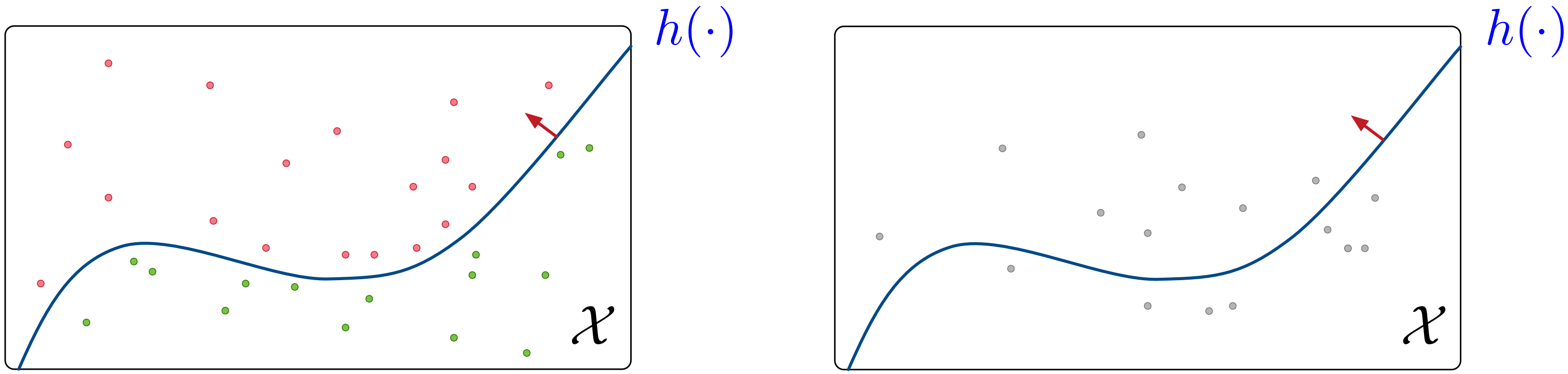
1. Which link between the **source** and the **target** are we ready to assume?

2. What kind of guarantees can we look for?

# Outline

1. Supervised induction: the classical setting
2. What about Out Of Distribution learning (OOD)?
3. Parallel transport, covariant derivative and transfer learning
   - What they are
   - ... in Machine Learning
4. A way to deal with different spaces of tasks
5. Conclusions

# Supervised induction



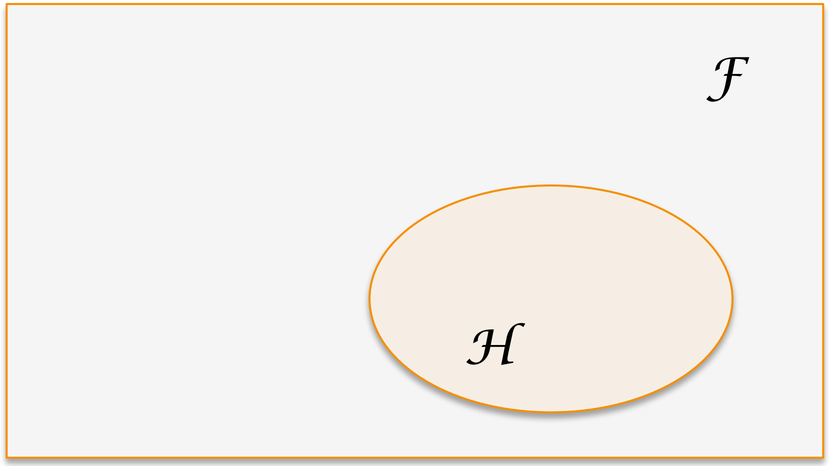

- Same distribution for **training** and **testing**
- Assumption: Empirical Risk Minimization is the way
  - a good hypothesis for the **training data** should be good as well for the **testing data**

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

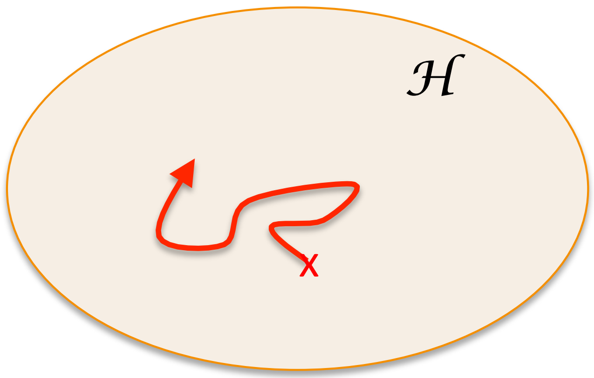
# Supervised induction: **guarantees**

- For this to hold, you need **prior assumptions**: biases

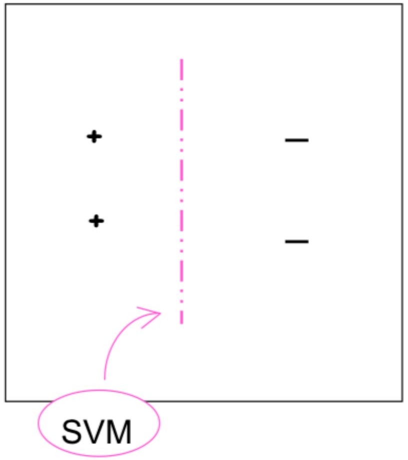
  - **Representation** bias
    - Well explored




  - **Search** bias
    - We know very little



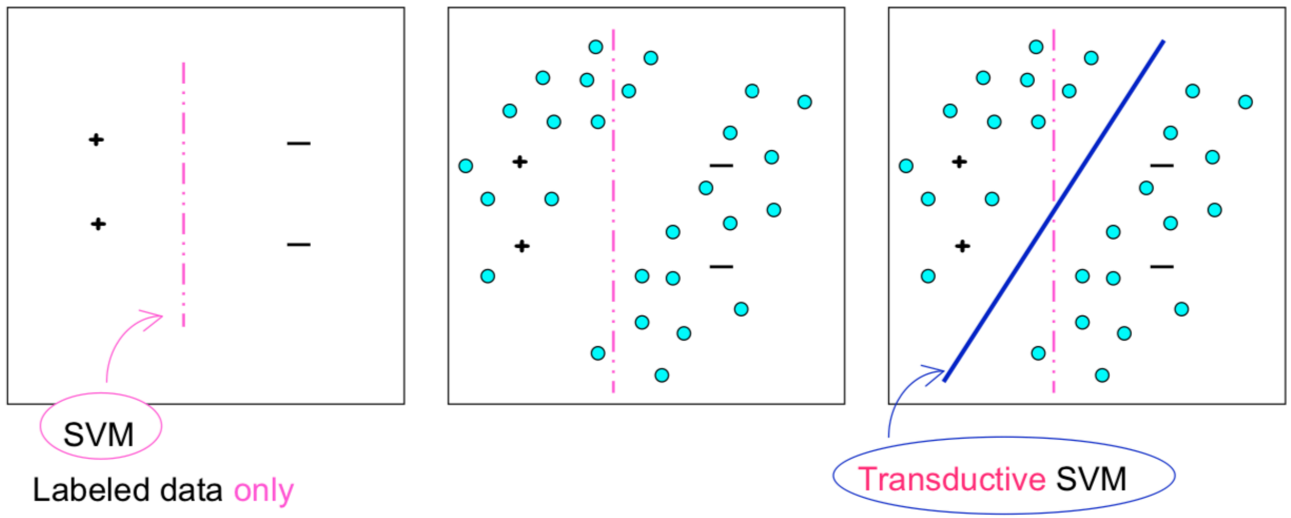
# Semi-supervised induction




Labeled data only

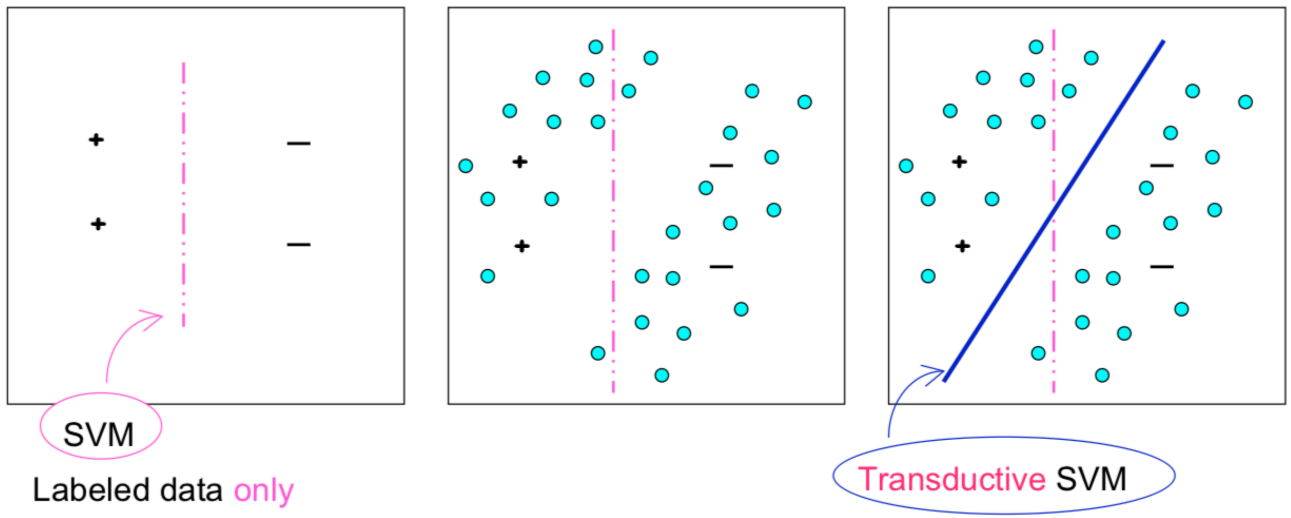
...

# Semi-supervised induction




...

# Semi-supervised induction




- Necessity of a **prior assumption**
  - The decision function **does not cut** through **high density regions** of $X$
    - $\mathbf{P}_X$ is related to $\mathbf{P}_{Y|X}$

# How to derive guarantees for semi-supervised learning?

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

  If the prior assumption on the unlabeled examples is verified

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

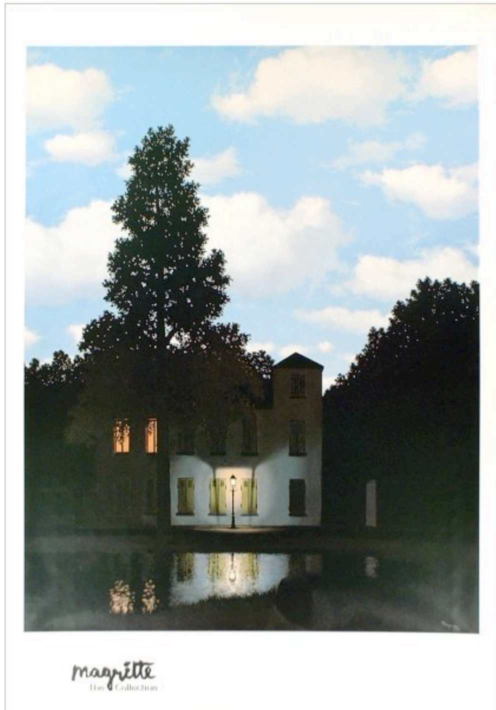
$$\overbrace{m_l \geq \frac{1}{\varepsilon}\left[\ln|\mathcal{H}| + \ln\frac{2}{\delta}\right]} \quad \text{and} \quad \overbrace{m_u \geq \frac{1}{\varepsilon}\left[\ln|\mathcal{H}_{\mathcal{D},\mathcal{X}}(\varepsilon)| + \ln\frac{2}{\delta}\right]}$$

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

# **Lesson** about the guarantees we can seek

- Type of guarantees

  - **If** the signal presents the properties **that we assume true**
  - **Then** the learning method is appropriate to PAC learn (probably approximately) the signal
    if there is enough data points (i.i.d.)




"Lamppost" theorems

# Outline

1. Supervised induction: the classical setting

2. What about Out Of Distribution learning (OOD)?

3. Parallel transport, covariant derivative and transfer learning
   - What they are
   - ... in Machine Learning

4. A way to deal with different spaces of tasks

5. Conclusions

# O.O.D. **scenarios**

1. Learning Using **Privileged Information** (LUPI)

2. Domain Adaptation **(covariate** shift)

3. **Concept** drift

4. **Transfer** learning

# Learning Using Privileged Information

Inspired by learning at school

V. Vapnik and A. Vashist (2009) "A new learning paradigm: Learning using privileged information". *Neural Networks*, vol. 22, no. 5, pp. 544–557, 2009

# Learning Using Privileged Information

Inspired by learning at school

- The goal is to learn a function $h : \mathbf{x} \in \mathcal{X} \to y \in \{-1, +1\}$

- Suppose that at **learning** time there is $\underbrace{\text{more available information}}$ than at **test** time

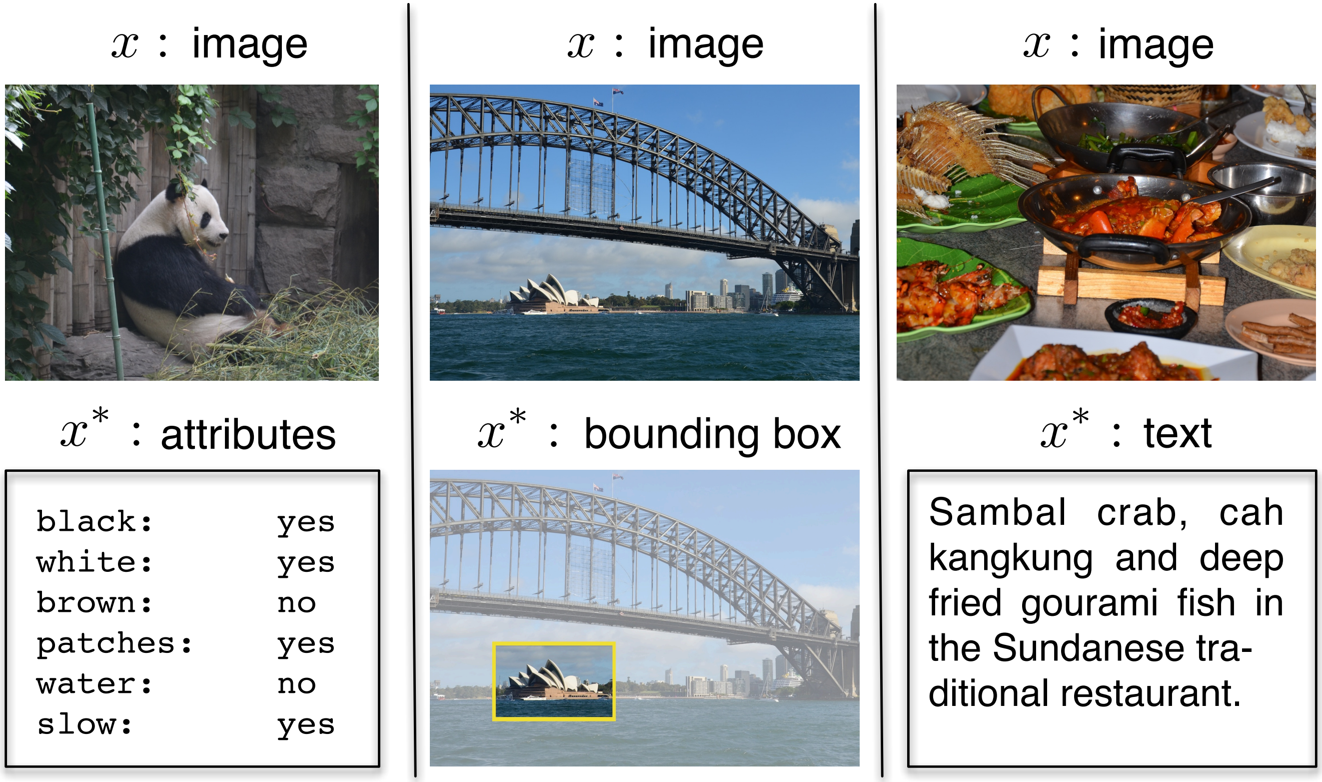
$$\mathcal{S}^* = \{(\mathbf{x}_i, \mathbf{x}_i^*, y_i)\}_{1 \leq i \leq m}$$

- **Can we then improve the generalization performance** wrt. the one obtained with *S* only?

V. Vapnik and A. Vashist (2009) "A new learning paradigm: Learning using privileged information". *Neural Networks*, vol. 22, no. 5, pp. 544–557, 2009

# Learning Using Privileged Information

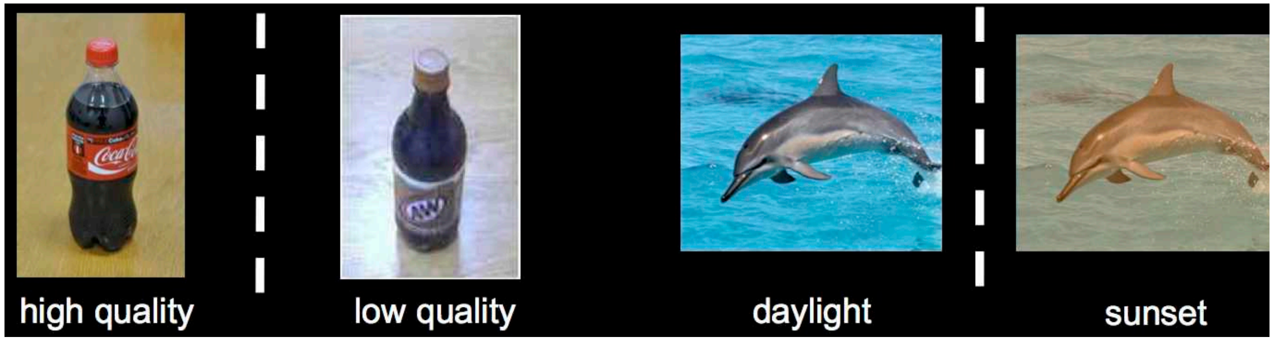
## Illustration in computer vision




V. Sharmanska, N. Quadrianto, and Ch. Lamper (2014) "Learning to transfer privileged information".
*ArXiv preprint arXiv:1410.0389,* 2014

# O.O.D. **scenarios**

- Domain **adaptation**
  - $X_S = X_T$ and $Y_S = Y_T$
  - but **different** distributions $\mathbf{P_X}$
    - E.g. Recognition of the same objects but in a **different environment**



# O.O.D. **scenarios**

- **Concept** shift
  - $X_S = X_T$ and $Y_S = Y_T$
  - but **different** distributions $\mathbf{P_{Y|X}}$
    - E.g. Spam detection for ≠ users

      *conference announcements* are interesting to me

      and a nuisance for my children

# O.O.D. **scenarios**

- **Transfer** learning
  - $X_S \neq X_T$ and/or $Y_S \neq Y_T$
    - E.g. learning to **play chess** after having learned to **play checkers**

Recall the Two questions

1. Which link between the **source** and the **target**?

2. What kind of guarantees can we look for?

# Which **link** between **training** and **testing**?

## LUPI

- “At the core of our work lies the insight that **privileged information** allows us to **distinguish between easy and hard examples** in the training set.

- **Assuming** that examples
  - that are easy or hard with respect to the **privileged** information
  - **will also** be easy or hard with respect to the **original data**,

  we enable information transfer from the privileged to the original data modality.

# One solution: SVM+

- The classical optimization problem

$$\begin{cases} \min \frac{1}{2}\langle \omega, \omega \rangle + C \sum_{i=1}^{m} \xi_i \\ \text{s.t. } y_i[\langle \omega, x_i \rangle + b] \geq 1 - \xi_i, \qquad i = 1, \ldots, m. \end{cases}$$

- is changed into

$$\begin{cases} \min \frac{1}{2}\left[\langle \omega, \omega \rangle + \gamma \langle \omega^*, \omega^* \rangle\right] + C \sum_{i=1}^{m} \left[\langle \omega^*, x^* \rangle + b^*\right] \\ \text{s.t. } y_i[\langle \omega, x_i \rangle + b] \geq 1 - [\langle \omega^*, x_i^* \rangle + b^*], \qquad i = 1, \ldots, m, \\ \quad [\langle \omega^*, x_i^* \rangle + b^*] \geq 0, \qquad i = 1, \ldots, m, \end{cases}$$

*C* and *γ* are hyperparameters

- Intuition:
  - Identify the **difficult examples** (outliers)
  - Thus coming back to the **realizable case** and obtain **convergence rates** of **1/n** instead of 1/sqrt(n)

# Bounds between the **real** risk and the **empirical** risk

By removing the “problematic” examples, you go

- From the **non realisable** case ($\mathcal{H}$ finite)

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

- To the **realisable** one ($\mathcal{H}$ finite)

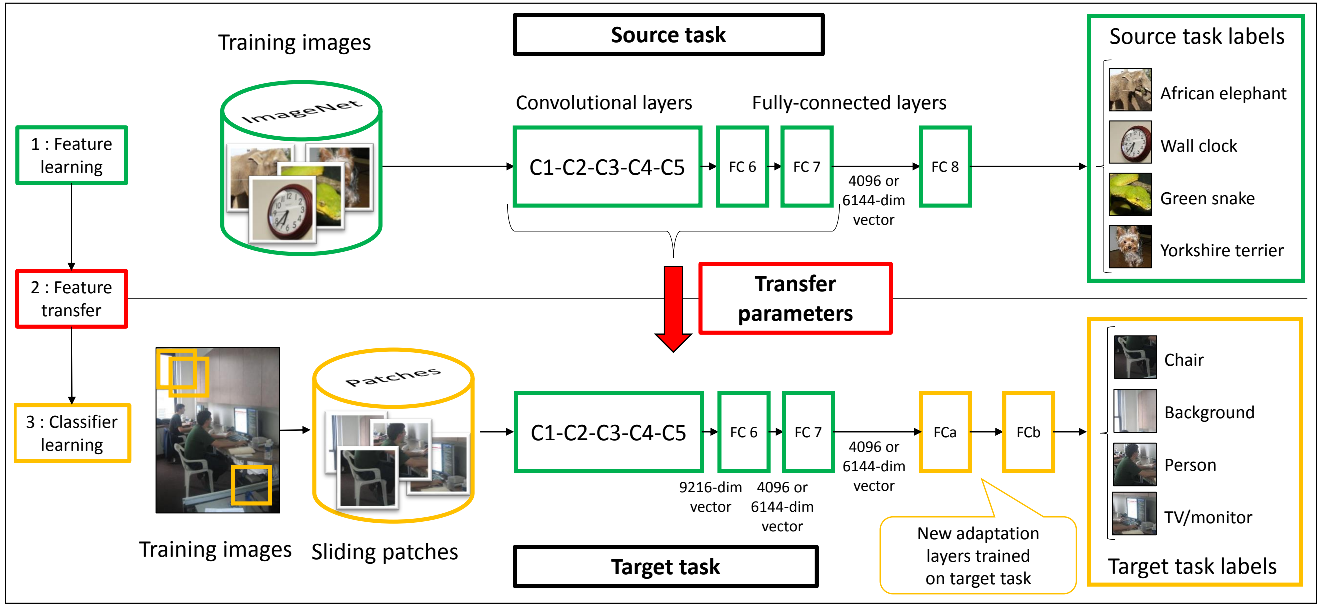
$$\forall h \in \mathcal{H}, \forall \delta \leq 1 : \quad P^m \left[ R_{\text{Réel}}(h) \;\leq\; R_{\text{Emp}}(h) \;+\; \frac{\log |\mathcal{H}| + \log \frac{1}{\delta}}{m} \right] \;>\; 1 - \delta$$

## Transfer Learning

# Which **link** between **training** and **testing**?

## Transfer Learning

– Reuse the **latent space** learnt on the source data




From Oquab, M., Bottou, L., Laptev, I., & Sivic, J. (2014). **Learning and transferring mid-level image representations using convolutional neural networks**. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 1717-1724).

Baldock, R., Maennel, H., & Neyshabur, B. (2021). **Deep learning through the lens of example difficulty**. *Advances in Neural Information Processing Systems, 34*.

# Which **link** between **training** and **testing**?

## Transfer Learning

- Reuse the **latent space** learnt on the source data

- Re-use the first layers of a NN trained on task **A**
- And fine-tune on task **B**

→ **Increases** the performance wrt. to training on task B alone

## Transfer Learning

- Guarantees function of

- Guarantees function of
  - The **quality** of the **source hypothesis** on the source task
    - The **better** $h_S$, the **better** $h_T$

# Transfer Learning

- Guarantees function of
  - The **quality** of the **source hypothesis** on the source task
    - The **better** $h_S$, the **better** $h_T$
  - A "**distance**" between the source task and the target one
    - The **smaller** the distance, the **better** the transfer

# Transfer Learning

Really?

- Guarantees function of
  - The **quality** of the **source hypothesis** on the source task
    - The **better** $h_S$, the **better** $h_T$
  - A "**distance**" between the source task and the target one
    - The **smaller** the distance, the **better** the transfer
  - The size of the **target training data**
    - The **larger** the target training data set, the **useless** the transfer

# Outline

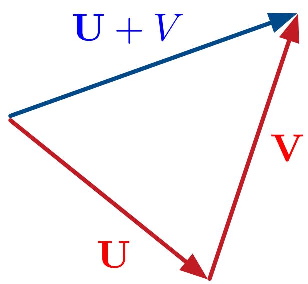
1. Supervised induction: the classical setting
2. What about Out Of Distribution learning (OOD)?
3. Parallel transport, covariant derivative and transfer learning
   - What they are
   - ... and in Machine Learning
4. A way to deal with different spaces of tasks
5. Conclusions

# Parallel Transport

# and Covariant Derivative

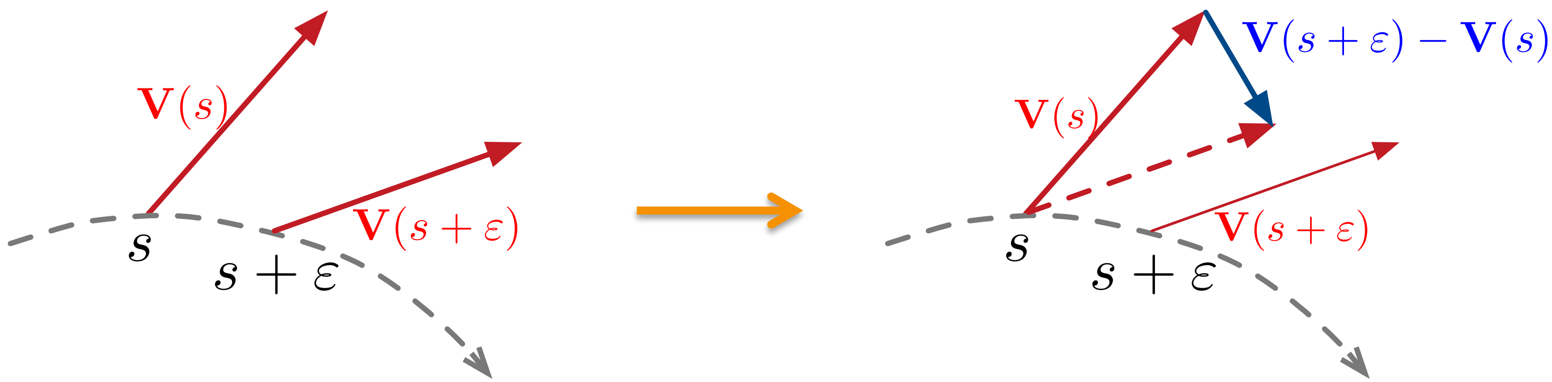
# Euclidian geometry

- **Addition** of vectors



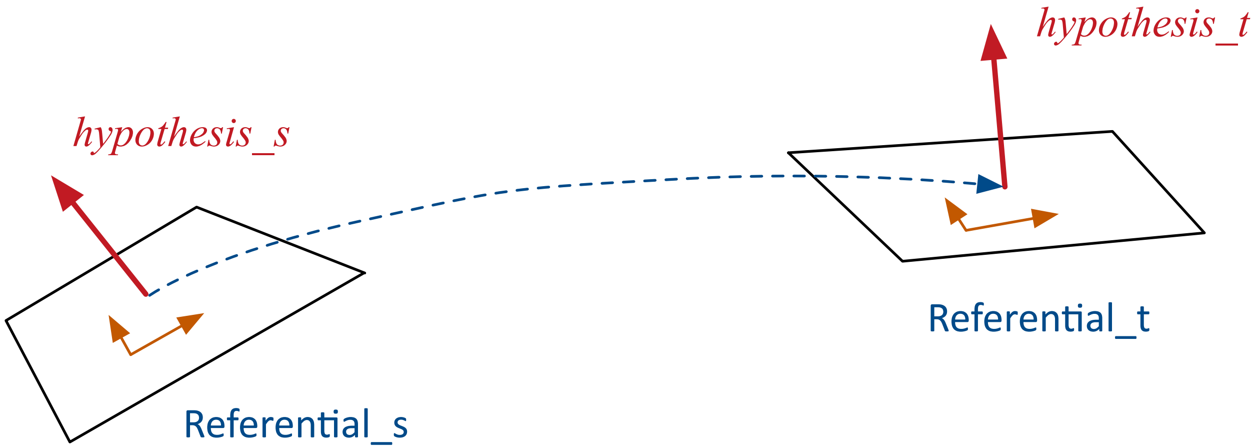

- **Substraction** of vectors and **derivative**

$$\frac{\mathrm{d}\mathbf{V}}{\mathrm{d}s} = \lim_{\varepsilon \to 0} \frac{\mathbf{V}(s+\varepsilon) - \mathbf{V}(s)}{\varepsilon}$$



# Non Euclidian geometry

- Substraction of vectors and **derivative**




We can **no** longer **directly compare** vectors (or tensors)

Necessity of the covariant derivative

# Parallel transport

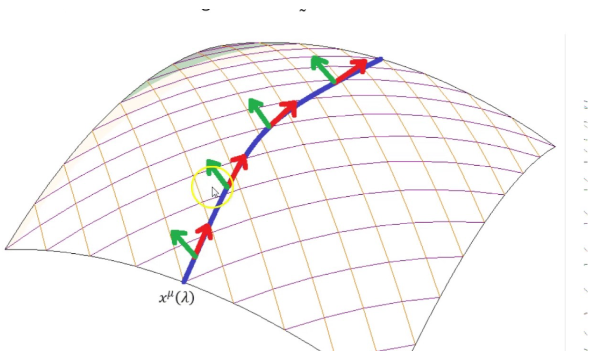
- **Transport** a vector (or a tensor) **parallel to itself** along a curve

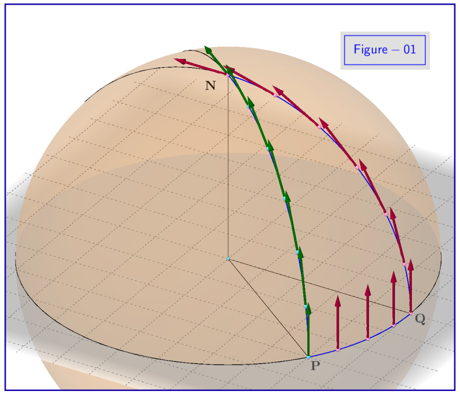
Covariant derivative = 0

Kronecker symbol

$$(\partial_k V^i)^{\text{covariant}} = \partial_k V^i + \Gamma^i_{jk} V^j$$

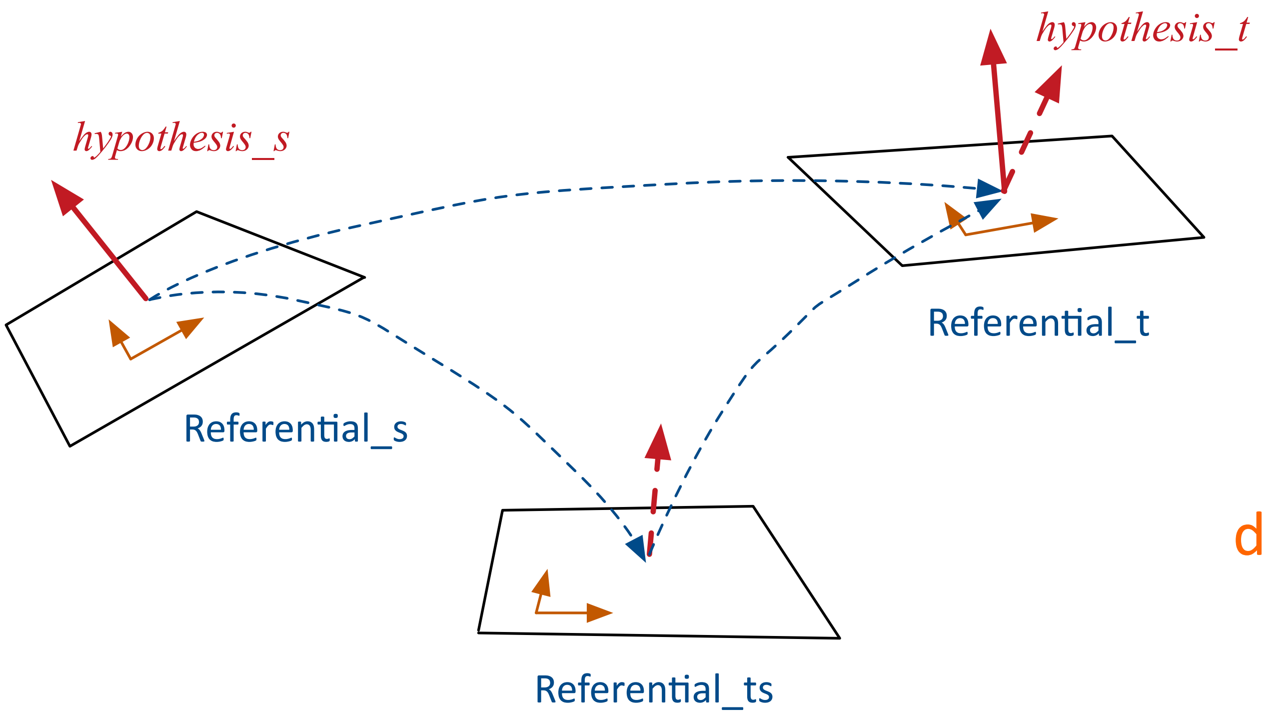
$$V^i(x^k)^{\text{parallel transported}} = V^i(x^k) + \Gamma^i_{jk} V^j \Delta x^k$$




Path **dependent!**

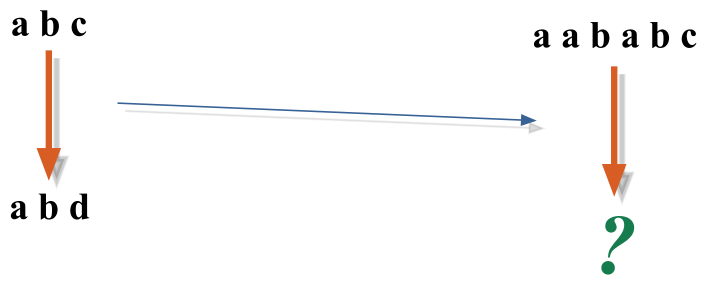
# Transfer and **path dependence**

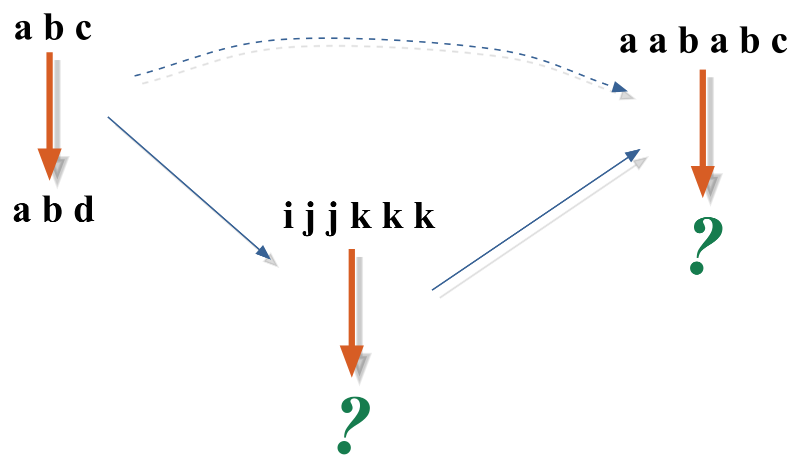



Path dependence

**Transfer** $\stackrel{?}{=}$ **Parallel transport** of hypothesis from source to target

# Transfer and **path dependence**




...

## Parallel transport in **ML works**

Transfer = parallel transport of the source hypothesis

1. Tracking

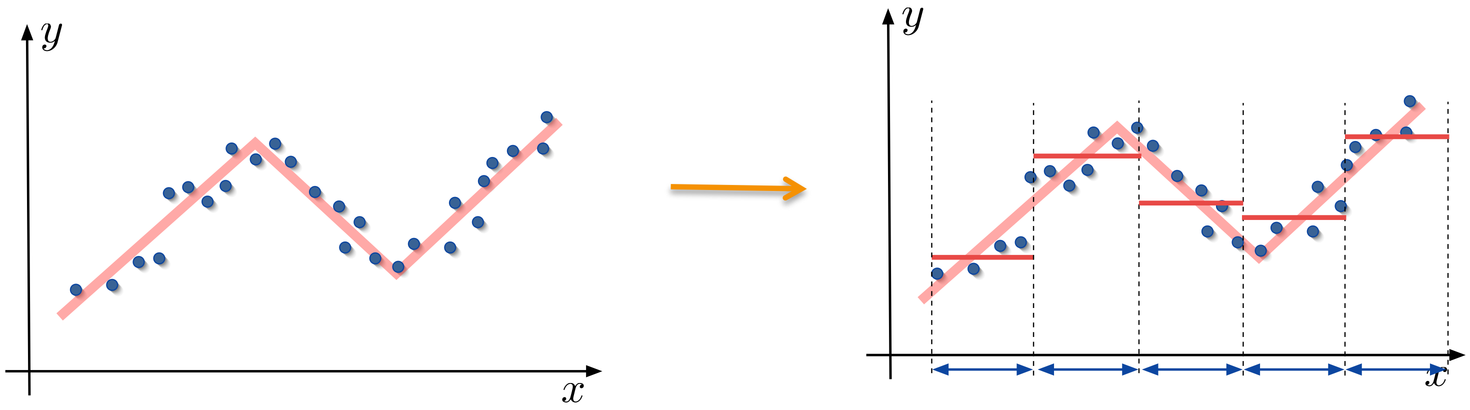
2. Computer vision

3. Curriculum learning

# Tracking

Instead of **learning a complex function** over the whole of $\mathcal{X}$

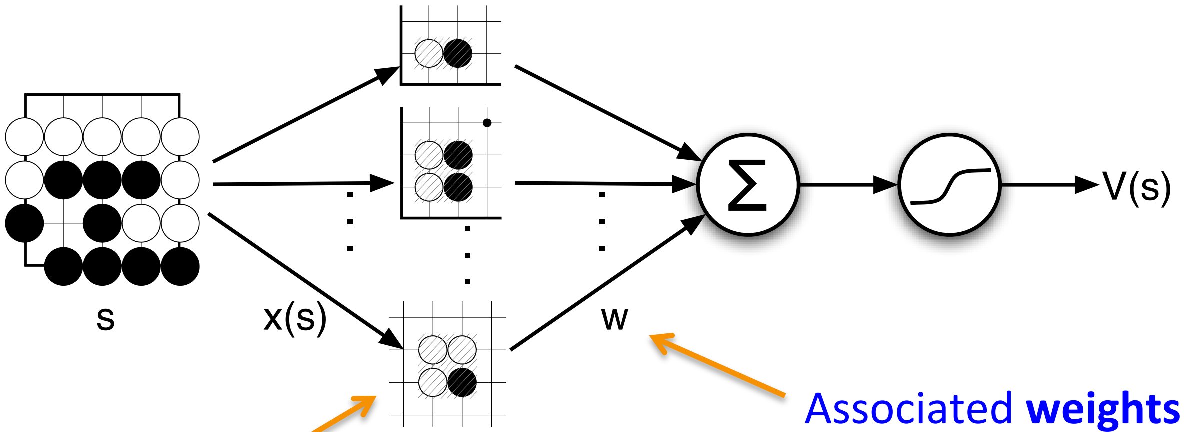
- **If** you know that the task is slowly evolving with time
- Learn a simple **local** function




R. Sutton and A. Koop and D. Silver (2007) *"On the role of tracking in stationary environments"* (ICML-07) Proceedings of the 24th international conference on Machine learning, ACM, pp.871-878, 2007.

## Tracking to play Go

- 5 x 5 Go
  - More than 5 x $10^{10}$ unique positions
- Usual approach: learn a **general** evaluation function V(*s*) valid $\forall$s



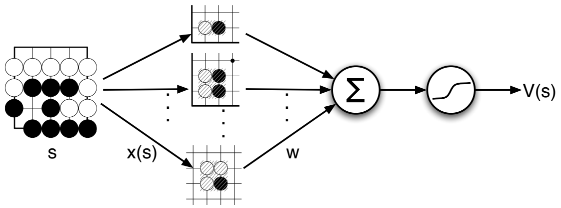

Associated **weights** (learnt)

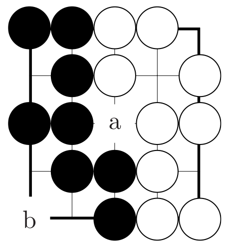
**Features** describing the situation

# Tracking in stationary environments

- Tracking approach: learn an **evaluation** function V(*s*)

  **local** to the current *s*

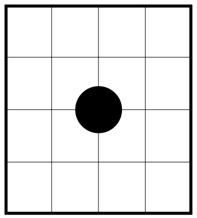



In general, playing (a) (center) is better than playing (b)

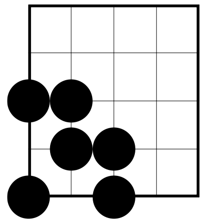
BUT

In this situation, playing (b) is better than playing (a)



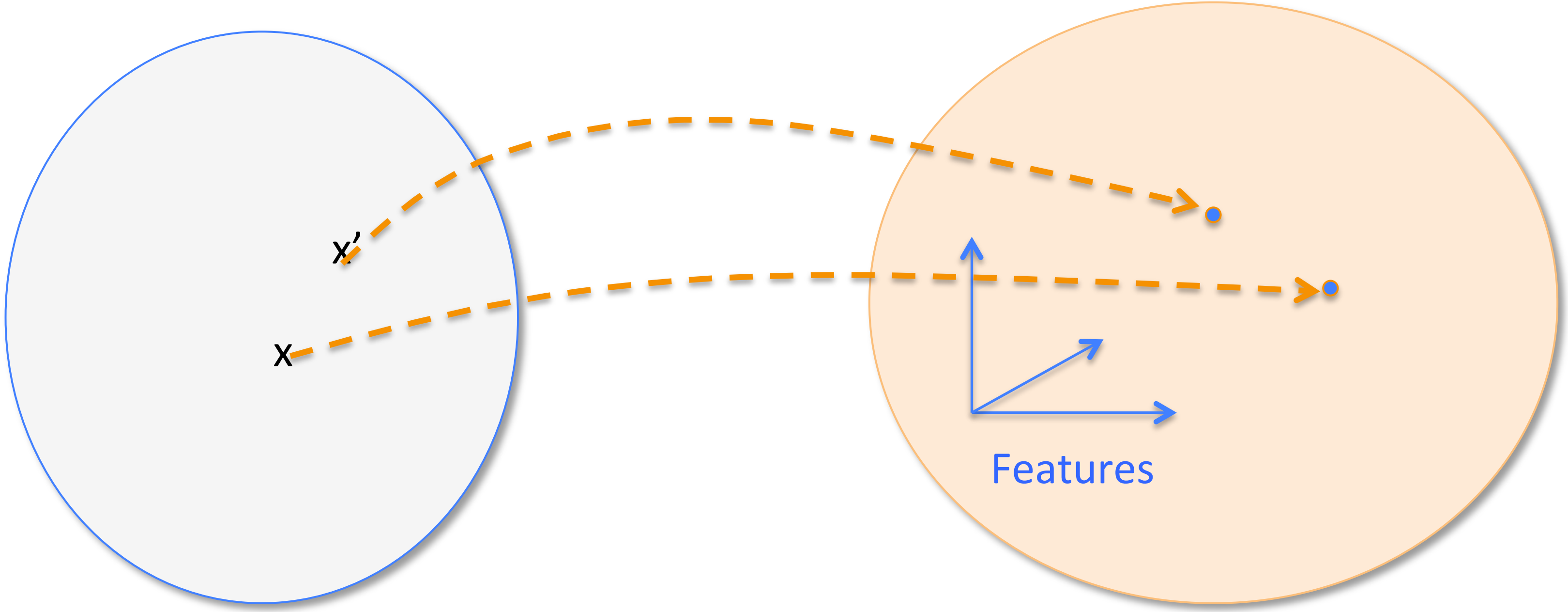
More weight

More weight



# Tracking as **local changes** of representation

...




Space of go positions

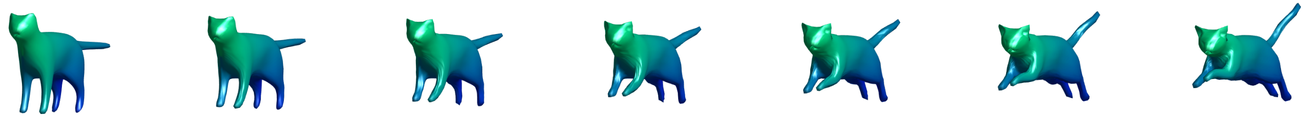
Embedding

Space of representations

Weighted features

# Computer vision

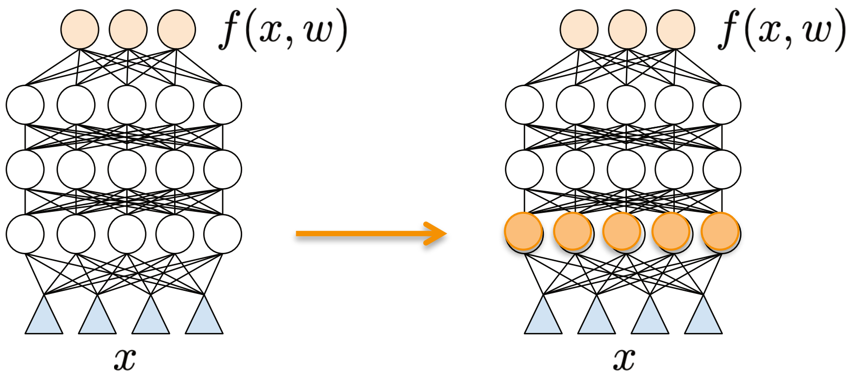
...



Bauer, M., Klassen, E., Preston, S. C., & Su, Z. (2018). **A diffeomorphism-invariant metric on the space of vector-valued one-forms**. arXiv preprint arXiv:1812.10867.

# Parallel transport in computer vision

...




Standard CNN

PTCNet

Parallel Transported
Convolution layer

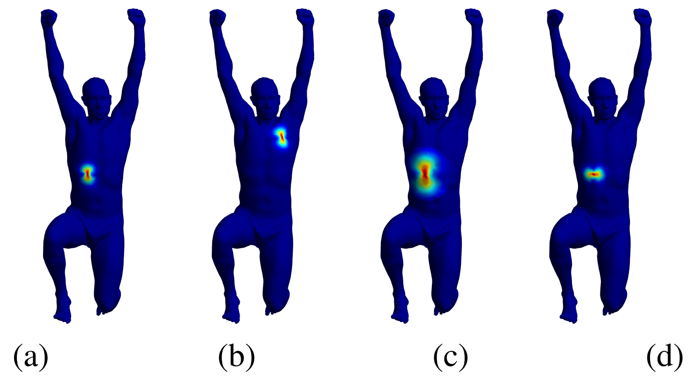



Figure 1: A compactly supported kernel (a) is transported on a manifold from the FAUST data set [2] through translation (b), translation + dilation (c) and translation + rotation (d).

Schonsheck, S. C., Dong, B., & Lai, R. (2018). **Parallel transport convolution: A new tool for convolutional neural networks on manifolds**. arXiv preprint arXiv:1805.07857.

# Curriculum building

- We expect that transfer is easy when source and target tasks are "close"
- And it may be difficult to transfer across tasks that are "far away"

But **how to measure** "*closeness*"

and "*far away*" for learning tasks?

Define a geometry over the space of tasks

# **Geometry** of the space of tasks

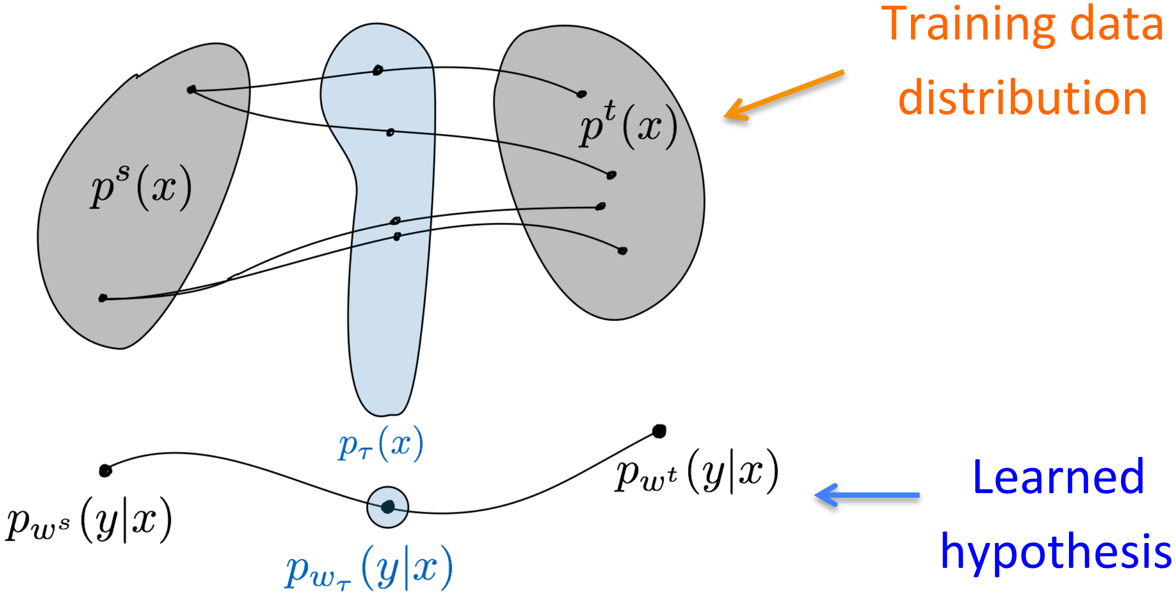
- Desiderata

  1. Should **incorporate the hypothesis space**,
     and not only the "distance" between the inputs (as is usually done)
     - For instance, it is often observed that *transferring larger models is easier*. The geometry should reflect this.

  2. The distance between tasks is **not symmetrical**

Gao, Y., & Chaudhari, P. (2021, July). **An information-geometric distance on the space of tasks**. In *International Conference on Machine Learning* (pp. 3553-3563). PMLR.

# Idea



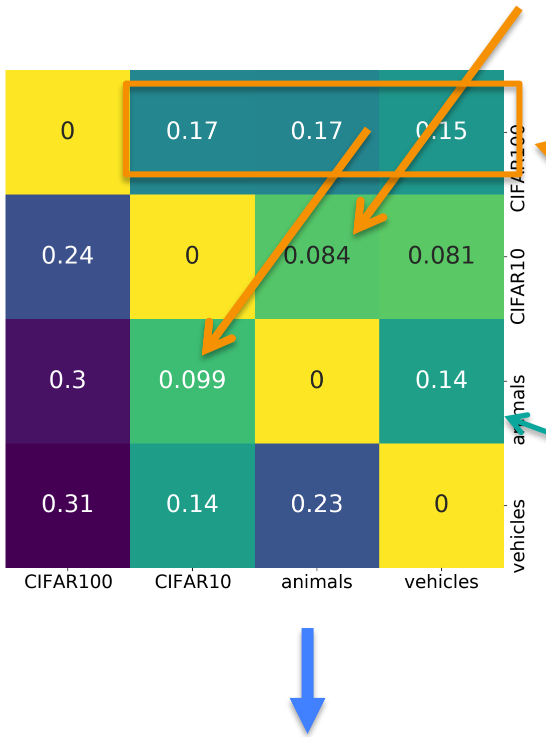

Modify **conjointly** the training data distribution and the **learned hypothesis**

Compute iteratively the intermediate training sets such that

- at each step $\tau$ the new task is close to
- what can be learned by **the current learner** (characterized by its **current hypothesis**)

# Experimental results

- Using an **8-layer convolutional NN** (ReLU, dropout, batch-normalization) with a final fully connected layer



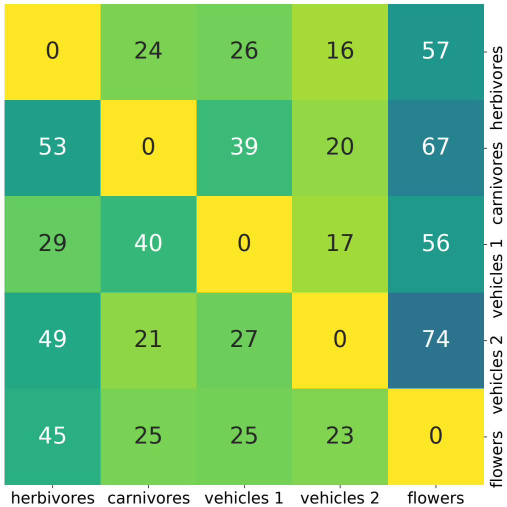

Distance is **asymmetrical**

- CIFAR-10 to animals < animals to CIFAR-10
- CIFAR-100 to any other is much easier than the reverse

Estimated task distances

# Experimental results

- Using an **8-layer convolutional NN**




- And a **wide residual network** (WRN-16-4): larger capacity


Distance is much **reduced** using a **larger capacity** model

# Conclusions

- **Interesting** work
  - New definition of **distance** between tasks
    - **Asymmetrical**
    - Depends on the **capacity** of the learning system
  - New way to build a **curriculum**

# Conclusions

- Interesting work
  - New definition of **distance** between tasks
    - **Asymmetrical**
    - Depends on the **capacity** of the learning system
  - New way to build a **curriculum**

- **Limits**
  - Still a **crude** way to build intermediate tasks
  - **Same** input-output source and target domains!!!
  - **Same hypothesis space** in both source and target domains!!!

# Conclusions

- Interesting work
  - New definition of **distance** between tasks
    - **Asymmetrical**
    - Depends on the **capacity** of the learning system
  - New way to build a **curriculum**

- Limits
  - Still a **crude** way to build intermediate tasks
  - **Same** input-output source and target domains!!!
  - **Same hypothesis space** in both source and target domains!!!

Not **general** transfer learning

What if the space of tasks is not continuous?

# Outline

1. Supervised induction: the classical setting

2. What about Out Of Distribution learning (OOD)?

3. Parallel transport, covariant derivative and transfer learning

   – What they are

   – ... in Machine Learning

4. A way to deal with different spaces of tasks

5. Conclusions

# A LUPI type of algorithm for transfer learning

**TransBoost**

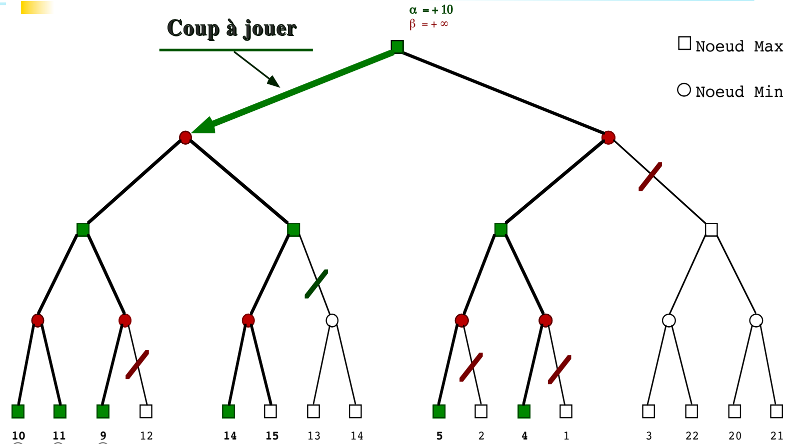
A **method** for transfer learning between different tasks

and **what it tells**

Cornuéjols, A., Murena, P. A., & Olivier, R. (2020). **Transfer learning by learning projections from target to source**. In 18th *International Symposium on Intelligent Data Analysis*, IDA 2020, Konstanz, Germany, April 27–29, 2020, Proceedings 18 (pp. 119-131). Springer International Publishing.

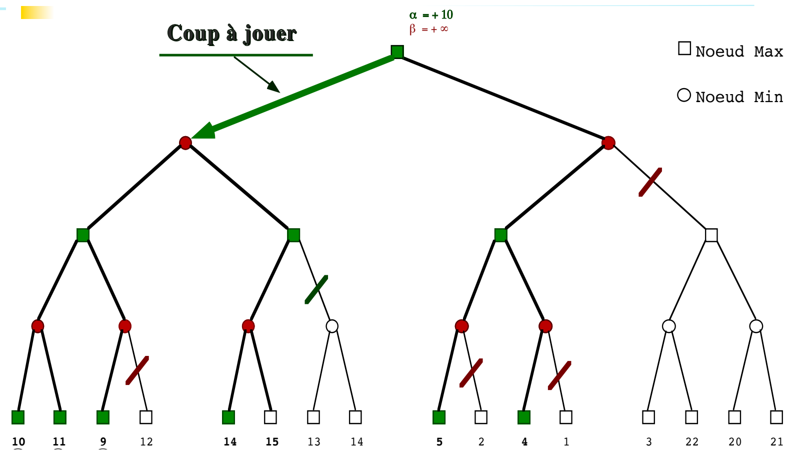
# A LUPI type of algorithm for transfer learning

Taking decision when the current information is **incomplete**




...

# Algorithms for games




Taking decision when the current information is **incomplete**

- Which move to play?

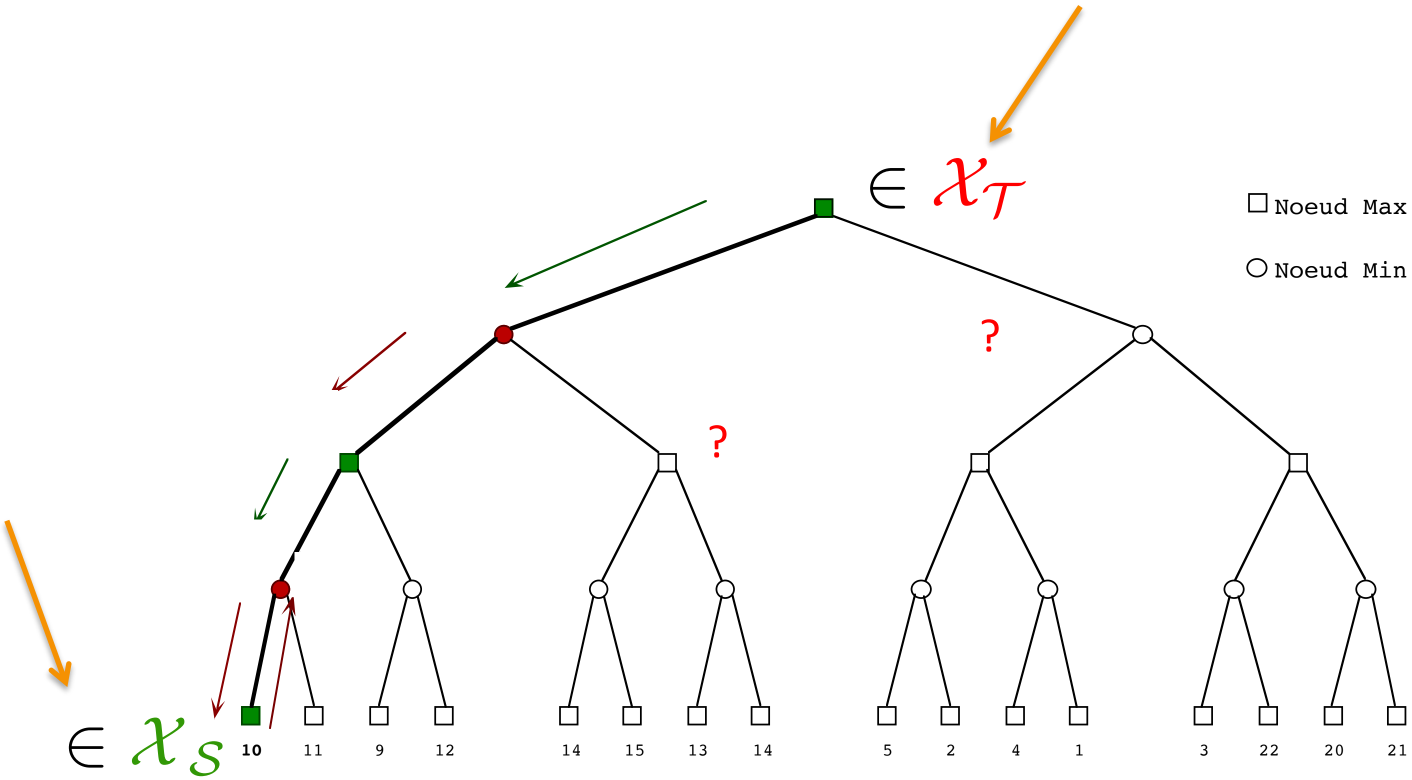
  The evaluation function is **insufficiently informed** at the root (current situation)

  1. **Query experts** that have more information about potential outcomes

  2. **Combination** of the estimates through MinMax

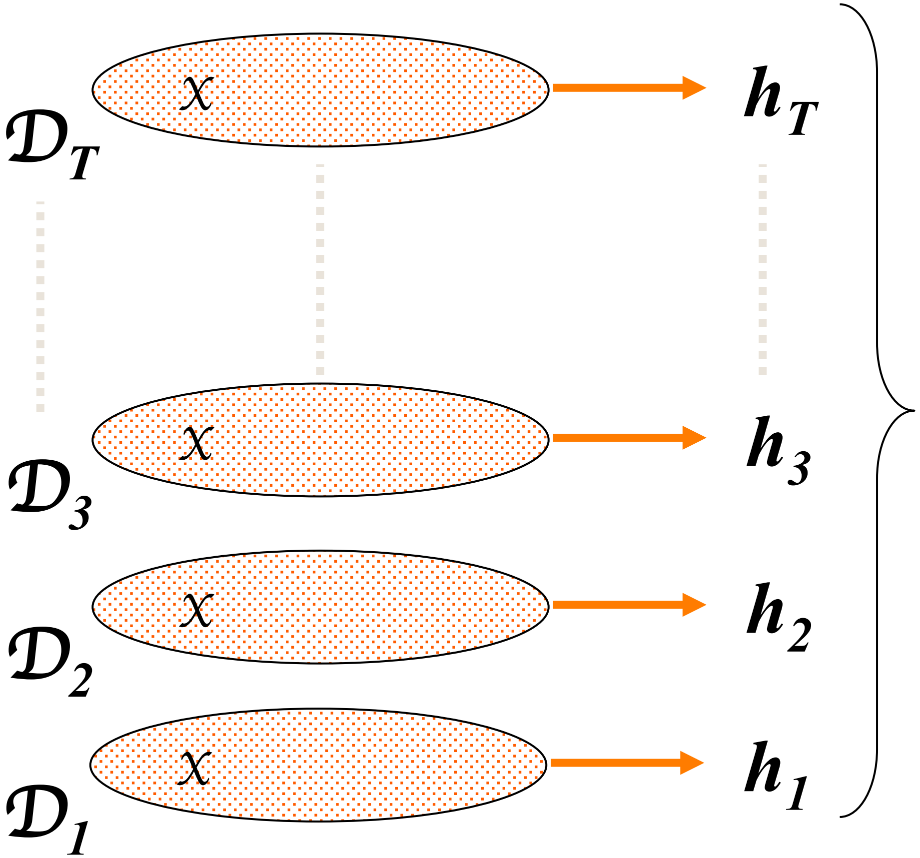
*“Experts” may live in **input spaces** that are **different***

...


$\in \mathcal{X}_T$
Noeud Max
Noeud Min
?
?
$\in \mathcal{X}_S$
10
11
9
12
14
15
13
14
5
2
4
1
3
22
20
21

Can we do the “same” for transfer learning?

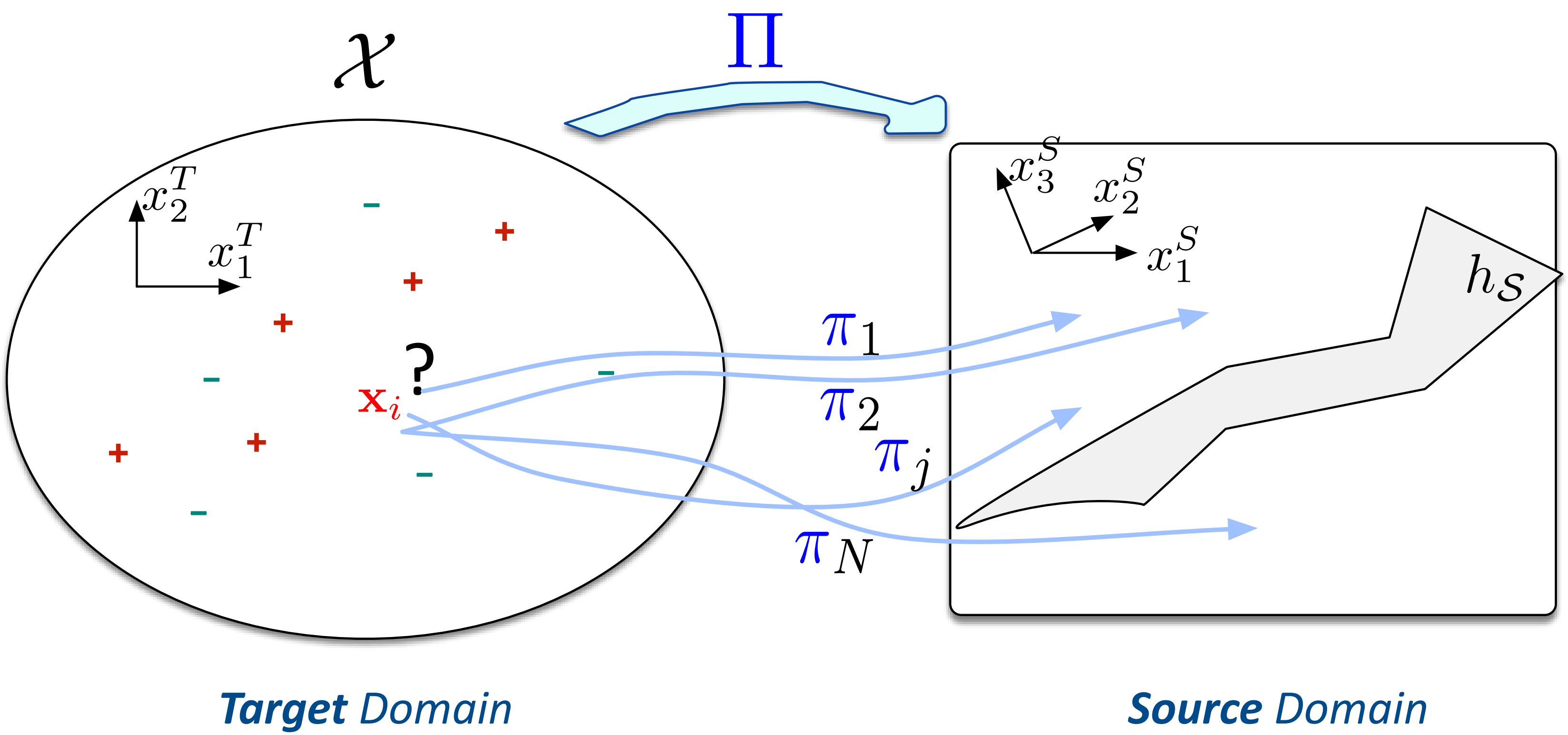
# Boosting




$$H(\mathbf{x}) \;=\; \mathrm{sign}\left[\sum_{t=1}^{T} \alpha_t \, h_t(\mathbf{x})\right]$$

- How to compute $\mathcal{D}_t$ from $\mathcal{D}_{t-1}$ and thus $h_t$?
- How to compute the $\alpha_t$ ?

# TransBoost




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

# TransBoost

- Principle:
  - Learn "*weak projections*": $\pi_i : \mathcal{X}_\mathcal{T} \rightarrow \mathcal{X}_\mathcal{S}$
    - Using the target training data: $S_\mathcal{T} = \{(\mathbf{x}_i^\mathcal{T}, y_i^\mathcal{T})\}_{1 \leq i \leq m}$
  - With boosting
    - **Projection** $\pi_n$ such that : $\varepsilon_n \doteq \mathbf{P}_{i \sim D_n}[h_\mathcal{S}(\pi_n(\mathbf{x}_i)) \neq y_i] < 0.5$
    - **Re-weight** the training time series and loop until termination
  - Result

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

# TransBoost

**Algorithm 1:** Transfer learning by boosting

**Input**: $h_{\mathcal{S}} : \mathcal{X}_{\mathcal{S}} \rightarrow \mathcal{Y}_{\mathcal{S}}$ the source hypothesis
$\mathcal{S}_{\mathcal{T}} = \{(\mathbf{x}_i^{\mathcal{T}}, y_i^{\mathcal{T}}\}_{1 \leq i \leq m}$: 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 $\pi_i : \mathcal{X}_{\mathcal{T}} \rightarrow \mathcal{X}_{\mathcal{S}}$ st. $h_{\mathcal{S}}(\pi_i(\cdot))$ performs better than random on $D_n(\mathcal{S}_{\mathcal{T}})$ ;

Let $\varepsilon_n$ be the error rate of $h_{\mathcal{S}}(\pi_i(\cdot))$ on $D_n(\mathcal{S}_{\mathcal{T}})$ : $\varepsilon_n \doteq \mathbf{P}_{i \sim D_n}[h_{\mathcal{S}}(\pi_n(\mathbf{x}_i)) \neq y_i]$ (with $\varepsilon_n < 0.5$) ;

Computes $\alpha_i = \frac{1}{2} \log_2\left(\frac{1-\varepsilon_i}{\varepsilon_i}\right)$ ;

Update, for $i = 1 \ldots, m$:

$$\begin{aligned} D_{n+1}(i) &= \frac{D_n(i)}{Z_n} \times \begin{cases} e^{-\alpha_n} & \text{if } \; h_{\mathcal{S}}\big(\pi_n(\mathbf{x}_i^{\mathcal{T}})\big) = y_i^{\mathcal{T}} \\ e^{\alpha_n} & \text{if } \; h_{\mathcal{S}}\big(\pi_n(\mathbf{x}_i^{\mathcal{T}})\big) \neq y_i^{\mathcal{T}} \end{cases} \\ &= \frac{D_n(i) \, \exp\big(-\alpha_n \, y_i^{(\mathcal{T})} \, h_{\mathcal{S}}(\pi_n(\mathbf{x}_i^{(\mathcal{T})}))\big)}{Z_n} \end{aligned}$$

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

**end**

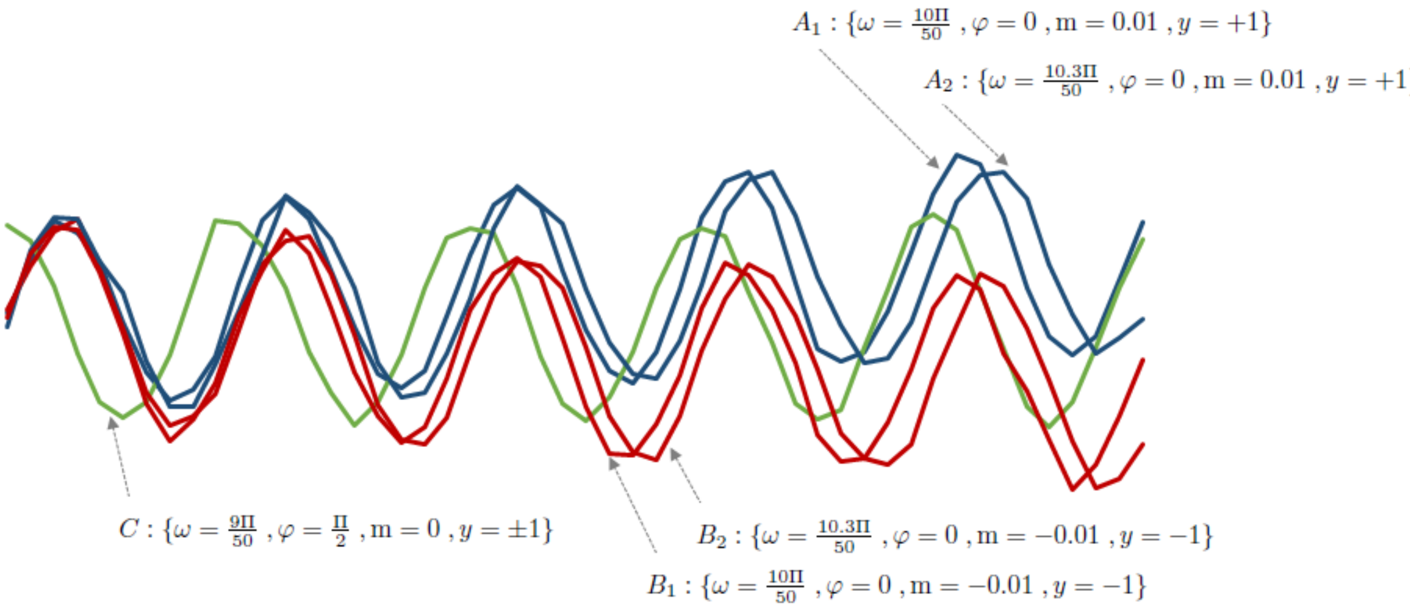
**Output**: the final target hypothesis $H_{\mathcal{T}} : \mathcal{X}_{\mathcal{T}} \rightarrow \mathcal{Y}_{\mathcal{T}}$:

$$\boxed{H_{\mathcal{T}}(\mathbf{x}^{\mathcal{T}}) \; = \; \text{sign}\left\{ \sum_{n=1}^{N} \alpha_n \, h_{\mathcal{S}}\big(\pi_n(\mathbf{x}^{\mathcal{T}})\big) \right\}} \tag{2}$$

# Controlled data

- The **slope** to distinguish between **classes**
- The **shapes** of time series within each class: variety
- The **noise level**

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

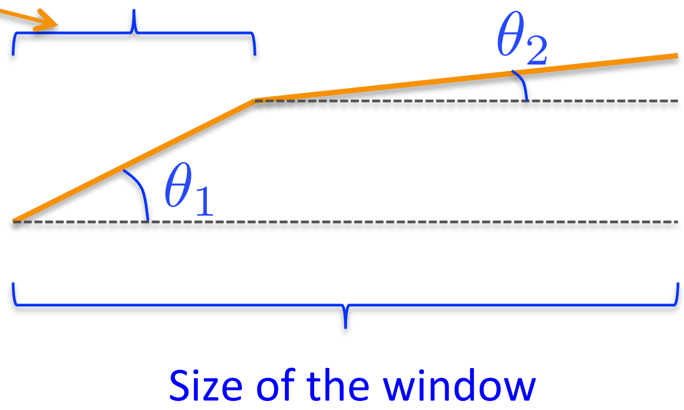


# The set of projections

Randomly generated within constraints

*Hinge functions* (4 parameters)

- **Abscisse** of the hinge
- **Angles** before and after
- Observed **window**



# Results

Learning from **target data only**

TransBoost

On the source domain

Naïve transfert

High noise level

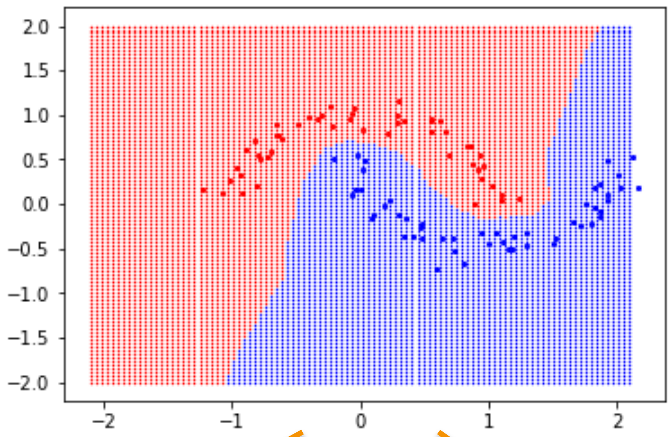
Easy

Large slope

| 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 | **0.08** ± 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 | **0.01** ± 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 | **0.04** ± 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 | **0.10** ± 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 | **0.08** ± 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 | **0.01** ± 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 | **0.02** ± 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 | **0.04** ± 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 | **0.07** ± 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 | **0.01** ± 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 | **0.02** ± 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 | **0.03** ± 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.

# Transfer learning using Transboost

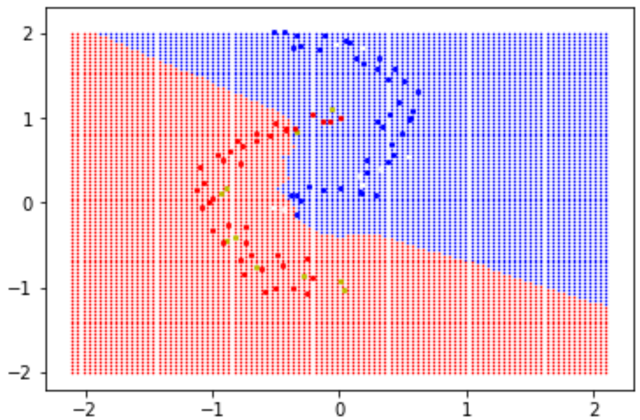


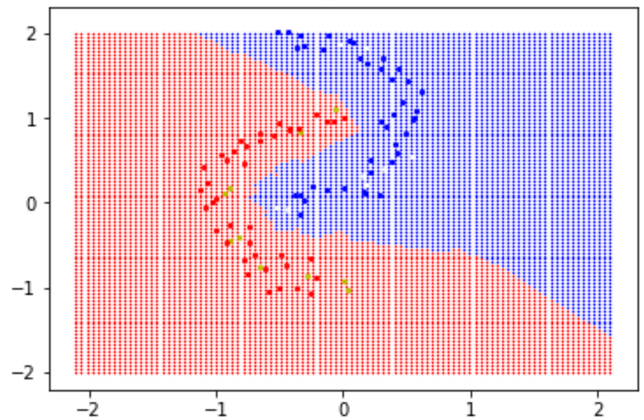

$$\pi_i(\mathbf{x}) \;=\; \mathbf{x} + \mathbf{v}_i$$

$$\pi_i(\mathbf{x}) \;=\; \mathbf{A}_i \cdot \mathbf{x} + \mathbf{v}_i$$

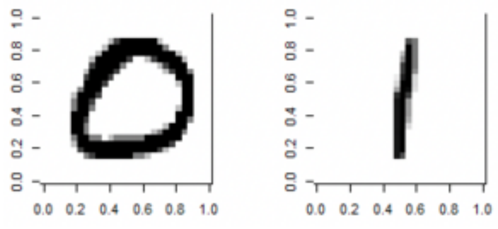
Learning on the target data
(**without transfer**)

Using Transboost

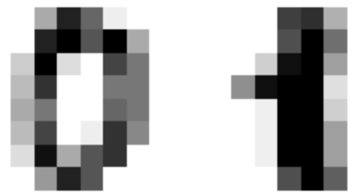


- Illustrations

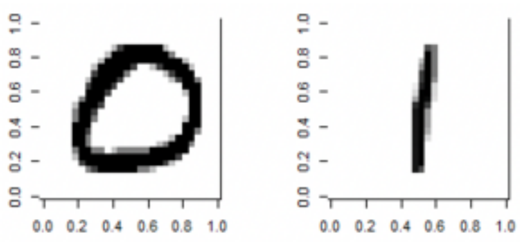



(a) Is it a zero or a one ?

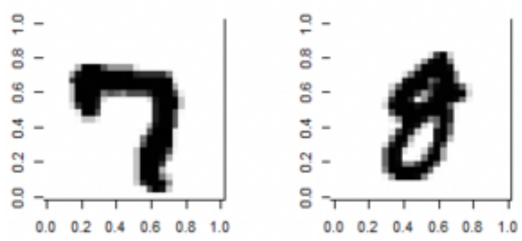


(b) Is it a zero or a one ?

**FIGURE 15:** Transfer learning of the source model 0/1 mnist so that it can distinguish 0/1 sklearn digits




(a) Is it a zero or a one ?




(b) Is it an eight or a seven ?

# Transfer learning using Transboost

- Illustrations

FIGURE 1: Trained model on the data source : is it a picture of a dog or a cat ?

Task A

$$\mathcal{X}_A \neq \mathcal{X}_B$$

Task B

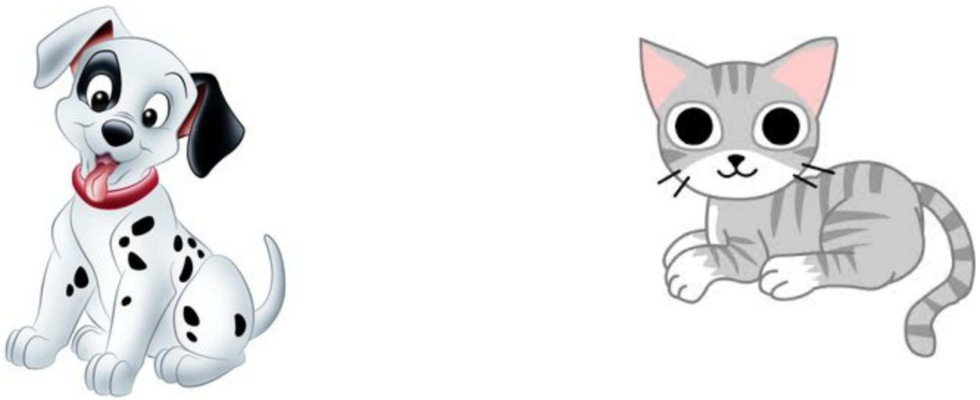
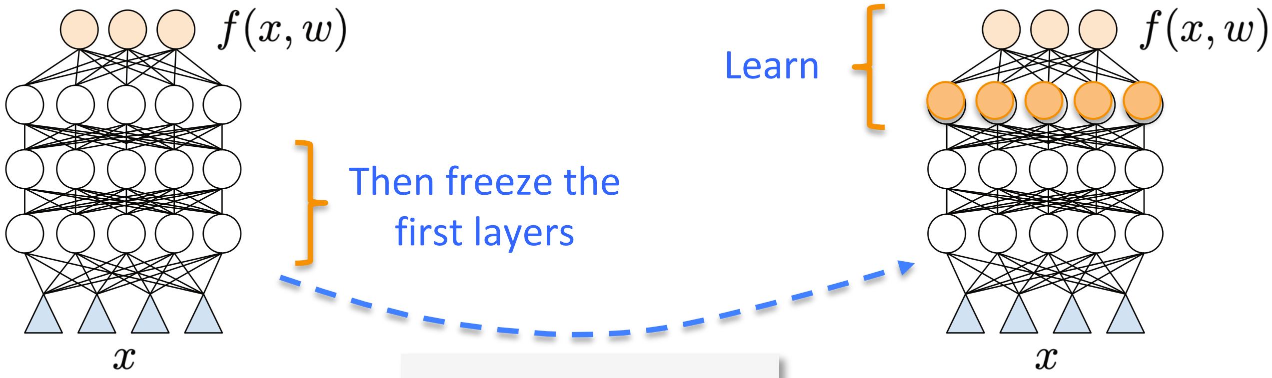


FIGURE 2: Model source transferred on the data target : is it a clip-art of a dog or a cat ?

# Standard Transfer with NNs

...




Transferring the **features**

Learn NN on task A

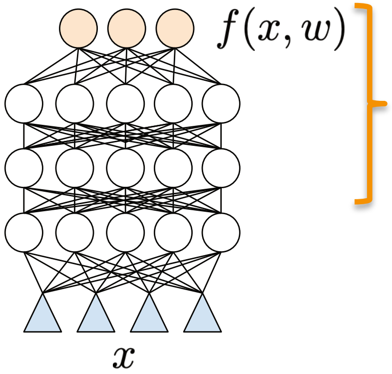
Learn the **last** layers on task B

Same input space $\mathcal{X}_A = \mathcal{X}_B$

From Oquab, M., Bottou, L., Laptev, I., & Sivic, J. (2014). **Learning and transferring mid-level image representations using convolutional neural networks**. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 1717-1724).

# TransBoost with NNs

...



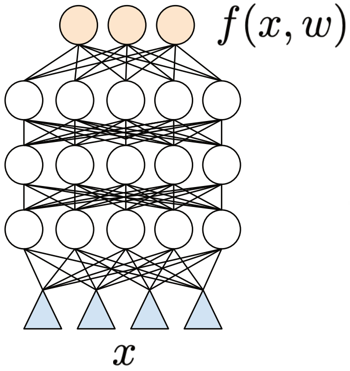

Then **freeze all layers** except the first (and second)

Learn NN on task A

# TransBoost with NNs

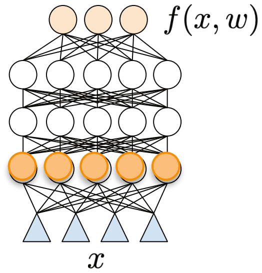
...

Transferring the **decision function**




Then **freeze all layers** except the first (and second)

Learn projection $\pi_1$

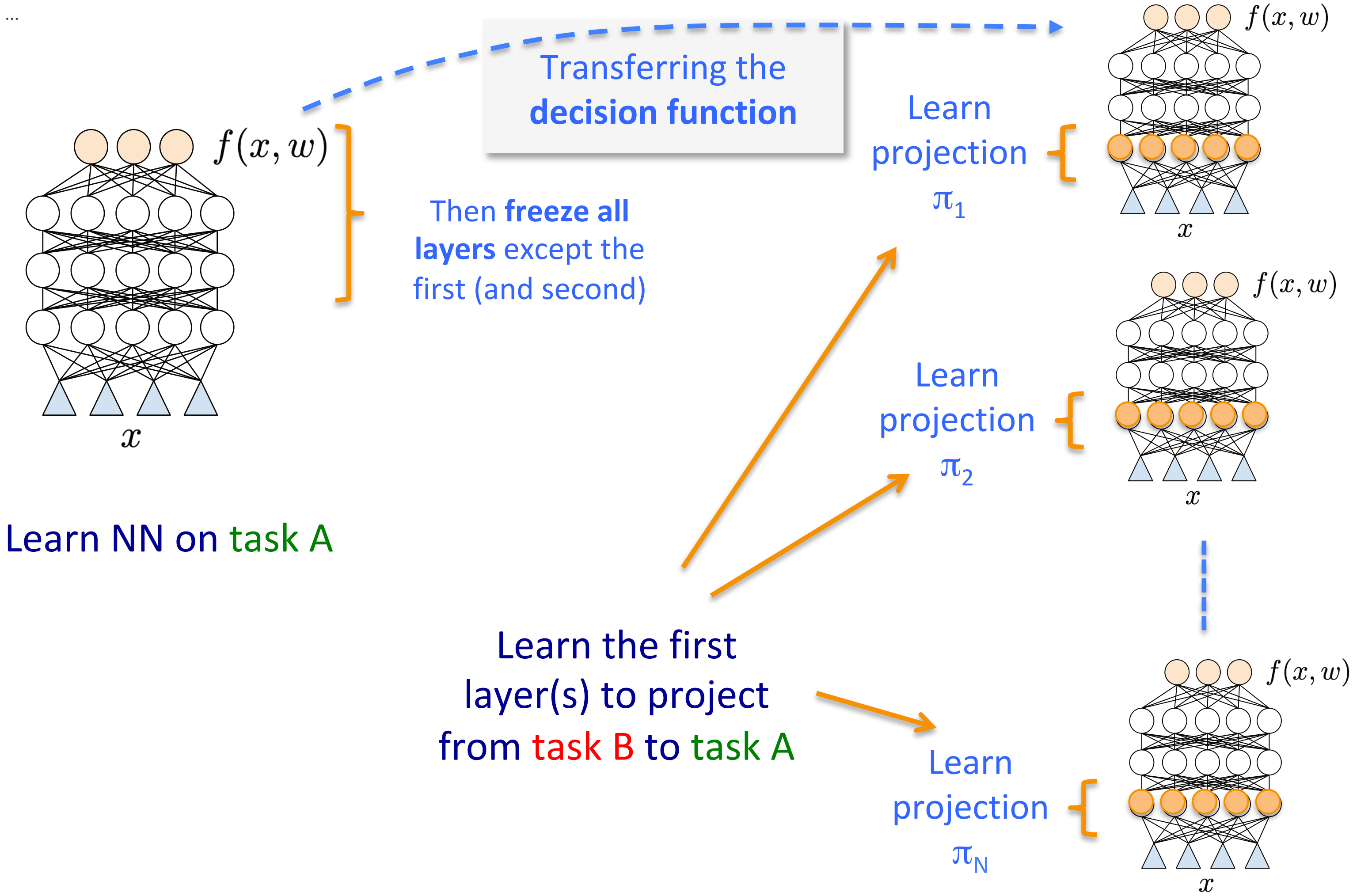



Learn NN on task A

Learn the first layer(s) to project from task B to task A

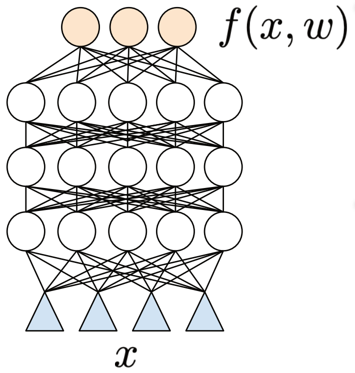
# TransBoost with NNs

...



# TransBoost with NNs

...

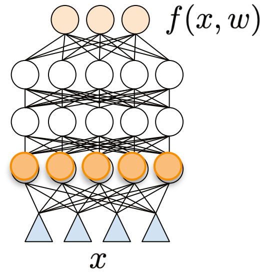



Transferring the **decision function**

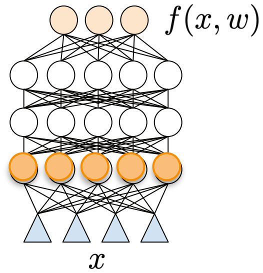
Then **freeze all layers** except the first (and second)

Learn NN on task A

Learn projection $\pi_1$

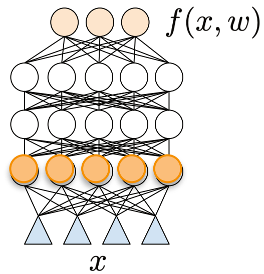



Learn projection $\pi_2$




Learn the first layer(s) to project from task B to task A

Learn projection $\pi_N$



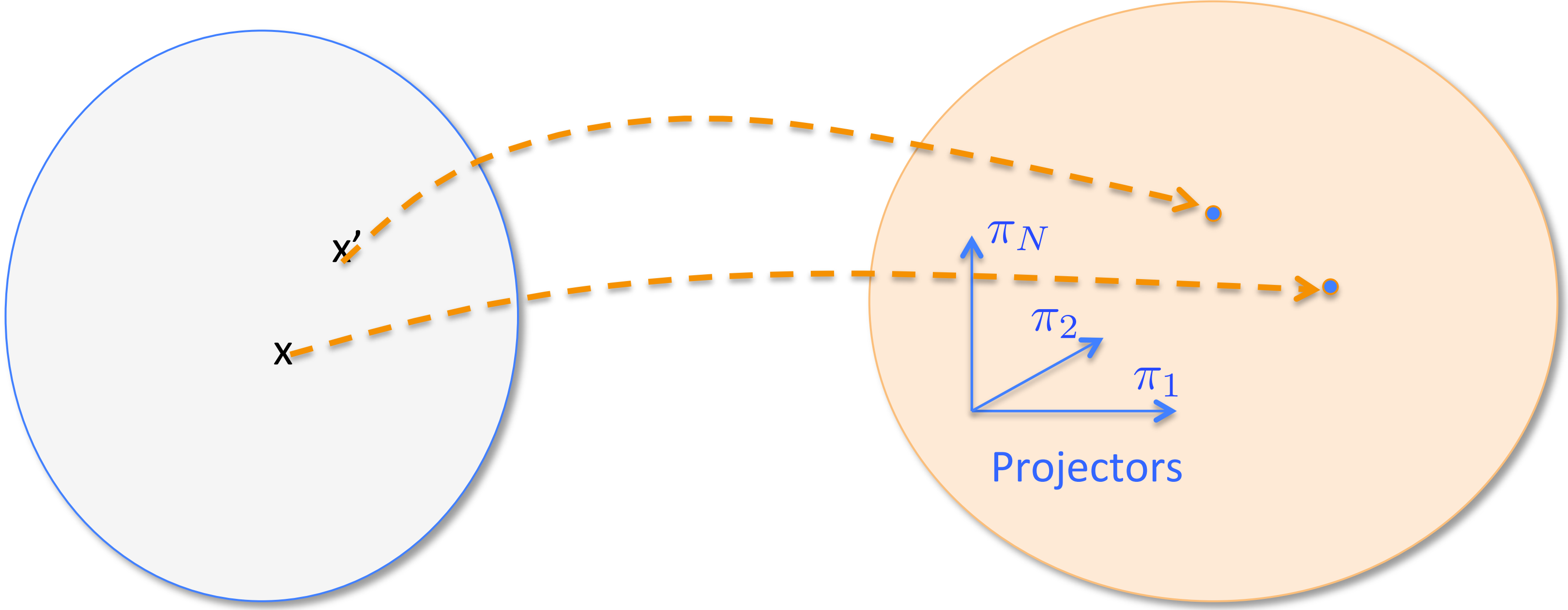

Different input spaces

$$\mathcal{X}_A \neq \mathcal{X}_B$$

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

# Transboost as **local changes** of representation

...




Space of learning tasks

Target training sets

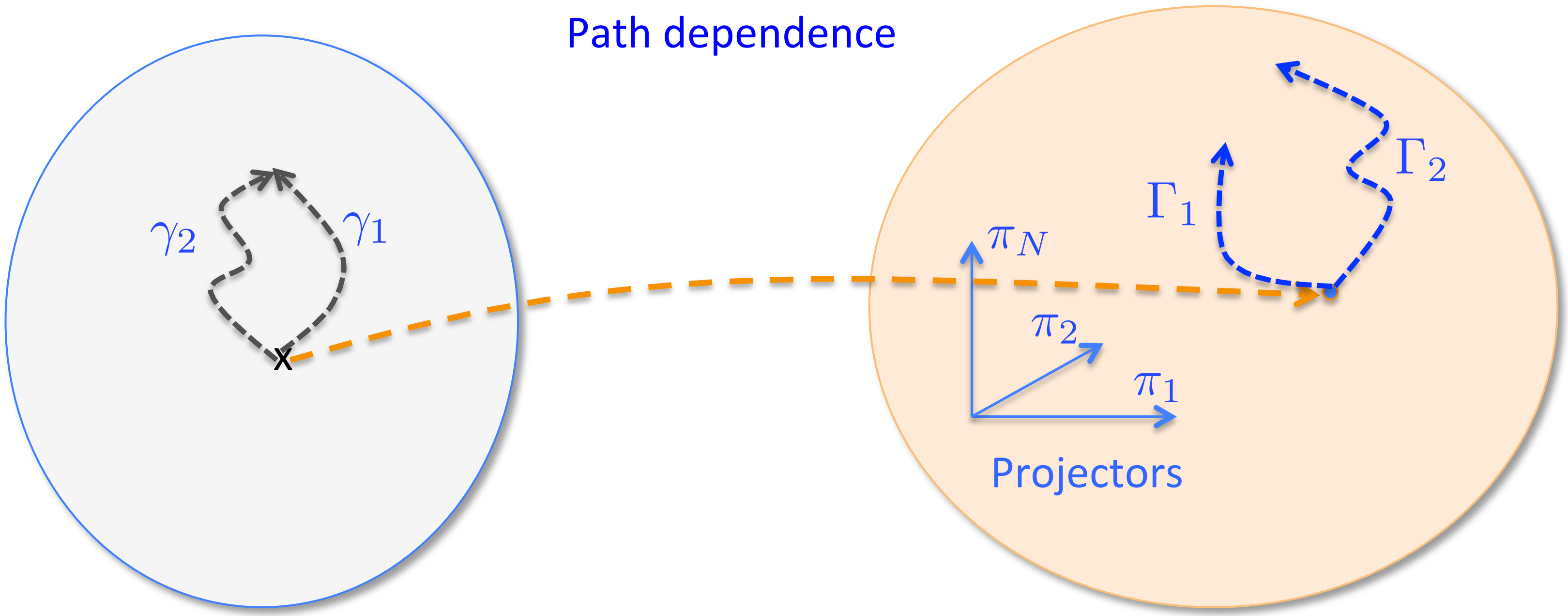
Embedding

Space of **projectors** $\pi_i$

Weighted projectors

# Transboost as **local changes** of representation

...




Space of learning tasks

Target training sets

Embedding

Space of **projectors** $\pi_i$

Weighted projectors

Does the quality of $h_S$ plays a **role**?

# What if ...

Source hypothesis a priori **without relation** to the target task

| | Learning **from target data only** | | TransBoost with "**irrelevant**" 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 | $0.44 \pm 0.02$ | $0.48 \pm 0.02$ | $0.06 \pm 0.02$ | $\mathbf{0.06} \pm 0.02$ |
| 0.005, 0.005, 70 | $0.11 \pm 0.04$ | $0.13 \pm 0.05$ | $0.02 \pm 0.01$ | $\mathbf{0.02} \pm 0.02$ |
| 0.005, 0.005, 70 | $0.10 \pm 0.04$ | $0.11 \pm 0.05$ | $0.01 \pm 0.01$ | $\mathbf{0.01} \pm 0.01$ |
| 0.005, 0.05, 70 | $0.11 \pm 0.04$ | $0.12 \pm 0.05$ | $0.04 \pm 0.02$ | $\mathbf{0.03} \pm 0.01$ |
| Hard: 0.001, 0.001, 70 | $0.42 \pm 0.03$ | $0.48 \pm 0.02$ | $0.33 \pm 0.02$ | $\mathbf{0.37} \pm 0.02$ |
| 0.01, 0.1, 70 | $0.06 \pm 0.03$ | $0.08 \pm 0.03$ | $0.08 \pm 0.02$ | $0.08 \pm 0.02$ |

Very good results!!

$h_S$ **randomly chosen** on the source task $\quad \widehat{R}(h_{\mathcal{S}}) \approx 0.5$

Does the quality of $h_s$ plays a **role**? NO!!

What is the **role** of $h_s$??

# Analysis

- The **quality of the source hypothesis** on the source data?
  - Plays no role

- The **proximity of the source and target** distributions $P_X$ and $P_Y$?
  - Plays no role

## But… !?

=> *No condition on the source!??*

Still some transfer learning problems

appear to us **more easy than others**???

# Interpretation

Transfer acts as a bias and $h_S$ is a strong part of this bias

- **If** the **source hypothesis** is **well chosen**: the **bias** is well informed
  - Which **does not mean** that $h_S$ must be good on the source task
- **Otherwise**: Learning is **badly directed**

  or there is **over-fitting** if the capacity of $h_{\mathcal{S}} \circ \pi$ is too large

# Lessons

- The learning problem now becomes the problem of **choosing** a good set of (weak) projections

- Theoretical guarantees exist

# Analysis

- The **generalization properties** of TransBoost

  can be imported from the ones for **boosting**

$$\mathcal{H}_{\mathcal{T}} \;=\; \left\{ \mathtt{sign}\left[\sum_{n=1}^{N} \alpha_n \, h_{\mathcal{S}} \circ \pi_n\right] \middle| \alpha_n \in \mathbb{R}, \pi_n \in \Pi, n \in [1, N] \right\}$$

$$d_{\mathtt{VC}}(\mathcal{H}_{\mathcal{T}}) \;\leq\; 2(d_{h_{\mathcal{S}} \circ \Pi} + 1)(N+1)\log_2\big((N+1)\,e\big)$$

$$R(h) \;\leq\; \widehat{R}(h) \;+\; \mathcal{O}\left(\sqrt{\frac{d_{h_{\mathcal{S}} \circ \Pi} \;\ln(m_{\mathcal{T}}/d_{h_{\mathcal{S}} \circ \Pi}) + \ln(1/\delta)}{m_{\mathcal{T}}}}\right)$$

# Outline

1. Supervised induction: the classical setting
2. What about Out Of Distribution learning (OOD)?
3. Parallel transport, covariant derivative and transfer learning
   - What they are
   - ... in Machine Learning
4. A way to deal with different spaces of tasks
5. Conclusions

# Conclusions (1)

Transfer learning → mostly heuristical approaches so far

1. **Parallel transport** is a natural way for looking at **transfer** learning
   - The **covariant derivative** is then a measure of difference
     - **How** to compute it?
       - Pioneering works in **computer vision**
     - What about when the **source** and **target** domains are **different**?
       - TransBoost: a **proposal**

2. Transfer learning is **path dependent** in general
   - The study of these path dependencies is **important** ...
     - Curriculum learning
     - Longlife learning
   - ... and a wide **open research question**

# Conclusions (2)

- The **theoretical guarantees** for transfer learning:
  - Do not necessarily depend on the **performance of the source hypothesis** $h_S$

    But depend on the **bias** that $h_S$ determines

  - Involve the **capacity** of the space of **transformations**

    (and **the path** followed between source and target)

Still to be explored

# Bibliography


- Baldock, R., Maennel, H., & Neyshabur, B. (2021). **Deep learning through the lens of example difficulty**. *Advances in Neural Information Processing Systems*, *34*.
- Bauer, M., Klassen, E., Preston, S. C., & Su, Z. (2018). **A diffeomorphism-invariant metric on the space of vector-valued one-forms**. arXiv preprint arXiv: 1812.10867.
- Ben-David, S., Blitzer, J., Crammer, K., Kulesza, A., Pereira, F., & Vaughan, J. W. (2010). A theory of learning from different domains. *Machine learning*, *79*(1-2), 151-175.
- Cornuéjols A., Murena P-A. & Olivier R. "*Transfer Learning by Learning Projections from Target to Source*". Symposium on Intelligent Data Analysis (IDA-2020), April 27-29 2020, Bodenseeforum, Lake Constance, Germany.
- Kuzborskij, I., & Orabona, F. (2013, February). Stability and hypothesis transfer learning. In *International Conference on Machine Learning* (pp. 942-950).
- Mansour, Y., Mohri, M., & Rostamizadeh, A. (2009). Domain adaptation: Learning bounds and algorithms. *arXiv preprint arXiv:0902.3430*.
- Redko, I., Morvant, E., Habrard, A., Sebban, M., & Bennani, Y. (2019). *Advances in Domain Adaptation Theory*. Elsevier.
- Schonsheck, S. C., Dong, B., & Lai, R. (2018). **Parallel transport convolution: A new tool for convolutional neural networks on manifolds**. arXiv preprint arXiv:1805.07857.
- V. Vapnik and A. Vashist (2009) "A new learning paradigm: Learning using privileged information". *Neural Networks*, vol. 22, no. 5, pp. 544–557, 2009
- H. Venkateswara, S. Chakraborty, and S. Panchanathan, "Deep-learning systems for domain adaptation in computer vision: Learning transferable feature representations," *IEEE Signal Processing Magazine*, vol. 34, no. 6, pp. 117–129, 2017.
- Yosinski, J., Clune, J., Bengio, Y., & Lipson, H. (2014). How transferable are features in deep neural networks?. In *Advances in neural information processing systems* (pp. 3320-3328).
- Zhang, C., Zhang, L., & Ye, J. (2012). Generalization bounds for domain adaptation. In *Advances in neural information processing systems* (pp. 3320-3328).

# Bibliography


- Ben-David, S., Lu, T., Luu, T., & Pál, D. (2010). Impossibility theorems for domain adaptation. In *International Conference on Artificial Intelligence and Statistics* (pp. 129-136).
- Ben-David, S., Blitzer, J., Crammer, K., Kulesza, A., Pereira, F., & Vaughan, J. W. (2010). A theory of learning from different domains. *Machine learning*, *79*(1-2), 151-175.
- Cornuéjols A., Murena P-A. & Olivier R. "*Transfer Learning by Learning Projections from Target to Source*". Symposium on Intelligent Data Analysis (IDA-2020), April 27-29 2020, Bodenseeforum, Lake Constance, Germany.
- Kuzborskij, I., & Orabona, F. (2013, February). Stability and hypothesis transfer learning. In *International Conference on Machine Learning* (pp. 942-950).
- Mansour, Y., Mohri, M., & Rostamizadeh, A. (2009). Domain adaptation: Learning bounds and algorithms. *arXiv preprint arXiv: 0902.3430*.
- Redko, I., Morvant, E., Habrard, A., Sebban, M., & Bennani, Y. (2019). *Advances in Domain Adaptation Theory*. Elsevier.
- H. Venkateswara, S. Chakraborty, and S. Panchanathan, "Deep-learning systems for domain adaptation in computer vision: Learning transferable feature representations," *IEEE Signal Processing Magazine*, vol. 34, no. 6, pp. 117–129, 2017.
- Yosinski, J., Clune, J., Bengio, Y., & Lipson, H. (2014). How transferable are features in deep neural networks?. In *Advances in neural information processing systems* (pp. 3320-3328).
- Zhang, C., Zhang, L., & Ye, J. (2012). Generalization bounds for domain adaptation. In *Advances in neural information processing systems* (pp. 3320-3328).