Classification supervisée et moindres généralisés

Fabien Torre

GRAppA & Mostrare

Jeudi 6 mai 2010
département STID - Lille 2

Parcours

Formation

- étude universitaire à Orsay (Paris XI) ;
- thèse en apprentissage automatique au LRI de Paris XI :
  - programmation logique inductive ;
  - apprentissage disjonctif.

Responsabilités de 2000 à 2008

- service MCF ;
- annuaires d'anciens étudiants ;
- élu deux ans au conseil d'administration de Lille 3 ;
- commission de spécialistes 27 ;
- responsable du Master GIDE ;
- participation au Master ID ;
- trois ans chargé de cours dans Master Recherche de Lille 1

Équipe projet Mostrare

Modeling Tree Structures, Machine Learning, and Information Extraction.

Objectifs

- automatiser le traitement de documents semi-structurés ;
- apprendre à partir d'exemples ;
- n'utiliser que la structure, pas le contenu textuel.

Applications

- classer des documents ou des nœuds de documents ;
- annoter des parties de documents ;
- en extraire certaines parties ;
- transformer des documents d'un format à l'autre.

Angles d'attaque (personnels)

Se ramener à de la classification

- classer ;
- annoter : les annotations deviennent des classes ;
- extraire : deux classes (`à extraire` / `pas à extraire`) ;
- (transformer)

Structures potentiellement manipulées

- vecteurs : feuilles, nœuds et arbres peuvent être décrits par un ensemble fixe d'attributs ;
- séquences : une feuille ou un nœud par le chemin depuis la racine, un arbre par un parcours des nœuds ;
- arbres : le document XML lui-même ;
- graphes : l'arbre enrichi d'axes XPath ou de relations d'égalité.

Classification supervisée : points à définir

1. Ce que l'on veut prédire : des classes binaires $\mathcal{Y} = \{+1, -1\}$;
2. modalité d'obtention des exemples : un échantillon d'exemples étiquetés est disponible pour l'apprentissage, pas de valeur manquante, pas de bruit ;
3. mode d'évaluation des prédictions : erreur mesurée sur un échantillon de test, compréhensibilité ;
4. nature des exemples, nature des classifieurs : dépendent du choix de la représentation des objets à classer $\mathcal{X}$, des hypothèses $\mathcal{H}$: $h \in \mathcal{H} : \mathcal{X} \to \mathcal{Y}$;
5. relation de subsomption entre hypothèses : $\succeq$ selon $\mathcal{H}$;
6. méthode d'apprentissage : méthodes génériques à base de *moindre généralisé*.

Vecteurs en attributs-valeurs

Les exemples sont des vecteurs de valeurs discrètes ou continues :

ClumpThickness	5	5
UniformitySize	1	3
UniformityShape	1	3
MarginalAdhesion	1	3
BareNuclei	1	3
BlandChromatin	3	4
NormalNucleoli	1	4
Mitoses	1	1
Conclusion	**B**	**M**

Domaine attributs-valeurs.

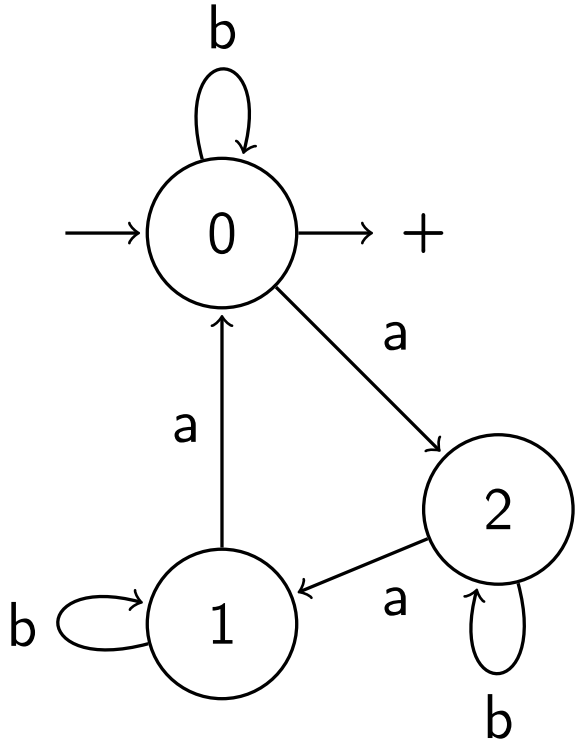
if ClumpThickness $\leq$ 6.0
then class B `[weight=0.07]`

if MarginalAdhesion $\leq$ 8.0
then class M `[weight=0.05]`

Séquences et IG

Les exemples sont des séquences :

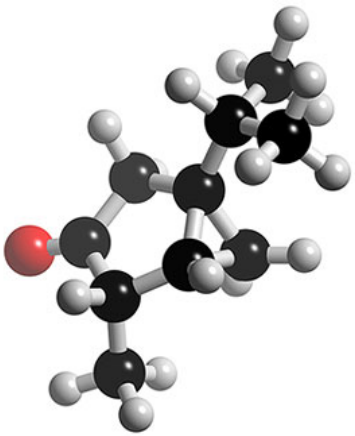
- $+$: ϵ, *aaa*, *b*, *abbaa*, *aaaaaa*
- $-$: *bba*, *aab*, *bbaababa*



Inférence Grammaticale.

Graphes et ILP

Les exemples sont des graphes :



active(M) $\leftarrow$ atom(M,A1,carbon,22),
atom(M,A2,carbon,10),
bond(M,A1,A2,1).

Programmation Logique Inductive.

Plan

Motivations et Intuitions

Une devinette d'abord...

Difficultés (pour les arbres de décision sur *tic-tac-toe*)

- tester plusieurs attributs à la fois ;
- capturer les problèmes disjonctifs.

x	x	x	x	o	o	x	o	o	positive
x	x	x	o	o	b	b	b	b	positive
x	x	x	o	x	o	x	o	o	positive
x	x	x	?	?	?	?	?	?	positive

... et sur des attributs continus ?

5	1	1	1	2	1	3	1	benign
5	4	4	5	7	10	3	2	benign
[5 ;5]	[1 ;4]	[1 ;4]	[1 ;5]	[2 ;7]	[1 ;10]	[3 ;3]	[1 ;2]	benign
6	8	8	1	3	4	3	7	benign
[5 ;6]	[1 ;8]	[1 ;8]	[1 ;5]	[2 ;7]	[1 ;10]	[3 ;3]	[1 ;7]	benign

Plus formellement

Définition : moindre généralisé

Étant donné un ensemble d'exemples $E \subseteq \mathcal{X}$, une hypothèse $h \in \mathcal{H}$ est dite *moindre généralisée* de E si et seulement si :

- $\forall e \in E : h \succeq e$;
- il n'existe pas h' vérifiant $\forall e \in E : h' \succeq e$ et $h \succeq h'$.

Cette définition n'implique pas l'unicité.

Soit $\mathcal{E} = \mathbb{R}^2$, pour chaque $\mathcal{H}$ possible : est-ce que le moindre généralisé est unique ? comment le calculer ?
Candidats : rectangles, carrés, cercles, etc.

Deux profils

Deux vues algorithmiques :

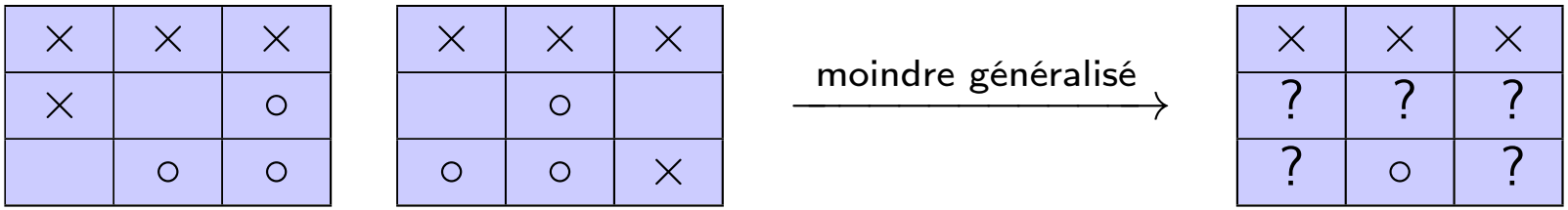
- $\texttt{MG}(e_1, e_2, \ldots, e_n \in \mathcal{X})$ returns $h \in \mathcal{H}$;
- $\texttt{MG}(h_{n-1}, e_n)$ returns $h \in \mathcal{H}$.

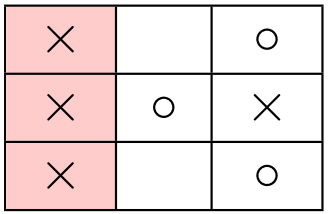
on préfère la deuxième version, plus efficace pour l'apprentissage.

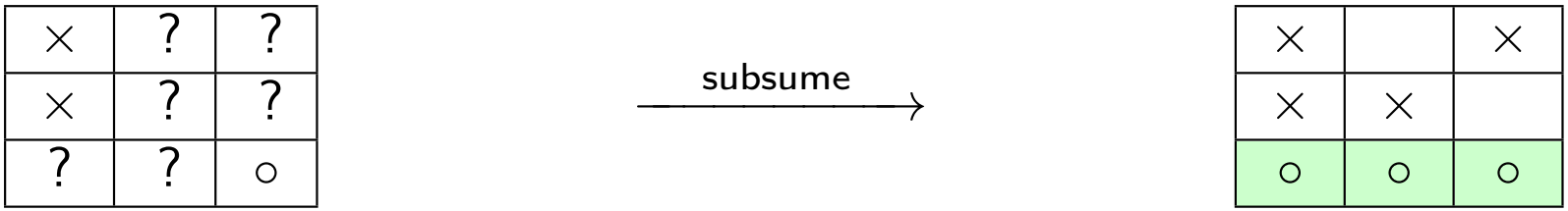
La suite : utiliser les classes !

Correction et disjonctif : intuitions

Généralisation d'exemples d'une même classe sans couvrir aucun exemple d'une autre classe.







Calcul [Torre, 1999]

une graine et sa classe	exemples de la même classe	généralisation maximalement correcte
x_1, y_1	$x_5\ x_8\ \not{x_2}\ x_{14} \ldots$	$g_1 = \mathrm{MG}(\{x_1, x_5, x_8, x_{14}, \ldots\})$ $g_1 \rightarrow y_1$
x_2, y_2	$\not{x_4}\ x_3\ \not{x_{10}}\ x_{12} \ldots$	$g_2 = \mathrm{MG}(\{x_2, x_3, x_{12}, \ldots\})$ $g_2 \rightarrow y_2$

- Exemple : âge $\in [25, 40] \rightarrow$ positif ;
- instable : dépend de la graine et de l'ordre des exemples ;
- pour un exemple donné, un moindre généralisé correct conclut sur une unique classe (-1 ou $+1$) ou s'abstient (0).

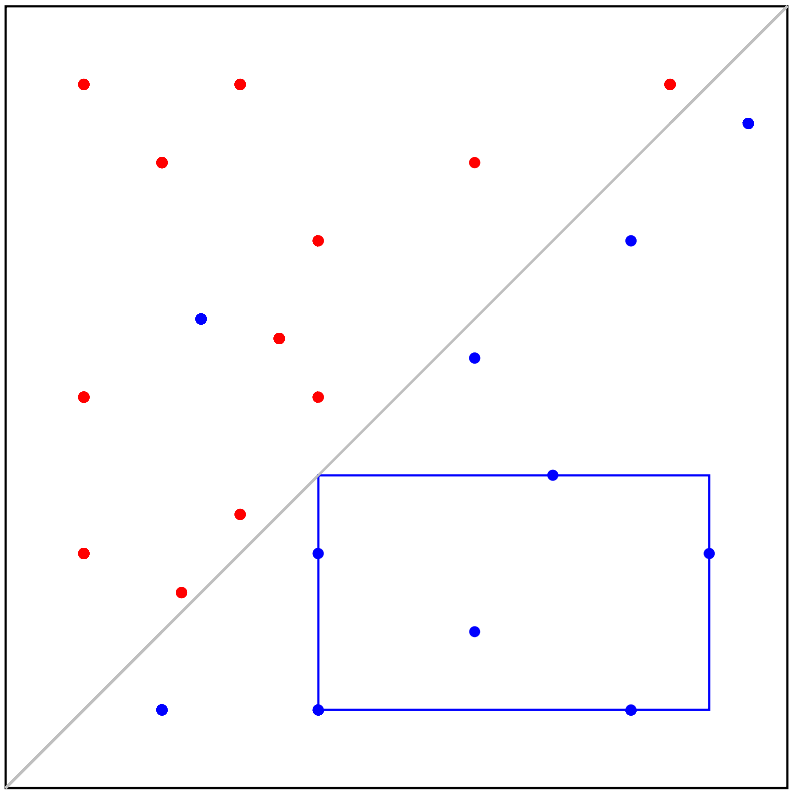
Algorithme MGC

Entrées : $E = [e_1, \ldots, e_n] \subseteq \mathcal{X}$ un ensemble *ordonné* de n exemples de la même classe, N un ensemble de contre exemples.

Sortie : $h \in \mathcal{H}$ une généralisation de E, maximalement correcte par rapport à E et N.

1: $h = e_1$
2: **for** $i = 2$ to n **do**
3: $h' = \mathtt{MG}(h, e_i)$ {Généralisation entre deux hypothèses.}
4: **if** $(\forall e \in N : h' \not\succeq e)$ **then**
5: $h = h'$ {h' (correcte) devient la généralisation courante.}
6: **end if**
7: **end for**
8: **return** h

Déroulement de MGC



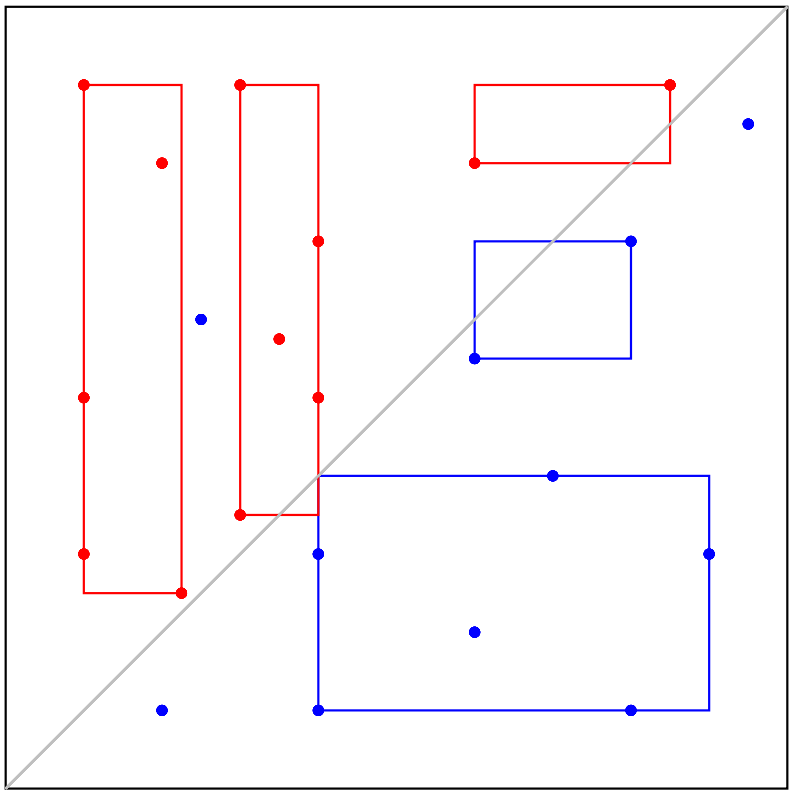
L'algorithme DLG [Webb and Agar, 1992]

Entrées : A un ensemble de n exemples (x_i, y_i).
Sortie : H un ensemble de règles.

1: $H = \emptyset$; $O = A$; $j = 1$
2: **while** $(O \neq \emptyset)$ **do**
3: $target$ = classe du premier exemple de O
4: $P = [x_i \in O | y_i = target]$
5: $N = [x_i \in A | y_i \neq target]$
6: $h_j = \texttt{MGC}(P,N)$
7: $O = O - [x \in O : h_j \succeq x]$
8: ajouter h_j à H ; $j = j + 1$
9: **end while**
10: **return** H

Déroulement

Couverture gloutonne des exemples : on répète le calcul de moindre généralisé correct sur les exemples non couverts jusqu'à couverture complète des exemples.



Bilan DLG

Avantages

- apprend rapidement ;
- permet d'appréhender les attributs pertinents ;
- donne des indications sur la difficulté du problème d'apprentissage : nombre de règles apprises, couverture de chaque règle.

Inconvénients

- glouton ;
- peu prédictif ;
- peu compréhensible.

Alternative : le système GloBo [Torre, 1999]

Protocole

- Algorithmes en présence : C4.5 [Quinlan, 1993], DLG, GloBo ;
- 20 problèmes du *repository UCI* [Blake and Merz, 1998] ;
- validations croisées 10 fois ;
- chaque apprentissage de GloBo est répété 10 fois ;
- nombre d'erreurs moyen.

Tableau de résultats I

Problème	C4.5	DLG	GloBo
audiology	18.20	24.16	20.76
breast-cancer	4.87	4.87	3.89
car	7.69	9.95	10.17
cmc	48.07	50.31	50.60
crx	14.79	20.57	16.50
dermatology	6.23	8.18	8.51
ecoli	15.89	22.35	23.88
glass	28.72	32.91	5.57
hepatitis	20.70	17.88	20.09
horse-colic	13.63	16.63	22.29
house-votes-84	3.22	4.83	7.39
ionosphere	7.96	14.84	8.74

Tableau de résultats II

iris	5.33	6.00	7.47
pima	29.29	30.47	27.04
promoters	18.17	19.17	22.60
sonar	28.97	32.18	31.09
tic-tac-toe	14.40	1.35	1.11
vowel	21.21	31.72	23.99
wine	8.83	12.33	8.94
zoo	7.51	4.38	5.55
Moyennes	16.18	*18.25*	*16.31*
	C4.5	DLG	GloBo

Plan

Combinaisons d'hypothèses faibles

Idées (issues du *boosting*)

- on dispose d'un apprenant L ;
- on le contraint à produire diverses hypothèses h_t ;
- les h_t sont combinées au sein d'un vote.

Vocabulaire (abus de langage)

- L est un apprenant faible ;
- chaque h_t est une hypothèse faible.

Méthode d'ensemble

Éléments constitutifs

Une méthode d'ensemble se définit par :

- l'espace des hypothèses $\mathcal{H}$;
- un apprenant faible L ;
- une stratégie utilisant L pour produire les $h_{t=1...T} \in \mathcal{H}$;
- une méthode de combinaison des h_t.

Sortie

Une méthode d'ensemble fournit un classifieur H :

- $H(x) = \text{sign}\left(\sum_{t=1}^{T} \alpha_t . h_t(x)\right)$
- on a changé d'espace... gain d'expressivité !

Aparté : Vapnik-Chervonenkis

Définition : Pulvérisation

Un ensemble $A \subseteq \mathcal{X}$ est dit *pulvérisé* par $\mathcal{H}$ ssi pour chaque étiquetage possible des exemples de A, $\mathcal{H}$ contient une hypothèse qui sépare parfaitement les classes.

Définition : VC dimension

La *VC dimension d'un ensemble d'hypothèses* $\mathcal{H}$ est la taille n du plus grand ensemble d'exemples A que $\mathcal{H}$ parvient à pulvériser.

... il existe un A de taille n tel que pour tout étiquetage...

Dimension VC de trois droites qui votent ?

voir trois droites qui vaporisent 9 points

GloBoost [Torre, 2005]

Idées [Dietterich, 2000b] : rendre L instable et voter à égalité.

Entrées : n exemples (x_i, y_i) et T un nombre d'itérations.
Sortie : H le classifieur final.

1: **for** $t = 1$ to T **do**
2: $target$ = classe choisie au hasard
3: $P = \{x_i | y_i = target\}$
4: $N = \{x_i | y_i \neq target\}$
5: mélanger P aléatoirement
6: $h_t = \texttt{MGC}(P,N)$ {Appel à l'algorithme MGC}
7: **end for**
8: **return** $H(x) = \text{sign}\left(\sum_{t=1}^{T} h_t(x)\right)$

aussi bagging [Breiman, 1996], boosting [Freund and Schapire, 1995].

Tableau de résultats I

Problème	C4.5	GloBo	AB+S	AB+MG	GloBoost
audiology	18.20	20.76	15.79	19.22	17.07
breast-cancer	4.87	3.89	4.16	3.10	3.65
car	7.69	10.17	13.36	9.74	11.52
cmc	48.07	50.60	44.87	49.61	47.67
crx	14.79	16.50	16.53	14.47	13.93
dermatology	6.23	8.51	4.38	4.63	3.94
ecoli	15.89	23.88	15.59	19.25	19.19
glass	28.72	5.57	22.53	4.59	4.55
hepatitis	20.70	20.09	18.23	18.39	17.62
horse-colic	13.63	22.29	21.71	18.42	16.54
house-votes-84	3.22	7.39	5.31	6.12	5.00
ionosphere	7.96	8.74	12.19	7.44	7.05

Tableau de résultats II

iris	5.33	7.47	5.33	5.33	5.67
pima	29.29	27.04	25.39	25.13	24.88
promoters	18.17	22.60	18.67	7.00	6.03
sonar	28.97	31.09	15.03	23.34	23.96
tic-tac-toe	14.40	1.11	1.67	0.00	0.14
vowel	21.21	23.99	14.65	9.30	13.44
wine	8.83	8.94	18.07	5.73	4.37
zoo	7.51	5.55	5.25	3.21	4.05
Moyennes	*16.18*	*16.31*	*14.94*	*12.70*	12.51
	C4.5	GloBo	AB+S	AB+MG	GloBoost

Constats et explications partielles

De bons résultats empiriques et pourtant :

- le rasoir d'Occam semble contredit ;
- les hypothèses théoriques du *boosting* ne sont pas vérifiées.

Éléments d'explication :

- biais, variance et erreur empirique ;
- les marges [Rätsch and Warmuth, 2003, Harries, 1999] ;
- la diversité [Dietterich, 2000a, Melville and Mooney, 2004, Kuncheva and Whitaker, 2003].

Résumé de l'architecture Volata

Trois niveaux :

- le premier niveau fournit l'opération MG permettant de calculer l'hypothèse moindre généralisée d'un ensemble d'exemples quelconque, découle de $\mathcal{H}$ et $\succeq$;
- le deuxième prend en compte les classes des exemples pour produire des hypothèses correctes, ou quasi-correctes si du bruit de classe est présent (MGC) ;
- le dernier niveau permet l'apprentissage d'un classifieur complet, par combinaison de règles élémentaires apprises par le niveau précédent (DLG, GloBo, GloBoost, BaggingMG, AdaBoostMG).

Seul le premier dépend des langages de représentation $\mathcal{E}$ et $\mathcal{H}$.

Plan

Séquences et alphabet

Notations

- un alphabet Σ, ensemble fini quelconque ;
- un mot w, une séquence finie de lettres de Σ ;
- mots possibles sur Σ : $\mathcal{X} = \Sigma^*$;
- on note ϵ le mot vide.

Par exemple

- $\Sigma = \{a, b\}$;
- $+$: ϵ, *aaa*, *b*, *abbaa*, *aaaaaa* ;
- $-$: *bba*, *aab*, *bbaababa*.

Objectifs : identifier un langage cible, apprendre à classer de nouveaux mots.

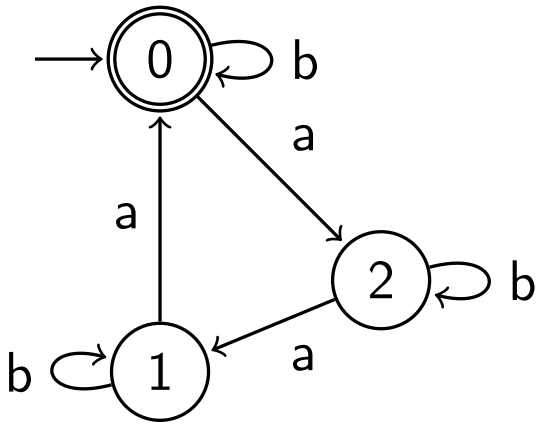
Quel ensemble $\mathcal{H}$ pour les mots ?

Les automates finis déterministes (AFD)

Définition : AFD

Un AFD est donné par $(Q, \Sigma, \delta, q_0, F)$:

- Q l'ensemble des états ;
- Σ l'alphabet ;
- $\delta : Q \times \Sigma \to Q$ la fonction de transition ;
- $q_0 \in Q$ l'état initial ;
- $F \subseteq Q$ les états finaux.



Caractéristiques et familles d'automates

Nombreux candidats pour $\mathcal{H}$:

- finis,
- minimaux,
- ambigus ou pas,
- déterministes ou non (équivalents),
- déterministes dans les deux sens (0-réversibles),
- k-TSS,
- k-réversibles,
- AFER, etc.

Classes étudiées en inférence grammaticale du point de vue de l'*apprenabilité à la limite*... Les preuves fournissent parfois un calcul de moindre-généralisé.

Automates 0-réversible

Définition : 0-réversible

Un automate A est 0-réversible s'il est déterministe dans les deux sens : A est déterministe et son miroir A^R l'est aussi.

L'algorithme ZR de [Angluin, 1982] débute par la construction du PTA de l'échantillon.

- automate en forme d'arbre, sans boucle ;
- reconnaît strictement l'échantillon, pas un mot de plus ;
- il faudra le généraliser.

MG-ZR : l'algorithme I

Algorithme MG-ZR (h, w)

Entrées : $h = (Q, \Sigma, q_0, F)$ un automate 0-réversible, w un mot

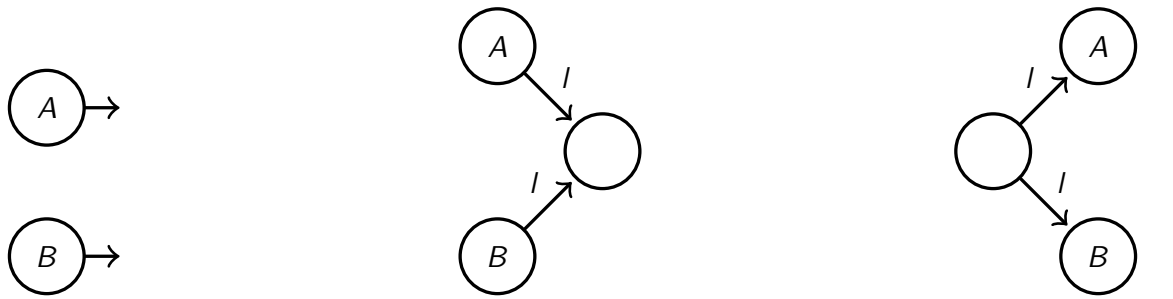
Sortie : h' un 0-réversible minimal généralisant h reconnaissant w

1: $i = 1$; $q = q_0$
{suivi des états existants}
2: **while** $\delta(q, w_i)$ is defined **do**
3: $q = \delta(q, w_i)$; $i = i + 1$
4: **end while**
{création d'une nouvelle branche pour les lettres restantes}
5: **while** $i \leq |w|$ **do**
6: créer un état q' et l'ajouter à Q
7: ajouter (q, w_i, q') à δ ; $i = i + 1$
8: **end while**
9: ajouter q à F

MG-ZR : l'algorithme II

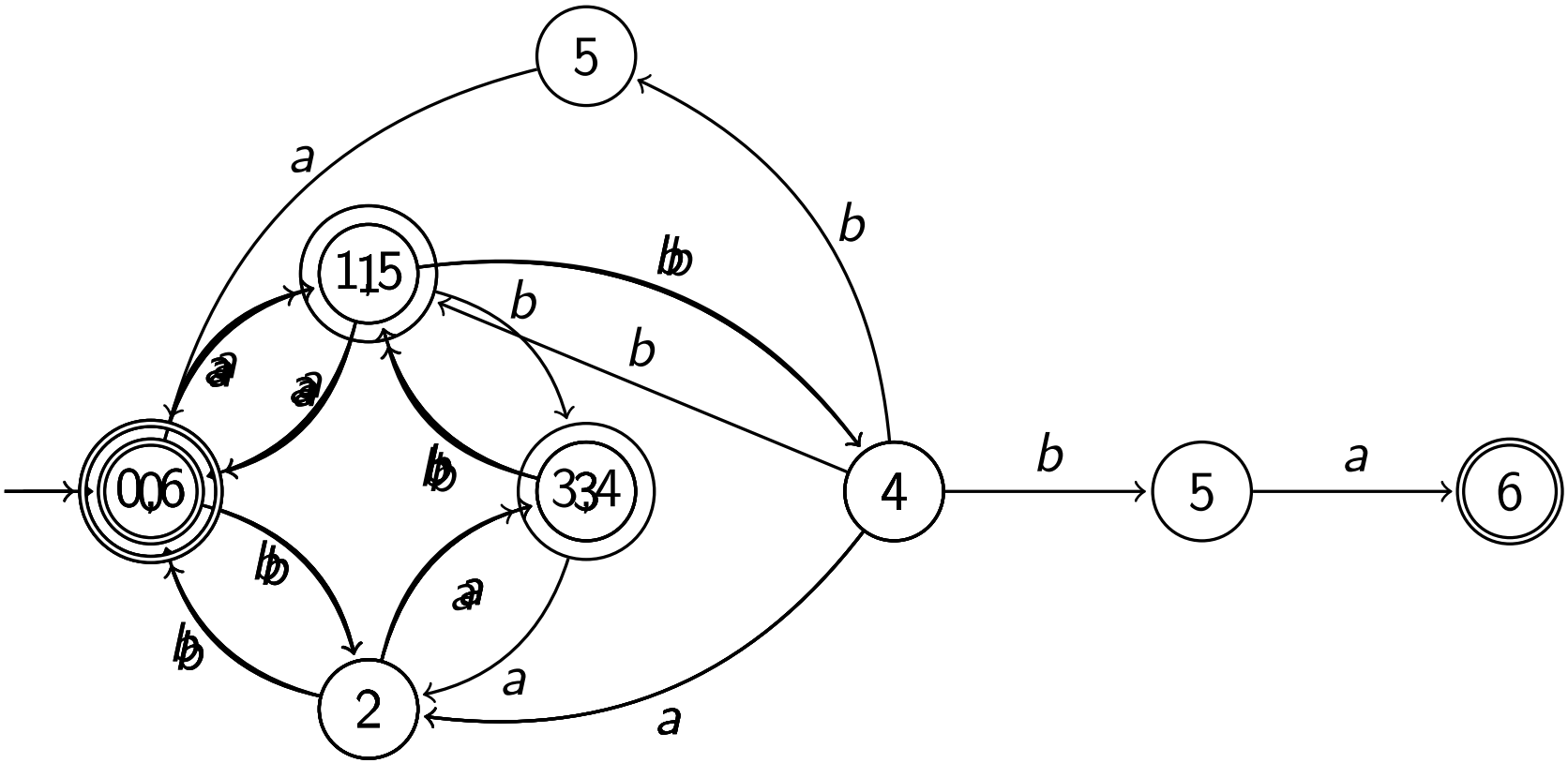
{mise en œuvre des fusions}
10: fusionner les états de F {deux états au plus}
11: **repeat**
12: si $\exists A, B \in Q$ et $\exists I \in \Sigma$ tels que $\delta(A, I) = \delta(B, I)$, fusionner A et B
13: si $\exists A, B, E \in Q$ et $\exists I \in \Sigma$ tels que $\delta(E, I) = \{A, B\}$, fusionner A et B
14: **until** no fusion
15: return $h' = (Q, \Sigma, \delta, q_0, F)$

Situations dans lesquelles les états A et B sont fusionnées par ZR :



MG-ZR : déroulement

$S = \{\epsilon, aa, bb, abab, baba\}$, ajout du mot *abba*.



Plan

Et à part les automates ? les boules de mots !

Boules : définitions [Tantini, 2009]

- trois opérations d'édition de coût unitaire :
  - insertion : $aab \rightarrow aa\underline{b}b$
  - suppression : $a\underline{a}b \rightarrow ab$
  - substitution : $a\underline{a}b \rightarrow a\underline{b}b$
- distance d'édition : $d(w_1, w_2)$ est le nombre minimal d'opérations qui permettent de passer de w_1 à w_2 ;
- un langage est donné par un mot-centre et un rayon : $B_2(bab)$.

Une boule de mots dénote un langage fini.

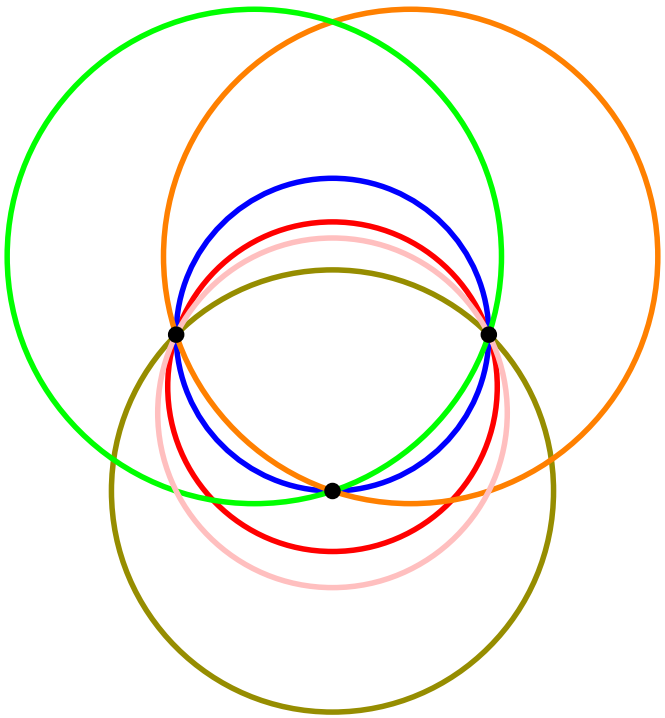
Intuitions sur les boules de mots [Tantini, 2009]

Attention, beaucoup de pièges...

- les boules de mots ne sont pas homogènes (la moitié des mots de la boule sont à distance maximale du centre) ;
- une boule de mots peut être contenue dans une autre de rayon plus petit : $B_5(ab) \subset B_4(abab)$;
- deux boules de mots de même rayon et de centres de même longueur peuvent dénoter des langages de tailles différentes : $|B_2(aaaabbbb)| \neq |B_2(abababab)|$.

Cercles : unicité ?

Une infinité de cercles moindres-généralisés pour capturer 3 points :



Boules de mots : unicité algorithmique

ajout de e à $B_r(c)$: pseudo-MG

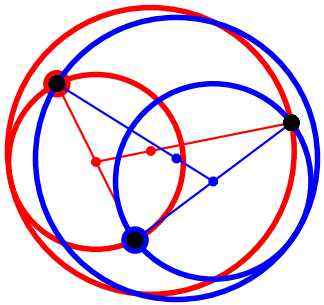
- Soit d la distance entre le c et e ;
- soit $r' = \frac{r+d}{2}$ le nouveau rayon ;
- le nouveau centre c' est sur le segment $[c; e]$, à une distance de e qui vaut r'.

ajout de *chiens* à $B_3(niche)$

- $d = \text{distance}(niche, chiens) = 5$;
- nouveau rayon : $r' = \frac{3+5}{2} = 4$;
- chemin d'édition : *niche* ; *iche* ; *che* ; *ches* ; *chens* ; *chiens* ;
- nouvelle hypothèse : $B_4(iche)$.

VC-dimension des boules de mots ? Boules et tic-tac-toe ?

Généralisation en boules : illustration



Automates versus boules

Theorem ([Janodet, 2010])

La VC-dimension des boules, sur un alphabet $|\Sigma| = 2$, est infinie.

Démonstration.

- n mots de longueur n ;
- le i^{e} mot n'est constitué que de a à l'exception de la i^{e} lettre qui est un b ;
- k sont étiquetés positivement, $(n - k)$ négativement ;
- boule de centre le mot de longueur n avec b aux mêmes positions que les positifs et des a ailleurs, et de rayon $(k - 1)$.

□

Expérimentations

- problèmes séquentiels de [Blake and Merz, 1998] : morpion, badges, prénoms américains, génomique ;
- concurrents : RPNI, Traxbar, Redblue ;
- problème réel de reconnaissance de chiffres manuscrits ;
- concurrents : SeDiL [Boyer et al., 2008].

Jeux de données UCI

	tic-tac-toe	badges	promoters	first-name	splice
Majorité	65.34 %	**71.43 %**	50.00 %	81.62 %	50.26 %
RPNI	91.13 %	62.24 %	-	81.42 %	-
TraxBar	90.81 %	57.48 %	56.60 %	81.37 %	**58.33 %**
Red-Blue	**93.89 %**	61.09 %	**63.02 %**	**82.83 %**	54.65 %
G-tssi	91.47 %	**72.69 %**	**61.13 %**	**89.50 %**	**78.07 %**
G-zr	**98.36 %**	71.43 %	50.00 %	83.07 %	-
G-balls$_{1\,000}$	92.77 %	81.72 %	88.58 %	87.45 %	93.78 %
G-balls$_{10\,000}$	94.69 %	**82.21 %**	88.90 %	89.47 %	95.83 %
G-balls$_{100\,000}$	**95.03 %**	81.98 %	**89.74 %**	**89.93 %**	**96.12 %**

Reconnaissance de chiffres manuscrits

SeDiL	**95.86 %**
G-balls$_{1\,000}$	93.81 %
G-balls$_{10\,000}$	95.93 %
G-balls$_{100\,000}$	**96.32 %**

Bilan de la combinaison d'automates

Regardons quelques théories apprises.

- les algorithmes d'apprentissage par positifs seuls comme ZR sont utilisables dans nos méthodes génériques ;
- et permettent d'obtenir de bonnes performances raisonnables ;
- les boules de mots sont très compétitives malgré l'absence de moindre-généralisé unique ;
- boules creuses... moindre-généralisés ?

Plus de résultats dans
[Torre and Terlutte, 2009, Tantini et al., 2010].

Plan

Inductive Logic Programming [Muggleton, 1991, Muggleton and De Raedt, 1994]

ILP : apprentissage de programmes logiques.
Lien avec les bases de données.

- $\mathcal{E}$ et $\mathcal{H}$ sont des clauses dont la tête a pour symbole de prédicat le concept que l'on cherche à caractériser ;
- les clauses de $\mathcal{E}$ n'utilisent pas de variable ;
- cible : souvent une seule clause, pas récursive...
- possible existence d'une *théorie du domaine* à travers des prédicats déjà définis ; prise en compte ?
- quels sont les tests de subsomption possibles ? et le moindre généralisé (unicité, complexité) ?

L'implication logique

- La relation la plus naturelle ;
- mais indécidable entre clauses [Schmidt-Schauß, 1988] ;
- et indécidable entre clauses de Horn [Marcinkowski and Pacholski, 1992].

Nota bene : le moindre généralisé est défini, son calcul est de complexité exponentielle [Nienhuys-Cheng and de Wolf, 1996].

La θ-subsomption [Plotkin, 1970]

Définition : θ-subsomption

On dit que C θ-subsume D ($C \succeq_\theta D$) si et seulement s'il existe une substitution θ telle que $C\theta \subseteq D$.
C et D sont θ-équivalentes ($C \sim_\theta D$) ssi $C \succeq_\theta D$ et $D \succeq_\theta C$.

NP-difficile [Kapur and Narendran, 1986], équivalente à l'implication logique sur les clauses non récursives [Gottlob, 1987].

Pistes

- implémentations *efficaces* [Kietz and Lübbe, 1994, Scheffer et al., 1996, Maloberti and Sebag, 2004] ;
- subsomption *stochastique* [Sebag and Rouveirol, 1997] ;
- clauses *déterminées* [Muggleton and Feng, 1990].

θ-subsomption : exemple

grand-père(A,B) $\leftarrow$ père(A,C), père(C,B) $\succeq_\theta$
grand-père(abraham,bart) $\leftarrow$ père(abraham,homer), père(homer,bart) ?

grand-père(A,B)	grand-père(abraham,bart)	$A/abraham$, $B/bart$
père(A,C)	père(abraham,homer) père(homer,bart)	$A/abraham$, $C/homer$ $A/homer$, $C/bart$
père(C,B)	père(abraham,homer) père(homer,bart)	$C/abraham$, $B/homer$ $C/homer$, $B/bart$

Oui, avec $\theta = \{A/abraham, B/bart, C/homer\}$.

Autre exemple : q $\leftarrow$ p(X,Y), p(Y,X) et q $\leftarrow$ p(X,X) ?

Réduction sous θ-subsomption

Définition : clause non réduite

Une clause C est dite *non réduite* si C contient un littéral L tel que $C \sim_\theta C \setminus L$ ou, autrement dit, s'il existe une substitution θ telle que $C\theta \subsetneq C$.

Trois clauses

$C_0 : c(X) \leftarrow p(X, s(0))$
$C_1 : c(X) \leftarrow p(X, s(0)), p(A, B)$
$C_2 : c(X) \leftarrow p(X, s(0)), p(C, s(D)), p(X, E)$

- C_0 est réduite ;
- C_1 n'est pas réduite car $C_1\theta_1 \subset C_1$ avec $\theta_1 = \{X/A; s(0)/B\}$;
- C_2 non plus car $C_2\theta_2 \subset C_2$ avec $\theta_2 = \{X/C; 0/D; s(0)/E\}$;
- les trois clauses sont équivalentes.

Moindre généralisé sous θ-subsomption

Il y a unicité et un algorithme polynomial [Plotkin, 1970] :

- deux termes : on garde ce qui est commun, on remplace ce qui diffère par une nouvelle variable ;
- deux littéraux avec même symbole de prédicat : on généralise les arguments terme à terme ;
- deux clauses : on généralise les littéraux deux à deux (la taille du moindre généralisé est de l'ordre du produit des tailles des clauses de départ) ;
- idée clef : même variable si l'on retrouve les deux mêmes termes ;

Mais :

- la clause construite est souvent non réduite ;
- l'algorithme de réduction proposé par [Plotkin, 1970] utilise des tests de θ-subsomption...

Moindre généralisé : exemple 1

le concept grand-père

- Calcul de MG $\left(\begin{array}{l} \text{grand-père(a,b)} \leftarrow \text{père(a,c), père(c,b).} \\ \text{grand-père(d,e)} \leftarrow \text{père(d,f), mère(f,e).} \end{array} \right)$
- Appariements : $\begin{array}{l|l|l} \text{grand-père(a,b)} & \text{père(a,c)} & \text{père(c,b)} \\ \text{grand-père(d,e)} & \text{père(d,f)} & \text{père(d,f)} \end{array}$
- Première étape : $\begin{array}{l|l|l} \text{grand-père(A,b)} & \text{père(A,c)} & \text{père(c,b)} \\ \text{grand-père(A,e)} & \text{père(A,f)} & \text{père(d,f)} \end{array}$
- Résultat : grand-père(A,B) $\leftarrow$ père(A,C), père(D,E).

Le résultat est une clause non réduite...

Moindre généralisé : exemple 2

maisons avec fenêtres (petites ou grandes, blanches ou noires)

m(m1)←f(m1,g,b), f(m1,p,n).
m(m2)←f(m2,g,n), f(m2,p,b).

m(m1)←f(m1,g,b),	f(m1,p,n),	f(m1,g,b),	f(m1,p,n).
m(m2)←f(m2,g,n),	f(m2,p,b),	f(m2,p,b),	f(m2,g,n).
m(M) ←f(M,g,C1),	f(M,p,C2),	f(M,T1,b),	f(M,T2,n).

Clause réduite ?

Combinaisons : discussions

Leçons de [Plotkin, 1970]

- existence d'un moindre généralisé unique sous θ-subsomption ;
- apparition possible d'hypothèses non réduites ;
- réduction et tests de subsomption coûteux.

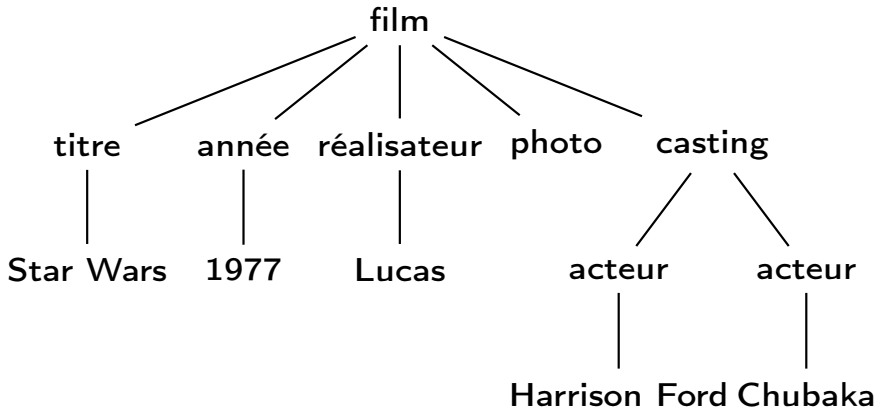
Manipulation de clauses quasi-déterminées [Decoster et al., 2010]

Exemple : codage d'arbres basé sur `first-child` et `next-sibling`.

Un codage ILP pour la classification d'arbres

Proposition

- coder les arbres à l'aide de clauses déterminées pour casser la complexité du test de θ-subsomption ;
- utiliser le calcul de moindre-généralisé de [Plotkin, 1970] ;
- intégration dans volata.



$$\text{ciné}(sw) \leftarrow \quad p_1(\text{film}), p_{1.1}(\text{titre}), p_{1.2}(\text{année}), p_{1.3}(\text{réalisateur}), p_{1.4}(\text{photo}), p_{1.5}(\text{casting}), p_{1.5.1}(\text{acteur}), p_{1.5.2}(\text{acteur}).$$

bilan Volata

- validation des principes de Volata ;
- implémentation générique disponible ;
- pour chaque structure de données, il suffit de définir la relation de subsomption appropriée et le calcul de moindre généralisé associé.

perspective SVM

- concurrents : SVM ;
- pour chaque structure de données, on doit définir une fonction noyau ;
- à comparer expérimentalement.

Bibliographie I


Angluin, D. (1982).
Inference of reversible languages.
Journal of the ACM, 29(3) :741–765.


Blake, C. and Merz, C. (1998).
UCI repository of machine learning databases
[http ://archive.ics.uci.edu/ml/].


Boyer, L., Esposito, Y., Habrard, A., Oncina, J., and Sebban, J. (2008).
Sedil : Software for edit distance learning.
In Daelemans, W., Goethals, B., and Morik, K., editors, *Proceedings of the 19th European Conference on Machine Learning*, pages 672–677. Springer.

Bibliographie II

Breiman, L. (1996).
Bagging predictors.
Machine Learning, 24(2) :123–140.

Decoster, J., Staworko, S., and Torre, F. (2010).
Apprentissage relationnel polynomial pour la classification d'arbres.
In Mephu Nguifo, E., editor, *12ème Conférence francophone sur l'Apprentissage automatique (CAp'2010)*, pages 189–200, Clermont-Ferrand. PUG.

Bibliographie III


Dietterich, T. G. (2000a).
Ensemble methods in machine learning.
In Kittler, J. and Roli, F., editors, *First International Workshop on Multiple Classifier Systems*, volume 1857 of *Lecture Notes in Computer Science*, pages 1–15. Springer-Verlag.


Dietterich, T. G. (2000b).
An experimental comparison of three methods for constructing ensembles of decision trees : Bagging, boosting, and randomization.
Machine Learning, 40(2) :139–158.

Bibliographie IV


Dupont, P. and Miclet, L. (1998).
Inférence grammaticale régulière : fondements théoriques et principaux algorithmes.
Technical Report RR-3449, INRIA.


Freund, Y. and Schapire, R. E. (1995).
A decision-theoretic generalization of on-line learning and an application to boosting.
In *European Conference on Computational Learning Theory*, pages 23–37.


Gottlob, G. (1987).
Subsumption and implication.
Information Processing Letters, 24(2) :109–111.

Bibliographie V

Harries, M. (1999).
Boosting a strong learner : evidence against the minimum margin.
In *Proc. 16th International Conf. on Machine Learning*, pages 171–180. Morgan Kaufmann, San Francisco, CA.

Janodet, J.-C. (2010).
The vapnik-chervonenkis dimension of balls of strings is infinite.
Personal Communication.

Kapur, D. and Narendran, P. (1986).
Np-completeness of the set unification and matching problems.
In *Proceedings of 8th Conference on Automated Deduction*, volume 230, pages 489–495. Springer-Verlag.

Bibliographie VI


Kietz, J.-U. and Lübbe, M. (1994).
An efficient subsumption algorithm for inductive logic programming.
In Cohen, W. W. and Hirsh, H., editors, *Proceedings 11th International Conference on Machine Learning*, pages 130–138. Morgan Kaufmann.


Kuncheva, L. I. and Whitaker, C. J. (2003).
Measures of diversity in classifier ensembles and their relationship with the ensemble accuracy.
Machine Learning, 51(2) :181–207.


Maloberti, J. and Sebag, M. (2004).
Fast theta-subsumption with constraint satisfaction algorithms.
Machine Learning, 55(2) :137–174.

Bibliographie VII


Marcinkowski, J. and Pacholski, L. (1992).
Undecidability of the horn-clause implication problem.
In IEEE, editor, *Proceedings of the 33rd Annual Symposium on Foundations of Computer Science*, pages 354–362, Pittsburgh, PN. IEEE Computer Society Press.


Melville, P. and Mooney, R. J. (2004).
Creating diversity in ensembles using artificial data.
Information Fusion : Special Issue on Diversity in Multiclassifier Systems.


Muggleton, S. (1991).
Inductive logic programming.
New Generation Computing Journal, 8(4) :295–317.

Bibliographie VIII


Muggleton, S. and De Raedt, L. (1994).
Inductive logic programming : Theory and methods.
Journal of Logic Programming, 19 :629–679.


Muggleton, S. and Feng, C. (1990).
Efficient induction of logic programs.
In *Proceedings of the 1st Conference on Algorithmic Learning Theory*, pages 368–381. Ohmsma, Tokyo, Japan.


Nienhuys-Cheng, S. and de Wolf, R. (1996).
Least generalizations and greatest specializations of sets of clauses.
Journal of Artificial Intelligence Research, 4 :341–363.

Bibliographie IX

Plotkin, G. (1970).
A note on inductive generalization.
In Meltzer, B. and Mitchie, D., editors, *Machine Intelligence*, volume 5, pages 153–165. Edinburgh University Press.

Quinlan, J. R. (1993).
C4.5 : programs for machine learning.
Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

Rätsch, G. and Warmuth, M. K. (2003).
Efficient margin maximizing with boosting.
submitted to Journal of Machine Learning Research (JMLR).

Bibliographie X

Scheffer, T., Herbrich, R., and Wysotzki, F. (1996).
Efficient θ-subsumption based on graph algorithms.
In Muggleton, S., editor, *Proceedings of the 6th International Workshop on Inductive Logic Programming*, pages 312–329. Stockholm University, Royal Institute of Technology.

Schmidt-Schauß, M. (1988).
Implication of clauses is undecidable.
TCS : Theoretical Computer Science, 59(3) :287–296.

Sebag, M. and Rouveirol, C. (1997).
Tractable induction and classification in FOL via stochastic matching.
In *Proceedings of the 15th International Joint Conference on Artificial Intelligence*, pages 888–892.

Bibliographie XI

Tantini, F. (2009).
Inférence grammaticale en situations bruitées.
PhD thesis, Université Jean Monnet de Saint-Étienne.

Tantini, F., Terlutte, A., and Torre, F. (2010).
Combinaisons de boules de mots pour la classification de séquences.
In Mephu Nguifo, E., editor, *12ème Conférence francophone sur l'Apprentissage automatique (CAp'2010)*, pages 161–176, Clermont-Ferrand. PUG.

Bibliographie XII


Torre, F. (1999).
GloBo : un algorithme stochastique pour l'apprentissage supervisé et non-supervisé.
In Sebag, M., editor, *Actes de la Première Conférence d'Apprentissage*, pages 161–168.


Torre, F. (2005).
Globoost : Combinaisons de moindres généralisés.
Revue d'Intelligence Artificielle, 19(4-5) :769–797.

Bibliographie XIII


Torre, F. and Terlutte, A. (2009).
Méthodes d'ensemble en inférence grammaticale : une approche à base de moindres généralisés.
In Bennani, Y. and Rouveirol, C., editors, *11ème Conférence francophone sur l'Apprentissage automatique (CAp'2009)*, pages 33–48, Hammamet (Tunisie). PUG.


Webb, G. I. and Agar, J. W. M. (1992).
Inducing diagnostic rules for glomerular disease with the DLG machine learning algorithm.
Artificial Intelligence in Medicine, 4 :419–430.