

On-line learning

Where are we so far?

Antoine Cornuéjols

MMIP, AgroParisTech, Paris

Int. workshop on Data Stream Management and Mining

Beijing, October 27th, 2008

“Incremental learning”: a new topic?

The first learning algorithms were all incremental:

- Perceptron [Rosenblatt, 1957-1962]
- CHECKER [Samuel, 1959]
- ARCH [Winston, 1970]
- Version Space [Mitchell, 1978, 1982], ...

However, **most existing learning algorithms are not!**

- C4.5 / Regression trees / ...
- SVM / Neural Networks / ...
- ILP systems / Grammatical inference / ...
- ...

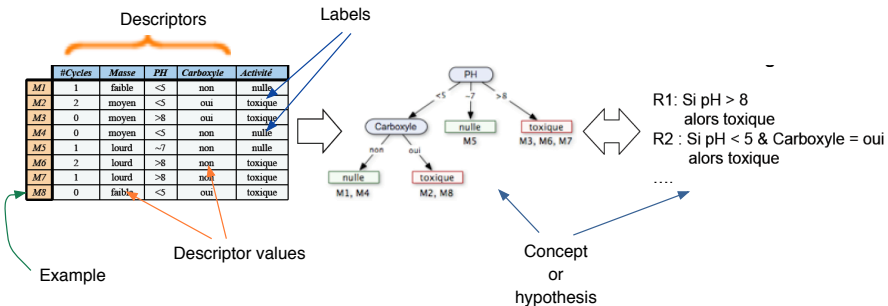
Outline

- 1 Introduction
- 2 On-line learning in a static world
- 3 On-line learning in a world in movement
- 4 Focus on changes, not on pictures
- 5 Conclusions

Outline

- 1 Introduction
 - The standard setting: one-shot and i.i.d.
 - On-line learning: a renewed interest
- 2 On-line learning in a static world
- 3 On-line learning in a world in movement
- 4 Focus on changes, not on pictures
- 5 Conclusions

Illustration

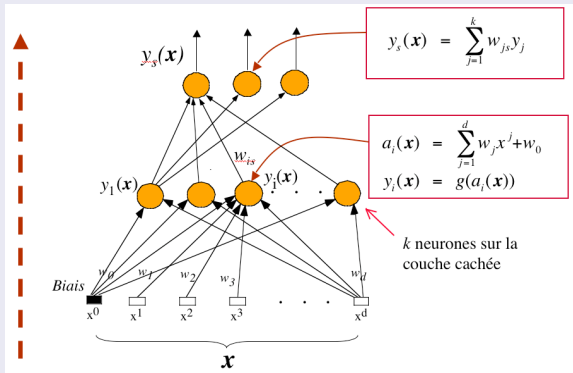


Choix de \mathcal{H}

Fonctions de décision à base de dictionnaire

- $h(\mathbf{x}, \mathbf{w}) = \sum_{i=1}^n w_i g_i(\mathbf{x}) + w_0$
- where the $g_i(\mathbf{x})$ are the **basis functions**

Exemple : Multi-Layer Perceptron

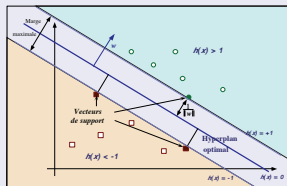


Choix de \mathcal{H}

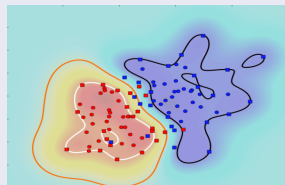
The Support Vector Machines: SVM

$$h^*(\mathbf{x}) = (\mathbf{w}^* \cdot \mathbf{x}) + w_0^* = \sum_{i=1}^m \alpha_i^* u_i \cdot \langle \phi(\mathbf{x}_i), \phi(\mathbf{x}) \rangle + w_0^*$$

SVM : redescription space $\Phi(\mathcal{X})$

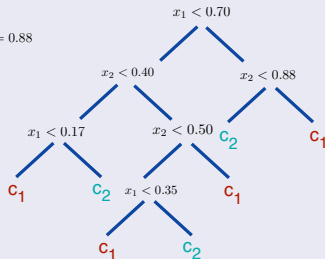
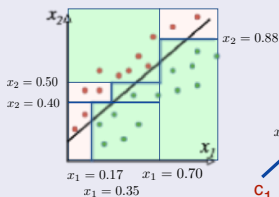


SVM : initial space \mathcal{X}



Decision Trees

Example :Decision Trees



The standard setting

Learning algorithms geared to the analysis of large data bases

- **Stationary and identical distribution** for learning and test
- **i.i.d. assumption** (independently and identically distributed)

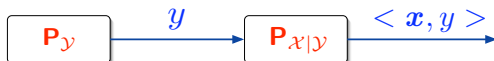


Figure: Generative process for the examples.

Almost correct prediction (most of the time) (PAC)

$$L(h) = P_{xy}\{h(x) \neq y\}$$

The standard setting

Optimizing the expected risk

Real risk: expected loss

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

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

Empirical risk Minimization

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

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

Key idea n°1:

Essential character of the inductive criterion

- **Expresses the problem:**
 - Cost of misclassification
 - Global measure of performance taken as a substitute for the real one
- Allows us to **analyze** the conditions for a successful induction
- Did **motivate** most modern learners (SVM, Boosting, ...)

The paradigm ... and its limits

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

Measuring and controlling a world in movement

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 ***independent and identically distributed***

On-line learning: why bother?

A wealth of new applications

- 1 **Limited resources:**
 - Learning from very large data bases (e.g. Telecoms: millions of examples ; EGEE: billions of examples, ...)
- 2 **“Anytime” constraints:** Data streaming
- 3 **Covariate shift:** stationary target concept but changing distribution
- 4 **Active learning**
- 5 **Concept drift**
- 6 **Transfer learning** from one task to another
- 7 **Tutored learning** with a professor

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

Outline

- 1 Introduction
- 2 **On-line learning in a static world**
 - Computational issues
 - The example of stochastic gradient approaches
 - Existing incremental algorithms: the heuristic approach
 - Data streams
- 3 On-line learning in a world in movement
- 4 Focus on changes, not on pictures
- 5 Conclusions

Empirical incremental learning

Lots of empirical and heuristic techniques

to solve computational and on-line issues

- **Reduce computational cost**
- **Heuristic approaches**
 - *k*-nearest neighbors
 - Decision Trees

- **Issues raised**

Stochastic gradient vs. total gradient

Total gradient

$$\hat{h} = \underset{h \in \mathcal{H}}{\text{ArgMin}} R_m(h) = \underset{h \in \mathcal{H}}{\text{ArgMin}} \frac{1}{m} \sum_{i=1}^m \ell(h(\mathbf{x}_i), y_i)$$

Total gradient

$$\begin{aligned} h_t &= h_{t-1} - \Phi_t \frac{\partial R_m(h_{t-1})}{\partial h} \\ &= h_{t-1} - \Phi_t \frac{1}{m} \frac{\partial}{\partial h} \sum_{i=1}^m \ell(h_{t-1}(\mathbf{x}_i), y_i) \end{aligned}$$

Linear convergence towards the optimum \hat{h} de $R_m(h)$: $(h_t - \hat{h})^2$ converges as e^{-t} .

Stochastic gradient vs. total gradient

Stochastic gradient

$$\hat{h} = \underset{h \in \mathcal{H}}{\text{ArgMin}} R_m(h) = \underset{h \in \mathcal{H}}{\text{ArgMin}} \frac{1}{m} \sum_{i=1}^m \ell(h(\mathbf{x}_i), y_i)$$

Stochastic gradient

$$h_t = h_{t-1} - \frac{1}{t} \Phi_t \frac{\partial L}{\partial h}(h_{t-1}(\mathbf{x}_t), y_t)$$

- Converges slowly towards a local optimum of $R_m(h)$: $(h_t - \hat{h})^2$ converges as $\frac{1}{t}$.
- In fact, converges quickly towards the region of the optimum but slowly then because of the noisy (stochastic) gradient.

Much **simpler** than batch

Stochastic gradient vs. total gradient

Computational complexity

- **Batch**
 - Store N examples
 - Gradient in $\mathcal{O}(N)$ operations
- **On-line**
 - Must memorize the “sufficient past” (in h_t)
 - Gradient in $\mathcal{O}(1)$ operations

Approximation

- **Batch**
 - Converges towards $h^* = \text{ArgMin}_{h \in \mathcal{H}} R(h)$ with approximation $\mathcal{O}(1/t)$
- **On-line**
 - Converges towards $h^* = \text{ArgMin}_{h \in \mathcal{H}} R(h)$ with approximation $\mathcal{O}(1/t)$
 - **But can consider more examples !!**
($\mathcal{O}(N \log N)$ instead of N for batch)

Key idea n^o2:

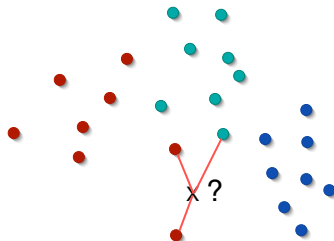
On-line learning is simpler (less costly)

... which can lead to better learning performance!

Incremental learning

Illustration

Nearest neighbors

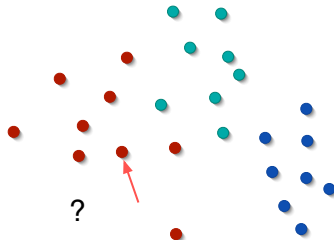


- Simple algorithm (“*lazy learning*”)
- Order independent
- But growing computational cost (time and space): $\mathcal{O}(m)$

Incremental learning

Illustration

Nearest neighbors (2) with limited memory



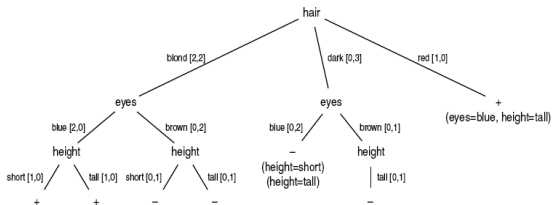
- Selection of “prototypes”
 - Eliminate the outlier
 - Eliminate the most ancien
 - Compute and keep center of gravity with the closest point
 - ...
- Order dependent

Incremental learning

Illustration

Incremental induction of decision trees (ID5R)

class	height	hair	eyes
-	short	blond	brown
-	tall	dark	brown
+	tall	blond	blue
-	tall	dark	blue
-	short	dark	blue
+	tall	red	blue
-	tall	blond	brown
+	short	blond	blue



- Actually memorizes all examples
- Order independent
- But computational time at each step : $\mathcal{O}(m \cdot d \cdot b^d)$

UTG89

Paul Utgoff (1989) "Incremental Induction of Decision Trees" Machine Learning Journal, vol.4, No.2, 161-186

Incremental learning

Assessment

Numerous heuristic algorithms

Motivations

- Computational constraints (e.g. constant time)
- Time series

New questions

- What to keep in memory?
- Sequence effects
 - How to reduce them?
 - (How to use the information in the sequence?)

Data streams

A first illustrative example

Paul presents a permutation of the n first integers, one by one, with one integer missing

E.g.: $n = 10$ 3, 7, 6, 8, 5, 1, 9, 2

Carol must identify the missing number.

Caveat : She cannot store all the numbers seen so far (which would take $\mathcal{O}(n \log n)$ digits).

Solution:

- **Carol** adds all the numbers seen so far ($\mathcal{O}(\log n)$ digits and computational cost $\mathcal{O}(\log n)$)
- and she subtracts that sum from $\frac{n \cdot (n+1)}{2}$

Total cost : space $\mathcal{O}(\log n)$ digits ; computation $\mathcal{O} \log n$

Data streams

A first illustrative example

What if **Paul** shows all but two numbers?

E.g.: $n = 10$ 3, 7, 6, 5, 1, 9, 2

Solution:

- **Carol** keeps the **sum** and **sum of squares** of numbers seen so far ($\mathcal{O}(n \log n)$ digits and computational cost $\mathcal{O}(n \log n)$)

- $\sum_{i=1}^9 x_i = \frac{n \cdot (n+1)}{2} = 9 \cdot 10/2 = 45$
- $\sum_{i=1}^9 x_i^2 = \frac{n(n+1)(2n+1)}{6} = \frac{9 \cdot 10 \cdot 19}{6} = 285$
- $3 + 7 + 6 + 5 + 1 + 1 + 9 + 2 = 33$
- $9 + 49 + 36 + 25 + 1 + 81 + 4 = 205$
- $x_1 + x_2 = 12$ et $x_1^2 + x_2^2 = 80$
- d'où $x_1 = 4$ et $x_2 = 8$

Data streams

Lessons

Lessons:

The solution relies on keeping **summaries** of the data

Here, the solution is exact. Most of the time, it can only be **approximate**

Data streams

Challenges

- **Not enough space** to store all the stream data
- Hence, impossible to rescan the whole data set
- Need to **adapt to changing data distribution**
- Processing should be **as fast as possible**

Data streaming is an incremental process

(i.e. new iterations are built based on previous ones)

Data streams

Frequently used learners

Desirable properties

- Low runtime complexity
- **Inherently incremental**

Frequently used learners

- Decision trees (VFDT, adaptive VFDT)
- Rule learners
- SVM (Support Vector Machines)

- **Naïve Bayes**
- **Instance-based learners** (e.g. k -nearest neighbors)

Outline

- 1 Introduction
- 2 On-line learning in a static world
- 3 On-line learning in a world in movement**
 - Issues: Computational constraints and non i.i.d. data
 - Covariate shift
 - Non stationary environment
 - Concept drift
- 4 Focus on changes, not on pictures
- 5 Conclusions

On-line learning: the issues

Computational constraints

Possible in principle with standard (one-shot and i.i.d.) approach, but too costly computationally

→ **Reduce time and space complexity**

- *Stochastic gradient / incremental learning*

Stationary environment but changing distribution + anytime constraints

Not i.i.d.

→ **Anticipate and take advantage of sequence information**

- *covariate shift / transductive learning / tracking*

Changing environment

Not i.i.d. + Non stationary

→ **Anticipate and take advantage of sequence information**

- *concept drift*
- *transfer learning / tutored learning*

Covariate shift

Definition

Changing \mathbf{P}_x

Examples:

- **Non stationary input data** (\mathbf{P}_x changes but not $\mathbf{P}_{y|x}$)
 - Medicine: seasonal variations
 - Spam filtering (adaptation to a new user)
- **Bias in the selection process** in learning
 - Artificially balanced training data (but not in test)
 - Active learning
 - Interpolation vs. extrapolation (in regression)

Covariate shift

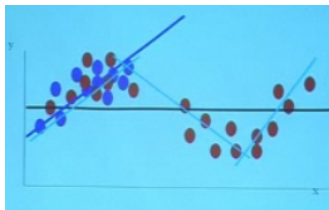
Why is it a problem?

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 $\mathbf{P}'_{\mathcal{X}}$



Covariate shift

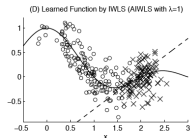
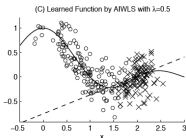
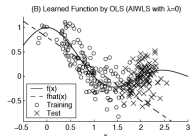
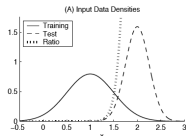
Approaches

“Importance weighted” inductive criterion

Principle : weighting the classical ERM

$$R_{Cov}(h) = \frac{1}{m} \sum_{i=1}^m \left(\frac{P_{\mathcal{X}'}(x_i)}{P_{\mathcal{X}}(x_i)} \right)^\lambda (h(x_i) - y_i)^2$$

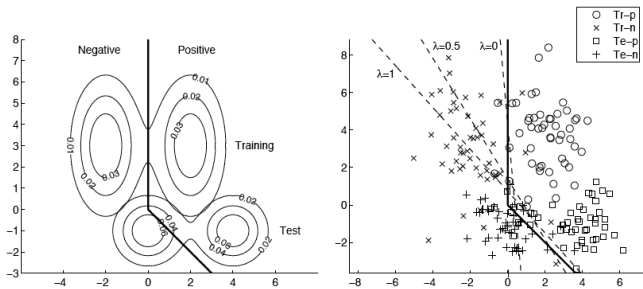
λ controls the
stability /
consistency
(absence of bias)



Covariate shift

Approaches

“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 'x' 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.

Covariate shift

Approaches

“Importance weighted” inductive criterion

How to get $\frac{P_{\mathcal{X}'}(x_i)}{P_{\mathcal{X}}(x_i)}$?

Empirical estimation

Semi-supervised learning

On-line learning: the issues

Computational constraints

Possible in principle with standard (one-shot and i.i.d.) approach, but too costly computationally

→ **Reduce time and space complexity**

- *Stochastic gradient / incremental learning*

Stationary environment but changing distribution + anytime constraints

Not i.i.d.

→ **Anticipate and take advantage of sequence information**

- *covariate shift / transductive learning / tracking*

Changing environment

Not i.i.d. + Non stationary

→ **Anticipate and take advantage of sequence information**

- *concept drift*
- *transfer learning / tutored learning*

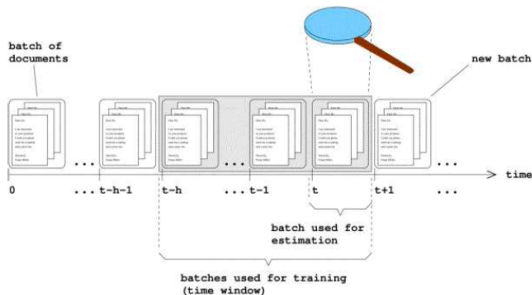
Concept drift

Definition

Drift of $P_{x|y}$

Exemples :

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



Concept drift

Definition

Drift of $P_{y|x}$

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

Problems:

- **Detecting variations** but be robust to noise
- Follow the evolutions but stay robust: **Control forgetting**

- The oldest the data, the more likely they are obsolete

But:

- The larger the training set, the better the generalization

Heuristic approaches

- **Window based approaches.** *Problem: control their size*
- **Weighting the examples** with respect to time. *Problem: control their weights*

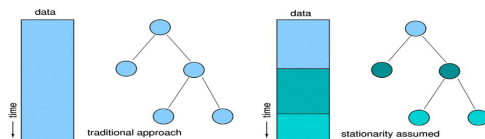
Key idea n°3:

Control the memory of the past

Concept drift

The decision tree approach: VFDT

- Stream Mining: VFDT = **Very Fast Decision Tree** Learner



- Key question: How much data is needed to safely induce node?
- Use Hoeffding-bound: estimate the nb of examples necessary to acknowledge $IG(X_{\text{best}}) - IG(X_{2\text{nd best}}) > \epsilon$, with prob. $1 - \delta$

DH00

Domingos, Hulten (2000) "Mining high-speed data streams" KDD 2000, 71-80.

Concept drift

Concept adapting VFDT

- VFDT assumes stationarity
- If no stationarity: node may violate $IG(X_{\text{best}}) - IG(X_{2\text{nd best}}) > \epsilon$
- If drift suspected:
 - start learning alternative subtree
 - exchange subtree if more accurate than current subtree
- Fast adaptation by forgetting
 - maintain window in memory to correct sufficient statistics at nodes
 - but window of fixed size (forgetting is not adaptive)

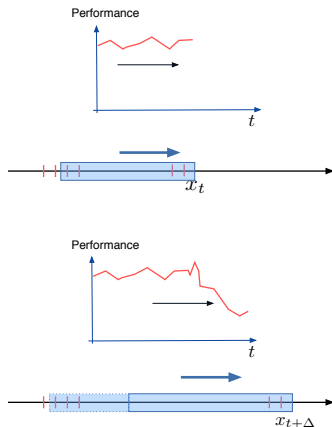
HSD01

Hulten, Spencer, Domingos (2001) "Mining time changing data streams" KDD 2001, 97-106.

Concept drift

Sliding window approach

Principle:



WK96

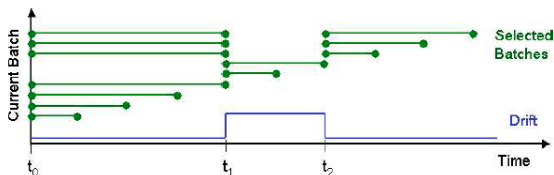
G. Widmer and M. Kubat (1996) "Learning in the presence of concept drift and hidden contexts" Machine Learning 23: 69–101, 1996.

Concept drift

Sliding window approach

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 $< \varepsilon$



SK

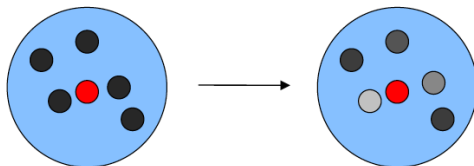
M. Scholz and R. Klinkenberg (1996) "Boosting classifiers for drifting concepts" Intelligent Data Analysis (IDA) Journal, Volume 11, Number 1, March 2007.

Concept drift

Locally weighted forgetting

Forget *redundant* data first

- All data starts with weight 1
- Weights of k nearest neighbors are adjusted
 - The closer the data to the new sample, the more the weight is decayed
 - If weight drops below some threshold, remove data



Concept drift

Ensemble methods

- Learn a number of models on different parts of the data
- Weigh classifiers according to recent performance
- If classifier performance degrades, replace it by a new classifier

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

Accuracy weighted majority

- Divide streams into chunks
- Learn new classifier from n such chunks and keep the k top performing classifiers
- Use most recent chunk to estimate expected accuracy of each classifier
- Weigh classifiers in the ensemble by expected accuracy (Results are provably better than by averaging decisions)

WFYH03

Wang, Fan, Yu, Han (2003) "Mining concept-drifting data streams using ensemble classifiers" KDD 2003, 226-235.

Concept drift

A Boosting approach

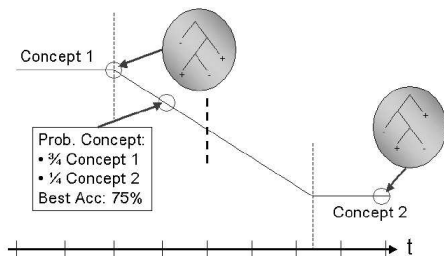


Fig. 4. Continuous concept drift, starting with a pure *Concept 1* and ending with a pure *Concept 2*. In between, the target distribution is a probabilistic mixture. It is optimal to predict *Concept 1* before the dotted line, and *Concept 2*, afterwards.

How to learn concept 2 **before** the end of the transition ?

SK

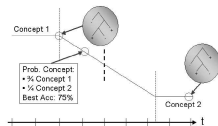
M. Scholz and R. Klinkenberg (1996) "Boosting classifiers for drifting concepts" Intelligent Data Analysis (IDA) Journal, Volume 11, Number 1, March 2007.

Concept drift

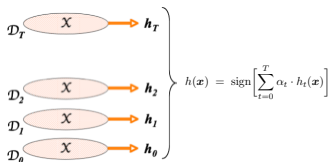
A Boosting approach

Principle :
gradually modify the distribution of the examples

by “substracting” the distribution associated with concept 1.



$$\forall (x, y) \in \mathcal{X} \times \mathcal{Y} : P_{D'}(x, y) = P_D(x, y) \cdot \frac{P_D[h_{t-1}(x) = \hat{y}] \cdot P_D[y = y^*]}{P_D[h_{t-1}(x) = \hat{y}, y = y^*]}$$



Concept drift

Assessment

Heuristics

- Efficient in their respective application domains
- Require a fine tuning
- Not easily transferable to other domains
- Lack theoretical foundations

Theoretical analyses

What are the conditions for PAC learning with an error of ε ?

- Depends upon $d_{\mathcal{H}}$ and the speed of the drift v (measured as the prob. that 2 subsequent concepts disagree on a randomly drawn example)
- Possible if $v = \mathcal{O}(\varepsilon^2/d_{\mathcal{H}}^2 \ln \frac{1}{\varepsilon})$ (usually impractically large)
- Rk: adversary protocol

Concept drift

... further issues

Desirable to:

- *Recognize and treat recurring contexts*
 - E.g. seasonal variations
 - → quickly recover old models if appropriate
- *Provide insights about change*
 - **Trends**, "Second derivative"
 - **Interpretability**: "What has changed and how?"
 - By comparison with the last model(s): **discover interesting new knowledge**

Useful properties:

- **keeping models**
- Having a **model for change** vs. a model for current context

Concept drift

... further issues

References:

- CMM05 [Catania, Maddalena, Mazza \(2005\)](#) "PSYCHO: A prototype system for pattern management" Proc. Int. Conf. on Very Large Data Bases, 2005, 1346-1349.
- GGR99 [Ganti, Gehrke, Ramakrishnan \(2005\)](#) "A framework for measuring changes in data characteristics" Proc. 18th Symp. on Principles of Database Systems, 1999, 126-137.
- GGR01 [Ganti, Gehrke, Ramakrishnan \(2005\)](#) "DEMON: Mining and monitoring evolving data" IEEE Trans. Knowledge Data Eng., 2001, 13, 50-63.
- HB07 [Höppner, Bötcher \(2007\)](#) "Matching partitions over time to reliably capture local clusters in noisy domains" Proc. Int. Conf. Principles and Practice of Knowledge Discovery in Databases PKDD (2007), 479-486.
- LT08 [Liu, Tuzhilin \(2008\)](#) "Managing large collections of data mining models" Communications of the ACM, 2008, 51, 85-89.
- MZ05 [Mei, Zhai \(2005\)](#) "Discovering evolutionary theme patterns from text: an exploration of temporal text mining" Proc. of KDD 2005, 198-207.
- WZFY03 [Wang, Zhou, Fu, Yu \(2003\)](#) "Mining changes of classification by correspondence tracing" SIAM Int. Conf. on Data Mining, 2003, 95-106.

Key idea n^o4:

Focus on the changes themselves

... and not only on models of the current context

Outline

- 1 Introduction
- 2 On-line learning in a static world
- 3 On-line learning in a world in movement
- 4 Focus on changes, not on pictures**
 - Definition
 - Analysis
 - A new inductive problem
 - Transfert
 - Apprentissage guidé
- 5 Conclusions

Tracking

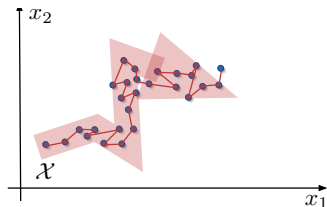
Motivation

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

Characteristics

If “temporal consistency” holds ...

... enormous advantage for learning

1 Less computational cost

- *Time*: take into account fewer examples at each time step
- *Space*: does not store every past examples

2 Intrinsically **on-line** and adapted to **non stationary environments**

But can we:

- 1 **Formalize** this?
- 2 **Measure** this advantage?
- 3 Turn this into a **learning strategy**?

Tracking

Analysis

A fundamental tradeoff

Temporal Consistency

i.i.d. data

Small memory
Simple \mathcal{H}



Large memory
"Complex" \mathcal{H}

Tracking

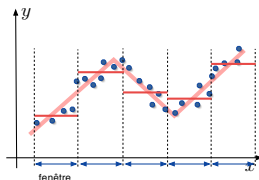
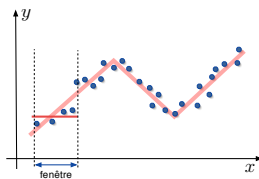
A new inductive problem

Notion of *temporal consistency*

$f(\cdot, \theta_t)$ continuous
and with bounded variation / θ_t

New inductive criterion

$$L_{\langle 0, T \rangle}(r) = \sum_{t=0}^T \ell(h_t(\mathbf{x}_t), y_t) \\ + \lambda \sum \|h_t - h_{t-1}\|^2 \\ + \text{Capacity}(\mathcal{R})$$



Do not optimize the choice of ONE h any longer!!

but optimize the learning rule ($r \in \mathcal{R}$) instead: $(h_{t-1}, \mathbf{x}_t) \xrightarrow{r} h_t$!!

Tracking

Issues in want of answers

New inductive criterion

$$L_{\langle 0, T \rangle} = \sum_{t=0}^T \ell(h_t(\mathbf{x}_t), y_t) + \underbrace{\lambda \sum \|h_t - h_{t-1}\|^2 + \text{Capacity}(\mathcal{R})}_{\text{new criterion}}$$

How to find a good learning rule $r \in \mathcal{R}$?

rule complexity

(memory + complexity)



**Local complexity
of target function**

Control of bias-variance (overfitting)

Transfert

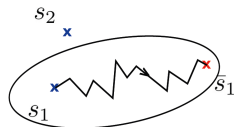
Définition

Ré-utilisation de connaissances acquises dans un contexte (pour résoudre une tâche) dans un autre contexte (pour résoudre une autre tâche)

Éventuellement changement de domaine

Exemples :

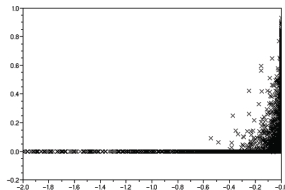
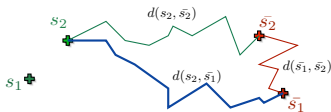
- Adaptive A* / Planification
- Règles apprises pour classer des factures utilisées pour classer des réclamations
- Analogie : $abc \rightarrow abd : ijk \rightarrow ?$



Transfert

Questions

Sous quelles conditions le transfert est avantageux ?
(et quand vaut-il mieux repartir de 0 ?)



FC08

L. Fedon et A. Cornuéjols (2008) "Comment optimiser A* adaptatif" Proc. of RFIA-08, 2008.

Apprentissage guidé

Choix des exemples effectué par un professeur

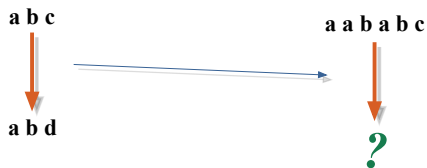
Questions

- Comment fournir **la séquence** d'exemples la plus efficace
 - pour un apprenant
 - dans un but donné

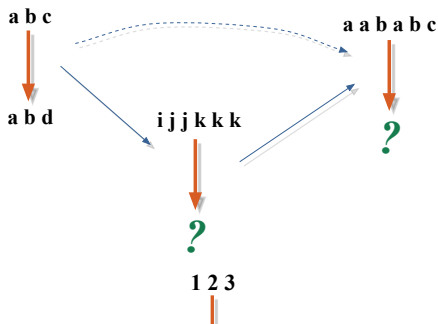
Les effets de séquences

Exemple

[Sur 24 étudiants de DEA, 1996]



- Long et difficile
- Grande variété de réponses



- Beaucoup plus rapide
- Spectre de réponses beaucoup plus serré

Questions

Effets de l'ordre

- Qu'est-ce qui les caractérise ?
 - Optimalité
 - Oubli du passé
- Comment les éliminer ?
- Comment les utiliser ?
- Que révèlent-ils sur la notion d'information transmise d'un état au suivant ?

Quelle **mémoire du passé** ? Que peut-on réutiliser ? Doit-on réutiliser ?

Nouveau critère inductif ?

Outline

- 1 Introduction
- 2 On-line learning in a static world
- 3 On-line learning in a world in movement
- 4 Focus on changes, not on pictures
- 5 Conclusions**

Conclusions

Emerging applications **can not** be solved within the classical setting

- **non i.i.d. data**: the sequence conveys information
- Learning is a **limited rationality activity**

Lots of open questions

- **How to deal with non i.i.d. data**
 - What to memorize? / What to forget?
 - How to cope with or take advantage of **ordering effects**?
 - How to **facilitate future learning**: change representations, ...?
- **What should the inductive criterion be?**
 - How to take the **computational resources** into the inductive criterion?
 - Optimize $h \in \mathcal{H}$ or $r \in \mathcal{R}$?

Already a growing body of works

- Covariate shift, transduction, concept drift, tracking ...
- **Transfer between tasks** / **Teachability**

A mature science of on-line learning is in demand