On-line Learning

Antoine Cornuéjols

AgroParisTech – INRA MIA 518

antoine.cornuejols@agroparistech.fr

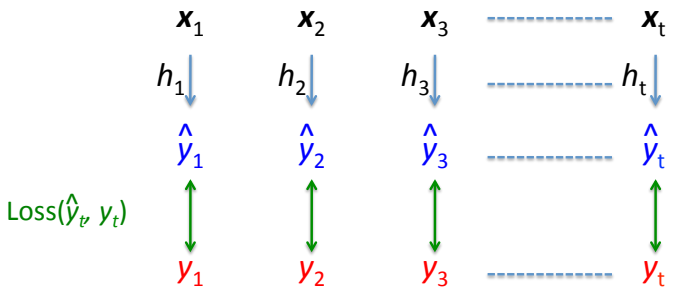
Data streams

Idea of dependencies between the data points (or not)

1. If **one** sequence
   - Internal structure
   - Markov chains, HMM, Grammars, …
   - Regression
   - On-line learning
2. If **a set** of sequences
   - Question of defining a **distance between sequences**
     - Clustering
     - Supervised learning

The online learning scenario

- A stream:



E.g. ***Choice of melons***. I see one, I make a prediction about its tastiness, then I eat it and know the answer.

Motivations

- **Very large training data base** with insufficient resources
  - Still the data is i.i.d.
- « anytime » context: **data stream**
- **Non stationary** environment
  - **Covariate shift**
  - **Concept drift**
- **Domain Adaptation**
- **Transfer** between tasks

Outline

Introduction

What do we want to do?

- **Predict** (time series)
  - **What will happen**: the future of the sequence(s)
  - The **class** of a sequence
    - ***high*** *or* ***low*** *electrical consumption at the end of the day*
    - *"**Bull**" (raising) or "**bear**" (descending) stock exchange price*
- **Answer a sequence of questions** (online learning)

What is special in data streams?

- **Time series**
  - **Dependencies** between the measurements
    - Not i.i.d.
      - Markov chains, HMMs, grammars, …
  - **No** vector space
    - Sequences may have different lengths

What is special in data streams?

- Time series
  - Dependencies between the measurements
    - Not i.i.d.
      - Markov chains, HMMs, grammars, …
  - No vector space
    - Sequences may have different lengths
- **Online learning**
  - Learning and deciding **while receiving new** training and test **instances**
  - Possibility of a **changing environment**

Terminology

- **Time series** vs. **data streams**
  - Time (or neighbor) dependencies
  - Data elements can be i.i.d. (**stationary environment**)
  - Or not i.i.d. (if **non stationary environment**)
- **Batch** vs. **online** or **anytime** learning
  - Only **one pass** at the data (and no direct storage of the data)
- **Incremental** learning
  - Idea of **construction** of the hypothesis
  - Possibly a (well-organized) **sequence** of questions or tasks
- **Online** learning
  - Potentially **against any sequence** of events (questions)
  - **Constant complexity** (and storage space) (or almost)

The question of the evaluation of learning

- Problems
  - Deciding is intermixed with learning
  - The environment may be changing
- Holdout evaluation (standard)
- **Prequential** evaluation (**pre**diction and se**quential**)
  - Aggregation of the **number of errors** of prediction during learning
  - Possibly using a window (if concept drift happens)

A remark

- All **first generation** learning systems were **incremental**
  - Perceptrons, Checker, ARCH, Version Space, ...
- **Most** widely **used** learning algorithms are **batch** learning systems
- This may **change** again
  - **Very large** data base
  - Surge of **data streams** applications
  - Towards **long-life learning**: sequence of tasks, and changing environments

Environnement

non stationnaire

Le cadre classique

Real risk: expected loss

$$R(h) = \mathbb{E}[\ell(h(\boldsymbol{x}), y)] = \int_{\boldsymbol{x}\in\mathcal{X}, y\in\mathcal{Y}} \ell(h(\boldsymbol{x}), y)\, \mathbf{P}_{\mathcal{X}\mathcal{Y}}\, d(\boldsymbol{x}, y)$$

But $\mathbf{P}_{\mathcal{X}\mathcal{Y}}$ is unknown, then use: $\mathcal{S}_m = \{(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_m, y_m)\} \in (\mathcal{X} \times \mathcal{Y})^m$

Empirical risk Minimization

$$\hat{h} = \underset{h\in\mathcal{H}}{\text{ArgMin}}\left[R_m(h)\right] + \text{Reg}\right] = \underset{h\in\mathcal{H}}{\text{ArgMin}}\left[\frac{1}{m}\sum_{i=1}^{m}\ell(h(\boldsymbol{x}_i), y_i)\right] + \lambda\,\text{Capacity}(\mathcal{H})\right]$$

1. All examples are equal: **no forgetting**
2. Commutative criterion: **no information from the sequence**

Le cadre classique

- Qu'est-ce qui permet la « généralisation » ou l'induction ?

- Link between the past and the future:
  distributions $P_{\mathcal{X}}$ et $P_{\mathcal{Y}|\mathcal{X}}$ are supposed stationnary
- I.i.d. data

But … the world is constantly evolving

- New types of data
  - Data are made available through *unlimited streams* that continuously flow, possibly at high-speed
  - The underlying *regularities may evolve over time* rather than be stationary
  - The data is now often *spatially as well as time situated*

➡ The data can **no longer** be considered as **independently** and **identically distributed**

Nouveaux types de contextes

- Ressources limitées
- Apprentissage sur flux de données : contrainte « anytime »
- Covariate shift :
  - Distribution $\mathbf{P}_{Y|X}$ stationnaire
  - Distributions $\mathbf{P}_X$ différentes
- Apprentissage actif
- Dérive de concept (« concept drift ») : $\mathbf{P}_{Y|X}$ **non** stationnaire
- Apprentissage par transfert d'une tâche à une autre
- Apprentissage tutoré : quel curriculum ?

On-line adaptation

- *Assumption*: the current hypothesis h_t is **somewhat relevant** to label x_{t+1}.
  - A kind of **transfer** between successive "tasks"

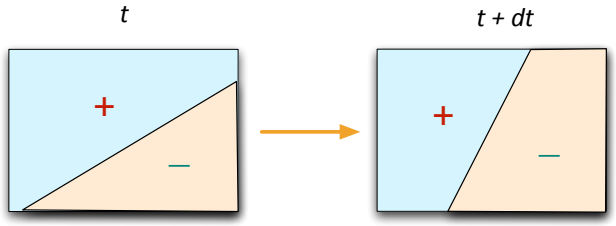
➢ How one should **control** and tune this **transfer**?
  - What should be the weight of the past?
  - The **plasticity** vs. **stability** dilemma

Apprentissage en-ligne

contre **toute** séquence

L'apprentissage « en-ligne »

- Le scénario
  - (X_{t-n}, Y_{t-n}), …, (X_{t-2}, Y_{t-2}), (X_{t-1}, Y_{t-1}), (X_t, Y_t) … (X_{t+1}, Y_{t+1}) ?



- Mesure de performance
  - Σ erreurs ; Moyenne erreurs

L'apprentissage « en-ligne »

- Apprentissage contre toute séquence
  - Plus d'hypothèse de stationnarité
  - Ni d'aucune régularité temporelle
- Comment savoir si l'algorithme est bon ?
- Idée de comité d'« experts » (*N* experts)

Utilisation d'un **comité d'experts**

	Expert_1	Expert_2	Expert_3	Expert_4	Expert_5	Expert_6
t_1	1	0	0	1	0	1
t_2	1	1	1	0	0	0
t_3	1	0	0	0	1	1
t_4	1	0	0	1	1	1
t_5	1	1	0	1	1	1
t_6	1	0	0	0	1	0

Quel algorithme de choix à chaque instant *t* ?

Algorithme de sélection d'expert

- Choix d'**un expert a priori sans changement**
  - Propriétés ?
    - Possibilité de perte ∞

Peut-on faire mieux ?

Utilisation d'un **comité d'experts**

	Expert_1	Expert_2	Expert_3	Expert_4	Expert_5	Expert_6
t_1	1	0	0	1	0	1
t_2						
t_3						
t_4						
t_5						
t_6						

Utilisation d'un **comité d'experts**

	Expert_1	Expert_2	Expert_3	Expert_4	Expert_5	Expert_6
t_1	1	0	0	1	0	1
t_2	1	1	1	0	0	0
t_3						
t_4						
t_5						
t_6						

Utilisation d'un **comité d'experts**

	Expert_1	Expert_2	Expert_3	Expert_4	Expert_5	Expert_6
t_1	1	0	0	1	0	1
t_2	1	1	1	0	0	0
t_3	1	0	0	0	1	1
t_4						
t_5						
t_6						

Algorithme glouton déterministe

Algorithme de sélection d'expert

- Algorithme **glouton déterministe**
  - ***Propriétés*** ?
    - Peut être très bon
    - **Pire cas ?**

Algo glouton déterministe : **pire cas**

	Expert_1	Expert_2	Expert_3	Expert_4	Expert_5	Expert_6
J_1	1	0	0	0	0	0
J_2	0	1	0	0	0	0
J_3	0	0	1	0	0	0
J_4	0	0	0	1	0	0
J_5	0	0	0	0	1	0
J_6	0	0	0	0	0	1

$$L \leq N(L^*) + N - 1$$

Perte algo — Perte meilleur expert

Apprentissage **en-ligne**

- Pourquoi comparer avec le meilleur expert ?
  - Notion de « regret »
- Pourquoi pas avec le meilleur algorithme possible ?

Algorithme de sélection d'expert

- Algorithme **glouton aléatoire**
  - ***Propriétés*** ?
    - Peut être très bon
    - **Pire cas ?**

$$L_{RG} \leq (\ln N + 1)(L^*) + \ln N$$

Le « cas **réalisable** »

- Classification à 2 classes
- ∃ un expert *i* inconnu ne faisant jamais d'erreur : $h_{i,t}(x_t) = y_t \quad \forall t$
- Quelle stratégie ?
  - On assigne **un poids** $w_t = 1$ à tous les experts
  - À chaque *t*
    - Prédire la classe majoritaire dans le vote : $H(x_t)$
    - Comparer la prédiction $h_{i,t}(x_t)$ avec y_t
    - Assigner $w_t = 0$ à tous les experts ayant fait une erreur

$$L_{CR} \leq \lfloor \log_2 N \rfloor$$

Cas **réalisable** : preuve

- Initialement : $W_0 = N$
- À chaque étape : $W_t \leq W_{t-1}/2$

$$L_{CR} \leq \lfloor \log_2 N \rfloor$$

Le cas **non** réalisable

- À t=0, $W_0 = N$
- À chaque t : $w_i(t) = \begin{cases} w_i(t) & \text{si } y(t) = h_{i,t}(\mathbf{x}_t) \\ \beta\, w_i(t) & \text{si } y(t) \neq h_{i,t}(\mathbf{x}_t) \end{cases}$

$w_{i,t} =$

$$W(t) \leq W(t-1)/2 + \beta W(t-1)/2$$

$$W(t) \leq W_0 \frac{(1+\beta)^t}{2^t} \quad \text{et} \quad W(t) \geq \beta^{L^*(t)}$$

$$\beta^{L^*(t)} \leq W_0 (1+\beta)^t / 2^m$$

$$L_{CR} \leq \left\lfloor \frac{\log_2 N + L^* \log_2(1/\beta)}{\log_2 \frac{2}{1+\beta}} \right\rfloor$$

Another perspective on the problem

- At each time step, there exists a distribution $\mathbf{P}_t$ over the space H of hypotheses
- At each round of learning:
  - Receive instance $x_t \in X$
  - Choose h_t randomly according to the current distribution $\mathbf{P}_t$ over H
  - Predict $\hat{y}_t = h_t(x_t)$
  - Receive the true label y_t
  - Computes the new distribution $\mathbf{P}_{t+1}$ using the **Multiplicative Weight algorithm**

$$\mathbf{P}_{t+1} = \frac{\mathbf{P}_t(h)}{Z_t} \times \begin{cases} e^{(-\eta)} & \text{if } h(\mathbf{x}_t) \neq y_t \\ 1 & \text{otherwise} \end{cases}$$

The multiplicative weight technique

- Provides theorems of the form:
  bound on the **learner's** cumulative loss in terms of
  the cumulative loss of the **best strategy in hindsight**
  + an **additional term** which can be shown to be relatively insignificant for large T

Bilan sur ce type d'analyse

- Permet d'obtenir des **théorèmes** !!
- Mais **trop exigeant** et peu réaliste
- Idée intéressante : **comité d'experts**
  et **poids multiplicatifs**

Apprentissage en-ligne :

Approches **heuristiques**

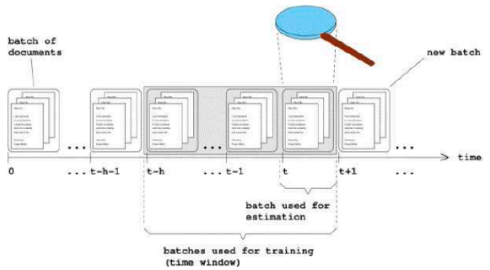
Concept drift

Dérive de concept (« concept drift »)

Drift of $\mathbf{P}_{\mathcal{X}|\mathcal{Y}}$

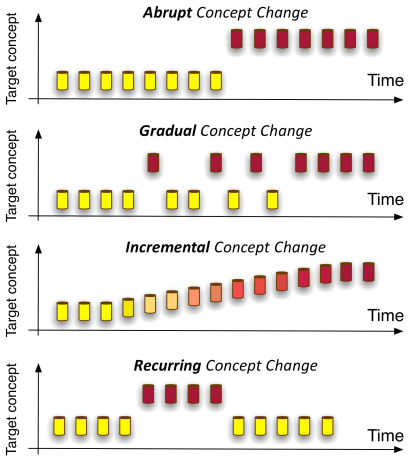
Exemples :

- Profiles of customers (purchases function of *income, age, ...*)
- Document filtering function of the interests of the user



Changement de concept

Types de changements de concepts



Desirable properties of a system that handle concept drift

- **Adapt** to concept drift **as soon as possible**

- Distinguish **noise** from **true changes**
  - Robust to noise but adaptive to changes

- Recognize and react to **recurring contexts**

- Adapt with **limited resources** (time and memory)

Dérive de concept : comment la **détecter** ?

Adapting to the Change

- **ADWIN** (average value in windows of training data)

[A. Bifet. Adaptive learning and mining for data streams and frequent patterns. ACM SIGKDD Explorations Newsletter, 11(1):55–56, 2009.]

- **DDM** (monitor the number of errors)

[J. Gama, P. Medas, G. Castillo, and P. Rodrigues. Learning with drift detection. In Advances in Artificial Intelligence–SBIA 2004, pages 286–295. Springer, 2004.]

- **EDDM** (monitor the distance between errors)

[M. Baena-García, J. del Campo-Ávila, R. Fidalgo, A. Bifet, R. Gavaldà, and R. Morales-Bueno. Early drift detection method. Fourth International Workshop on Knowledge Discovery from Data Streams, 2006.]

Changement de concept

- … à nouveau le problème du contrôle de la mémoire

Le dilemme ***plasticité-stabilité***

Dérive de concept

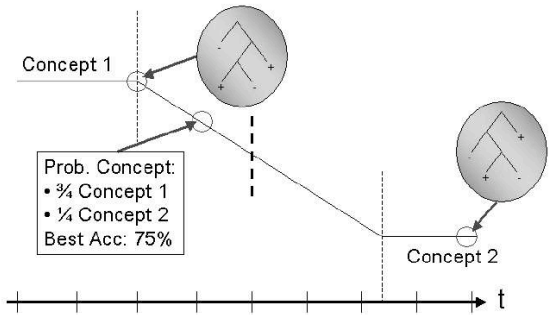
- Problèmes
  - **Apprendre le modèle le plus précis** possible
    - *Longue mémoire*
  - **Être réactif** mais en résistant au bruit
    - *Oublier rapidement*

Dilemme stabilité-plasticité

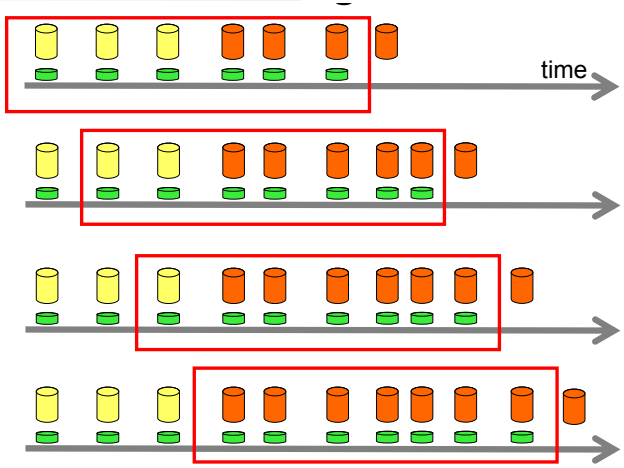
- **Approches**
  1. ***Fenêtre** glissante*
  2. ***Poids** sur les données*
  3. *Méthodes **d'ensemble***

Changement de concept

- Exemple



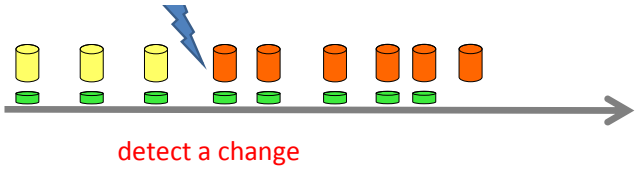
Fixed sliding windows



Fixed sliding windows

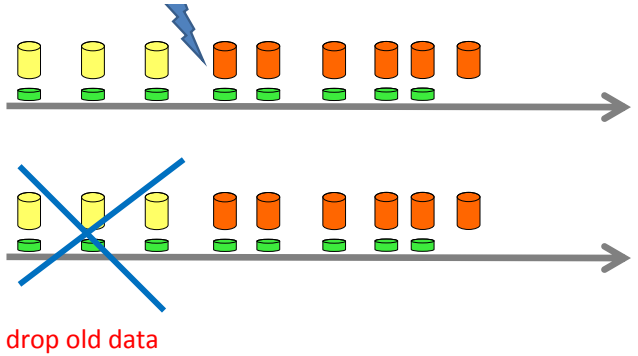
- How to choose the size
  - **Small** window size
    - Fast **adaptability**
    - **Less** precision
  - **Large** window size
    - **Good and stable learning results** if the environment is stationary
    - Does **not react quickly** to concept changes

Variable training windows

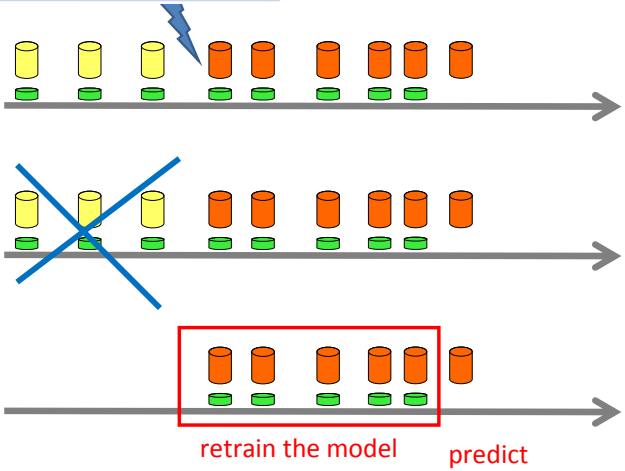


- Problem: how to detect a “true” change?

Variable training windows



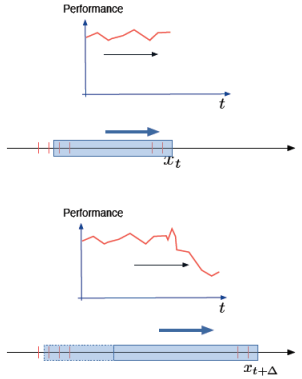
Variable training windows



- Pb: how to select the right window size?

Concept drift: **adaptive sliding windows**

Principle:

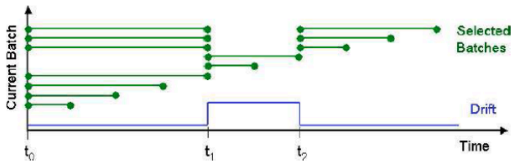


WK96 G. Widmer and M. Kubat (1996) *“Learning in the presence of concept drift ans hidden contexts”* Machine Learning 23: 69–101, 1996.

Concept drift: **sliding windows**

One (among many) method for the selection of windows

- Learn a classifier on the last batch
- Test it on every preceding windows
- Keep the windows where error < ε



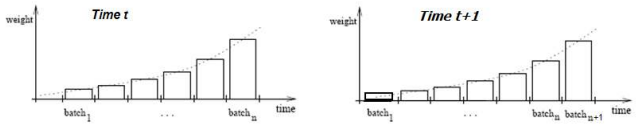
SK M. Scholz and R. Klinkenberg (1996) *"Boosting classifiers for drifting concepts"* Intelligent Data Analysis (IDA) Journal, Volume 11, Number 1, March 2007.

Instance weighting methods

- Examples are weighted depending on their age or relevance regarding the current concept
  - Store in memory sufficient statistics over all examples
- Recent examples are given more weight than past ones
  - Often an exponential weighting mechanism is used

    => decide on a decay factor λ

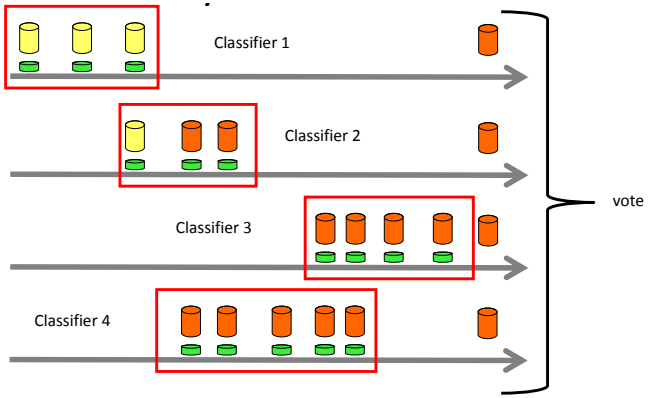
$$w(\mathbf{x}) = e^{-\lambda t_{\mathbf{x}}}$$



Dérive de concept : **méthodes d'ensemble**

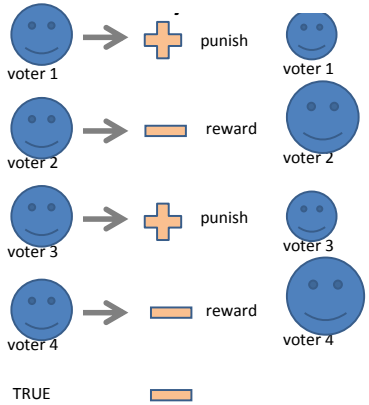
- Apprendre des **experts** sur des **fenêtres différentes**
- **Pondérer** les experts en fonction de leur performance (récente)
- **Remplacer** les plus mauvais experts

Ensemble methods

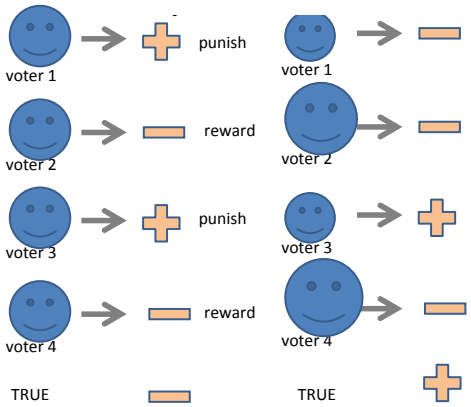


Ensemble methods

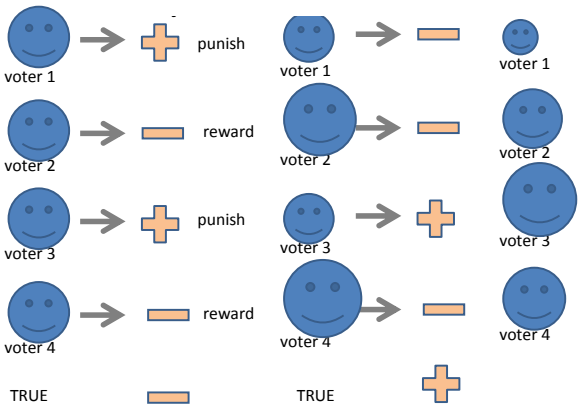
Ensemble methods



Ensemble methods



Ensemble methods



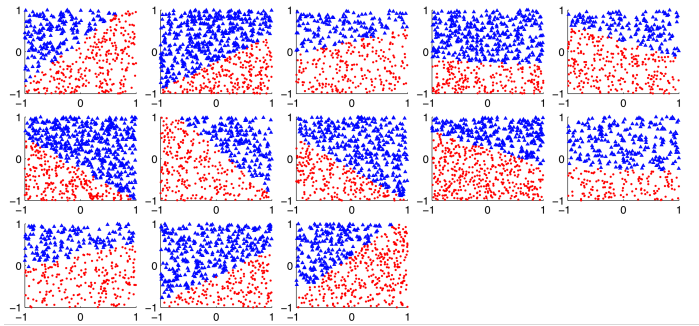
Concept drift: **ensemble methods**

Dynamic weighted majority

- Classifiers in ensemble have initially a weight of 1
- For each new instance:
  - If a ***classifier predicts incorrectly***, reduce its weight
  - If ***weight drops below threshold***, remove classifier
  - If ***ensemble*** then ***predicts incorrectly***, install new classifier
  - Finally, all classifiers are (incrementally) updated by considering new instance

KM03 Kolter, Maloof (2003) *"Dynamic weighted majority: a new ensemble method for tracking concept drift"* ICDM 2003, 123-130.

Concept drift: **ensemble methods**



The DACC system [Jaber et al., 2013]

Ensemble methods:

Diverse learners, multiple concept descriptions

- High diversity: reduces initial drop in accuracy just after drift *(plasticity)*
- Low diversity: more accurate results for *stability*

Explicit / Implicit memory management:

- Different memory sizes
- Remove outdated learners
- Add new learners with no memory of the past

[Ghazal Jaber, Ph.D thesis, 2013.
« *An approach for online learning in the presence of concept changes* »]

The DACC system [Jaber et al., 2013]

- Assumes that the environment is **piecewise stationary**
- Maintains **a set of base learners**
  - Each base learner **learns on-line** from the stream of examples
  - Every τ time steps, the larners are evaluated on a window size of t_{eval}, and a randomly chosen learner from the worst learners (worst half) is **removed** from the set
  - A **new base learner** is created and cannot be removed before t_{mat} steps
- At each time step the overall **decision** is taken by:
  - A **combination function of the individual predictions**: a majority vote or other types of combinations
  - Where the weight of a base learner depends on its measured prediction performance

[Ghazal Jaber, Ph.D thesis, 2013.
« *An approach for online learning in the presence of concept changes* »]

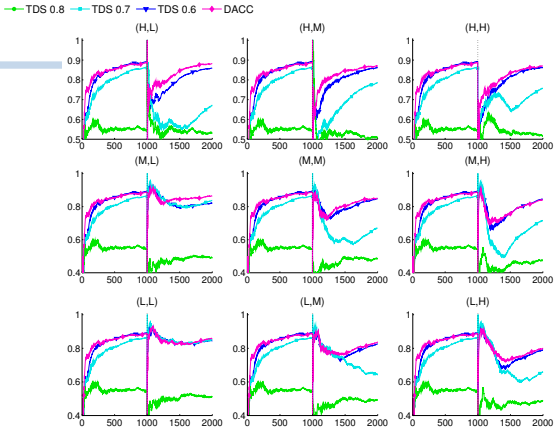
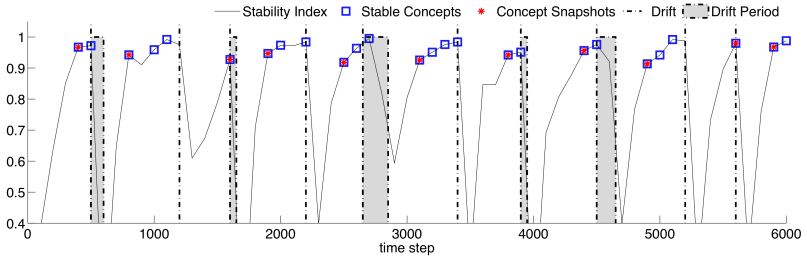


FIGURE 4.24: DACC vs TDS: The online performance on the databases of the *SineH* problem. The experts are lossless decision trees and $N = 20$. The x-axis represents the training examples (time step) and the y-axis represents the online classification performance (the percentage of correctly classified instances so far). The online performance is reset at time step 1,000 when the concept starts changing.

[Ghazal Jaber, Ph.D thesis, 2013.
« *An approach for online learning in the presence of concept changes* »]

From adaptation to anticipation: ADACC

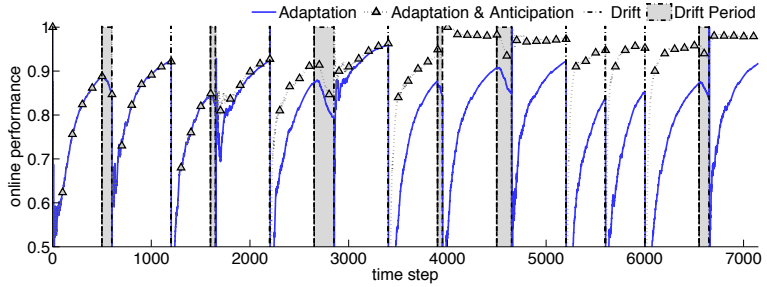


Stream		Adaptation			Anticipation	Total gain		Dues to anticipation		Due to recurring	
name	size	base learner	mean	std-dev	predictor	mean	std-dev	mean	std-dev	mean	std-dev
10-D Low	7,150	perceptron	**112.7**	4.6	Elman net	**1.0**	0.0	0.0	0.0	1.0	0.0
10-D Med.	7,150	perceptron	**826.7**	36.1	Elman net	**311.8**	33.9	60.6	23.2	251.2	20.6
10-D High	7,150	perceptron	**904.8**	33.4	Elman net	**345.4**	25.2	87.8	26.4	257.6	13.3
Robot	753	decision tree	**43.0**	2.6	-	**9.0**	1.9	-	-	9.0	1.9

Table 1: Summary of the experiments and the measured gains in prediction errors wrt. an adaptive only strategy.

G. Jaber, A. Cornuéjols, and P. Tarroux, "**Anticipative and Dynamic Adaptation to Concept Changes,**" in *Proc. ECML-PKDD-2013* (Workshop ``Real-World Challenges for Data Stream Mining''), Prague, Czech Republic, 2013.

From adaptation to anticipation



The drifting concept is a 10D linear separator.

G. Jaber, A. Cornuéjols & Ph. Tarroux (2013) "*Anticipative and adaptive adaptation to concept changes*". Submitted to IJCAI-2013.

Approches heuristiques de l'apprentissage en-ligne : **bilan**

- Efficaces dans certaines situations
- Demandent le réglage de paramètres
- En plein essor

- Manque de fondations théoriques solides

A problem when **studying a new problem**

- Lack of agreed benchmark data bases
- Real data often difficult to get due to privacy or proprietary reasons
- Therefore forced to rely on controlled "artificial" data
  - Often cause for bad reviews in papers

Au-delà de

l'Apprentissage en-ligne

Tracking: an intriguing idea

[Richard Sutton, Anna Koop & David Silver (2007). *On the role of tracking in stationary environments*. ICML-2007]

Even in stationary environments, it **can be advantageous to act as if the environment was changing**!!!

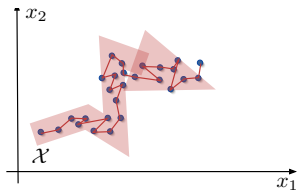
Tracking: an intriguing idea

In a lot of natural settings:

- Data comes *sequentially*
- *Temporal consistency*: consecutive data points come from "similar" distribution: not i.i.d.

This enables:

- Powerful learning
- with limited resources (time + memory)



SKS:07 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 in stationary environments

- A toy environment

$$y_t = \frac{1}{1 + e^{-w_t x_t}}$$

$$w_{t+1} = w_t + \alpha (z_t - y_t) x_t$$

Figure 1. The Black and White world. The agent follows a random walk right and left, occasionally observing the color above it. The states wrap.

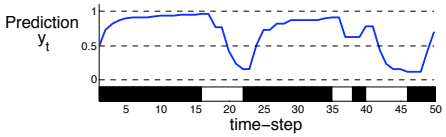
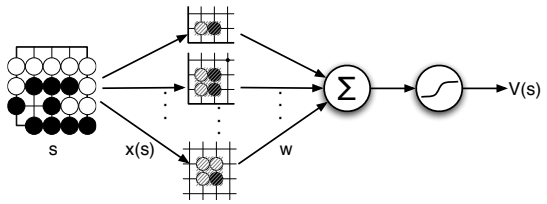


Figure 2. A sample trajectory in the Black and White world, showing the prediction on each time-step and the actual color above the agent. The prediction is modified only on time steps on which the color is observed. Here $\alpha = 2$.

Tracking in stationary environments

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)



Tracking in stationary environments

Tracking to play Go

- Comparison:
  - learn **a general evaluation function** V(s)
    - On 250,000 complete episodes of self-play
  - Learn **successive evaluation functions** Vt(s) attuned to the current states
    - On 10,000 episodes of self-play starting from the current position

Features	Tracking beats converging		
	Black	White	Total
1×1	82%	43%	62.5%
2×2	90%	71%	80.5%
3×3	93%	80%	86.5%

Table 1. Percentage of 5×5 Go games won by the tracking agent playing against the converging agent when playing as Black (first to move) and as White.

Tracking in stationary environments

- Comparison:
  - **learn a general evaluation function** V(s)
    - On 250,000 complete episodes of self-play
  - Learn **successive evaluation functions** Vt(s) attuned to the current state
    - On 10,000 episodes of self-play starting from the current position

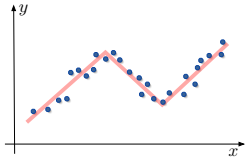
Features	Total features	CPU (minutes)	
		Tracking	Converging
1×1	75	3.5	10.1
2×2	1371	5.7	13.8
3×3	178518	9.1	22.2

Table 2. Memory and CPU requirements for tracking and converging agents. The total number of binary features indicates the memory consumption. The CPU time is the average training time required to play a complete game: 250,000 episodes of training for the converging agent; 10,000 episodes of training per move for the tracking agent.

Tracking: an intriguing idea

Assumptions:

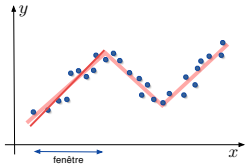
- Data streams
- *Temporal consistency*: consecutive data points come from "similar" distribution: not i.i.d.
- Limited resources: Restricted hypothesis space $\mathcal{H}$



"Local" learning

and local prediction :

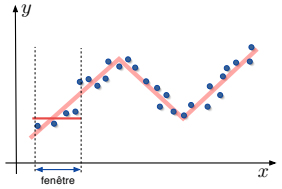
$$L_t = \ell(h_t(\boldsymbol{x}_t), y_t) = \ell(h_t(\boldsymbol{x}_t), f(x_t, \theta_t))$$



Tracking: an intriguing idea

Assumptions:

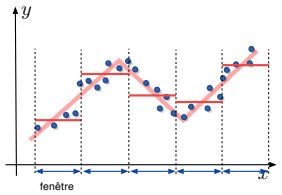
- Data streams
- *Temporal consistency*: consecutive data points come from "similar" distribution: not i.i.d.
- Limited resources: Restricted hypothesis space $\mathcal{H}$



"Local" learning

and local prediction :

$$L_t = \ell(h_t(\boldsymbol{x}_t), y_t) = \ell(h_t(\boldsymbol{x}_t), f(x_t, \theta_t))$$

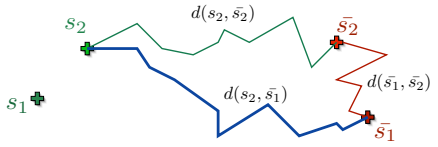


Adaptation de domaine

et **Transfert**

Adaptation de domaine et transfert entre tâches

- Un petit exemple
  - Sous quelles conditions le transfert est avantageux ?
  - Et quand vaut-il mieux repartir de zéro ?



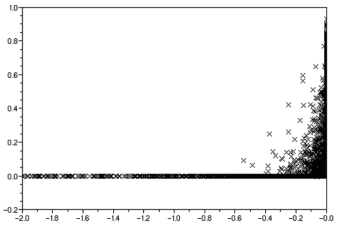


FIG. 5 – *Gain obtenu par l'adaptation d'AdaptiveA** en fonction de $\frac{d(s_2,\bar{s}_1)-d(s_2,\bar{s}_2)-d(\bar{s}_1,\bar{s}_2)}{d(s_2,\bar{s}_2)}$.

L. Fedon & A. Cornuéjols (2008) « *Comment optimiser A* adaptatif* ». Proc. of RFIA-08.

Le transfert de représentations

- Exemple des Réseaux de Neurones profonds
  - Transfert de **couches** d'une tâche à une autre
    - **Reconnaissance de l'écriture** manuscrite d'une langue à une autre (même très différente. E.g. de l'anglais à l'arabe)

L'adaptation de domaine

- **Comment apprendre** à partir d'une distribution source (apprentissage)un **classifieur** performant sur une distribution différente (prédiction) ?

Exemple



?

Personne pas de Personne Y-a-t'il une Personne ?

Corpus de **Photos étiquetées** Corpus de **Videos non étiquetées**

Domaine **source** Domaine **cible**

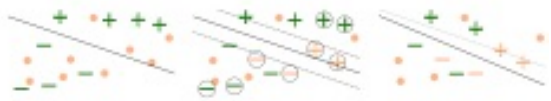
(from [Émilie Morvant Thèse, sept. 2013, « *Apprentissage de vote de majorité pour la classification supervisée et l'adaptation de domaine : approches PAC-Bayésiennes et combinaison de similarités* »])

L'adaptation de domaine

- Trois grandes approches

Repondérations des données
[Huang et al., 2007, Jiang and Zhai, 2007, Mansour et al., 2009b]

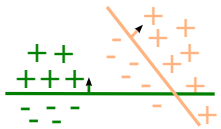
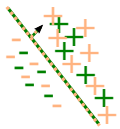
Auto-étiquetage
[Bruzzone and Marconcini, 2010]



Recherche d'un espace de représentation commun
[Blitzer et al., 2006, Blitzer et al., 2011, Chen et al., 2011, Daumé III, 2007, Daumé III et al., 2010]

L'adaptation de domaine

Question : étant donné que ***h*** est apprise sur le domaine source, quelle sera sa **performance** sur le domaine cible ?

Intuitivement : cela dépend de la **proximité** entre domaine source et domaine cible

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 :

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

$R_{D_S}(h)$: 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

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

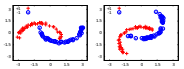
ou $\nu = R_{D_T}(h_T^*) + R_{D_S}(h_S^*)$,
$h_{\mathcal{X}}^*$ est la meilleure hypothèse sur le domaine $\mathcal{X}$ [Mansour et al., 2009a]

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

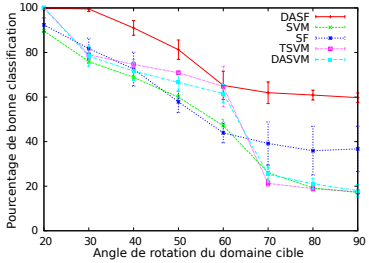
L'adaptation de domaine

- Solution d'Émilie Morvant [2013]

Problème jouet : "lunes jumelles"



- 1 domaine source
- 8 domaines cibles selon 8 angles de rotations
- 10 tirages aléatoire pour chaque angle
- Performances sur 1500 exemples cibles
- Noyau gaussien ou renormalisation (non SDP, non symétrique) du noyau



Taille	10	10	9	8	4	4	4	3

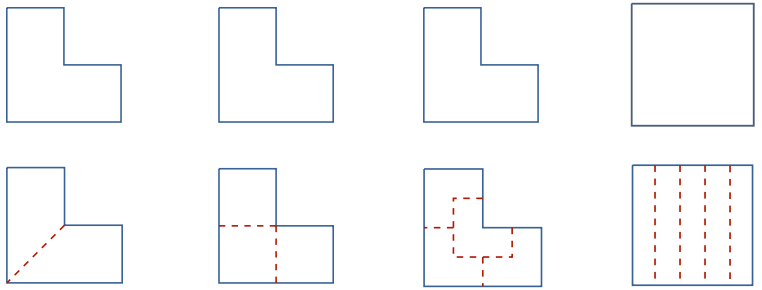
Sequencing effects

A fundamental question

Effets de séquences

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

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



Effets de séquences

- Comment les éliminer ?
- Comment les construire et en tirer avantage ?
- Comment bâtir un curriculum à usage des machines ?

Conclusions

Apprentissage à partir de flux de données

- Apprentissage en une passe (« à la volée »)
  - **Complexité** constante
- Adaptation continue avec contrainte « anytime »
  - Apprentissage « **incrémental** »
- Éventuellement avec adaptation à un environnement non stationnaire
  - **Gestion de l'oubli** encore plus importante

Bilan

- Domaine encore assez neuf pour l'AA
  - Peu focalisé sur la prédiction de la suite d'une séquence
- Source de questions intéressantes
  - Données **non vectorielles**
  - Données **non i.i.d.**
  - Apprentissage **en-ligne**
    - Algorithmes efficaces
    - Contrôle de la mémoire
    - Adaptation de l'hypothèse et de $\mathcal{H}$ en-ligne
  - Lié à problèmes généraux
    - **Transfert** entre tâches

Nouvelles questions

- Quelle mémoire du passé ?