HAL open science

Un outil pour la comparaison et la validation d'algorithmes d'apprentissage à partir de données

Michel Crucianu, Gilles Verley, Jean-Pierre Asselin de Beauville, Romuald Boné

▶ **To cite this version:**

Michel Crucianu, Gilles Verley, Jean-Pierre Asselin de Beauville, Romuald Boné. Un outil pour la comparaison et la validation d'algorithmes d'apprentissage à partir de données. La revue MODULAD, 2001, 26, pp.55-70. hal-01525464

HAL Id: hal-01525464

https://hal.science/hal-01525464

Submitted on 31 May 2017

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Distributed under a Creative Commons Attribution - NonCommercial - NoDerivatives 4.0 International License

Un outil pour la comparaison et la validation d'algorithmes d'apprentissage à partir de données

Michel Crucianu, Gilles Verley, Jean-Pierre Asselin de Beauville, Romuald Boné

Laboratoire Informatique de l'Université de Tours

École d'Ingénieurs en Informatique pour l'Industrie

64, avenue Jean Portalis

37200 TOURS

{crucianu, verley}@univ-tours.fr, jean-pierre.asselin@aupelf-uref.org

Résumé : *L'apprentissage supervisé à partir de données se trouve dans la situation paradoxale où il y a surabondance d'algorithmes et pénurie de méthodes permettant de comparer et d'appliquer ces algorithmes à bon escient. Remédier à cette situation suppose que la spécificité de tout nouvel algorithme soit explicitée de manière intelligible, interprétable et donc applicable, puis mise relation avec l'information* a priori *concernant les différentes classes de problèmes traités. Pour aider dans cette démarche, nous présentons un site Internet qui sert d'outil de gestion des connaissances en explorant les liens existant entre les spécificités des algorithmes et les caractéristiques des problèmes.*

Mots-clés : *apprentissage supervisé, validation, a priori, gestion des connaissances.*

Keywords: *supervised learning, validation, priors, knowledge management.*

1. Introduction

Au cours de la dernière décennie nous avons assisté à un développement explosif des techniques d'apprentissage supervisé à partir de données. Si une telle abondance a permis des avancées considérables dans des domaines d'application des plus variés, elle a aussi donné naissance à un contexte dans lequel certaines questions se posent avec plus d'acuité. Comment choisir les algorithmes les plus adaptés pour un type de problème particulier ? Comment déterminer le domaine d'intérêt d'un algorithme particulier ? Les méthodes et les connaissances permettant d'appliquer les algorithmes à bon escient n'ont malheureusement pas connu une évolution aussi rapide. Profiter de l'existant pour résoudre un problème dans le cadre d'une nouvelle application devient alors une démarche très laborieuse qui repose, au mieux, sur l'expérience individuelle du chercheur. Face à la difficulté de faire des choix pertinents en temps limité dans un univers aussi riche, il arrive même que l'on préfère concevoir un algorithme *ad hoc* pour traiter un problème.

Remédier à cette situation suppose que la spécificité de tout nouvel algorithme soit explicitée de manière intelligible et interprétable par rapport aux caractéristiques des problèmes traités. Ceci montre l'intérêt d'un outil de gestion des connaissances qui permettrait de répertorier les particularités des algorithmes et des problèmes, ainsi que d'aider à l'exploration conjointe et systématique de ces deux espaces.

Dans la première partie de ce papier, nous mettrons en évidence l'intérêt qu'il y a à connaître le comportement des algorithmes d'apprentissage supervisé candidats face aux caractéristiques des problèmes potentiels dans le domaine d'application considéré. [dire aussi que le but n'est pas de faire une synthèse exhaustive sur le sujet ? Non!] Nous insistons aussi sur les différentes façons d'obtenir et d'exploiter ces connaissances.

Ensuite, nous présenterons les idées qui nous ont guidés dans la réalisation d'un site Internet comme outil de gestion des connaissances liées à la validation et la comparaison d'algorithmes d'apprentissage supervisés. Les fonctionnalités offertes par le site dans sa version actuelle seront ensuite décrites.

2. Choix et validation d'algorithmes d'apprentissage

Les deux principales questions qui nous intéressent ici sont dans une large mesure symétriques :

1° Comment sélectionner un algorithme d'apprentissage pour traiter un problème et, plus généralement, comment comparer différents algorithmes par rapport à un ensemble de problèmes ?

2° Comment valider un algorithme d'apprentissage et, plus particulièrement, comment déterminer son domaine d'intérêt ?

Les réponses à ces deux questions dépendent, en premier lieu, du niveau des connaissances disponibles concernant les algorithmes candidats et les problèmes à traiter.

Avant d'aller plus loin, précisons un peu plus les notions de problème et d'algorithme d'apprentissage à partir de données. Un problème consiste en la modélisation d'une dépendance entre les éléments de deux ensembles finis $\mathbf{X}$ et $\mathbf{Y}$. La dépendance recherchée (cible) f peut être représentée à partir d'une distribution sur $\mathbf{Y}$ conditionnée par $\mathbf{X}$, $P(y|f,x) = f(x,y)$, $x \in \mathbf{X}$, $y \in \mathbf{Y}$. L'ensemble des données disponibles, D, est un ensemble ordonné et fini de paires (x,y). En plus de ces données, nous pouvons détenir des connaissances *a priori* concernant le problème, connaissances exprimables (souvent difficilement dans la pratique) sous la forme d'une distribution $P(f)$ sur l'espace de toutes les dépendances possibles. Le résultat de l'apprentissage, h, est aussi une distribution sur $\mathbf{Y}$ conditionnée par $\mathbf{X}$, $P(y|h,x) = h(x,y)$. L'objectif recherché est d'obtenir une bonne généralisation, c'est à dire d'arriver à un modèle h très proche de la dépendance cible f sur l'ensemble des paires (x,y) et non seulement celles de D.

Dans ce contexte, nous pouvons considérer qu'un algorithme d'apprentissage associe à D une distribution $P(h|D)$, éventuellement dégénérée [peut-on parler de dégénérescence pour des distributions sur des ensembles finis?] (algorithme déterministe). Par exemple, pour la modélisation d'une série chronologique à l'aide de réseaux de neurones, l'algorithme d'apprentissage comprend non seulement la technique d'estimation des poids du réseau, mais aussi toutes les techniques mises en œuvre pour déterminer le type et l'architecture exacte du réseau, ainsi que pour choisir et paramétrer la technique d'estimation des poids.

Pour un problème, en dehors des données, nous pouvons connaître les résultats obtenus par plusieurs algorithmes sur le problème ou détenir des informations *a priori* sur ses caractéristiques.

Dans le pire des cas, un algorithme d'apprentissage est utilisé comme une boîte noire sans références. Mais nous pouvons disposer de connaissances supplémentaires, comme les performances de l'algorithme sur un certain nombre d'autres problèmes, ou les détails de son fonctionnement.

Nous examinons dans la suite les options possibles en fonction du niveau des connaissances disponibles concernant le problème à traiter et les algorithmes d'apprentissage supervisé candidats. Cela nous permettra de mettre en évidence l'intérêt qu'il y a à connaître le comportement des algorithmes face aux caractéristiques des problèmes potentiels dans le domaine d'application considéré. Nous présenterons aussi différentes façons d'obtenir et d'exploiter ces connaissances.

2.1. Connaissance minimale

Nous nous trouvons dans cette situation quand notre connaissance du problème se réduit aux données, ou notre connaissance de l'algorithme se réduit à l'identité de la boîte noire.

Comment pouvons-nous déterminer quel algorithme possède le meilleur potentiel de généralisation sur le problème à traiter ? L'idéal serait d'arriver à choisir entre les différents algorithmes candidats *a priori*, donc sans avoir à effectuer le moindre essai sur

le problème. Pouvons-nous privilégier un algorithme indépendamment du problème à traiter ? Y a-t-il un algorithme meilleur que tous les autres quelle que soit la dépendance cible ? Les résultats *No Free Lunch* (NFL) de [Wolpert, 1996 a, b] montrent que si la distribution $P(\phi|)$ des dépendances cibles possibles est uniforme[1] alors tous les algorithmes se valent en moyenne. Aussi, le seul fait de savoir que cette distribution n'est pas uniforme ne nous apporte rien, car le même résultat peut être obtenu en considérant non plus une moyenne sur une $P(\phi|)$ uniforme, mais une moyenne uniforme sur l'ensemble de toutes les distributions $P(\phi|)$ non uniformes possibles. L'unique ligne de défense serait de connaître la distribution dont fait partie la dépendance cible du problème que l'on essaye de traiter, **et** de savoir que l'algorithme utilisé est bien adapté à cette distribution. Si une de ces deux informations est absente, l'autre ne nous permet pas de faire un choix.

Pourtant, de nombreux résultats obtenus dans le cadre de la théorie statistique de l'apprentissage (voir par exemple [Vapnik et Chervonenkis, 1971], [Ehrenfeucht et al., 1989], ou une synthèse récente dans [Vapnik, 1998]) donnent un avantage *a priori* aux algorithmes qui privilégient les modèles h peu complexes, car ils permettent d'obtenir des intervalles de confiance pour l'erreur en généralisation à partir d'une mesure de la complexité des modèles employés (souvent la dimension de Vapnik-Chervonenkis), du nombre de données d'apprentissage et de l'erreur en apprentissage. Cet avantage est pour le moment essentiellement théorique, dans la mesure où pour trouver des intervalles de confiance intéressants il faut disposer d'un nombre très élevé de données pour l'apprentissage (malgré des progrès récents, voir par exemple [Elisseeff et al., 1999]).

Sans entrer plus dans le débat ici, nous remarquerons seulement que les hypothèses de travail de la théorie statistique de l'apprentissage et des théorèmes NFL ne sont pas les mêmes [en particulier, Vapnik utilise des bornés de Rn et Wolpert des ensembles finis], et que la controverse demeure quant au choix des hypothèses les plus appropriées.

Quand nos connaissances sur les algorithmes candidats nous permettent de faire appel aux résultats mentionnés de la théorie statistique de l'apprentissage (et à condition de privilégier ses hypothèses de travail par rapport à celles de NFL) [à supprimer car on ne s'étend pas assez sur les fondements de ces différences d'hypothèses!], nous avons donc la possibilité de choisir *a priori* l'algorithme, avec les données d'apprentissage comme seule information concernant le problème. [Les résultats statistiques que l'algo satisfait permettent de donner a priori une borne supérieure de la confiance sur les performances en généralisation mais, en aucun cas, une borne supérieure de la performance elle-même. C'est alors un peu exagéré de dire que l'on peut alors choisir l'algo d'apprentissage avec les données d'apprentissage comme seule information concernant le problème]

En revanche, nous n'avons aucun moyen de choix *a priori* si nous possédons des connaissances détaillées relatives au problème à traiter, mais aucune concernant les algorithmes candidats.

---

[1] Dans les travaux mentionnés, l'ensemble des dépendances cibles possibles est un ensemble fini.

Dans cette situation de connaissance minimale il est donc rarement possible de choisir *a priori* entre les différents algorithmes candidats. Les circonstances s'améliorent-elles si on accepte d'essayer les différents algorithmes sur le problème et de faire le choix *a posteriori* ? Une réponse positive sera soumise à deux conditions : avoir une méthode fiable d'estimation de la généralisation à partir des résultats obtenus sur les données disponibles, et se limiter dès le départ à un nombre restreint d'algorithmes candidats.

Si on se place dans les hypothèses de travail des théorèmes NFL [Wolpert, 1996 a, b], la première condition ne peut pas être satisfaite sans connaissances supplémentaires. En revanche, si on préfère les hypothèses de travail de la théorie statistique de l'apprentissage – choix le plus fréquent dans la littérature – nous avons plusieurs moyens pour estimer l'erreur en généralisation à partir de l'erreur en apprentissage. Il faut remarquer qu'une bonne qualité d'estimation exige toujours un volume très important de données. Un premier moyen est de faire appel aux résultats mentionnés plus haut [Vapnik, 1998], qui permettent d'obtenir des intervalles de confiance pour l'erreur en généralisation grâce à une mesure de la complexité des modèles employés [au demeurant, assez rarement calculables, sauf progrès notables que j'ignore!].

Un deuxième moyen est de mettre en oeuvre des techniques de validation croisée [référence francophone ?] qui proposent des estimations pour l'erreur en généralisation. La complexité des modèles privilégiés par l'algorithme d'apprentissage n'intervient pas directement dans l'estimation, mais influe sur le choix de la technique appropriée (voir par exemple [Moody, 1994]). Pour une comparaison expérimentale de différentes techniques de ce type on peut consulter [Dietterich, 1998].

Enfin, la méthode la plus utilisée consiste à diviser les données disponibles en deux ensembles disjoints, un ensemble d'apprentissage plus un ensemble de test, et de se servir de l'ensemble de test pour évaluer la qualité de la généralisation. Cette méthode peut être appliquée même en l'absence de connaissances concernant l'algorithme considéré.

Après l'évaluation de la performance en généralisation pour chacun des algorithmes candidats, celui qui donne les meilleurs résultats est retenu et utilisé par la suite pour le problème traité. La nécessité de se limiter dès le départ à un nombre restreint d'algorithmes est une conséquence de la diminution de la fiabilité d'estimation des performances *in fine* avec l'augmentation du nombre de candidats (voir la troisième inégalité de [Hoeffding, 1963]). Les difficultés d'un choix *a priori*, déjà mentionnées, se manifestent de nouveau dans cette présélection des candidats.

2.2. Connaissances implicites

Lorsqu'un nouvel algorithme d'apprentissage est mis au point, il est fréquent qu'on l'évalue sur un ensemble de problèmes de référence, bien connus. En effectuant cette démarche pour un ensemble d'algorithmes nous obtenons, à partir d'un ensemble bien défini de problèmes de référence, une ou éventuellement plusieurs hiérarchies entre ces algorithmes. Ensuite, lorsqu'on cherche un algorithme performant *a priori* pour un nouveau problème à traiter, la tentation est grande de faire appel à ces connaissances implicites pour choisir l'algorithme approprié.

Deux approches doivent être mentionnées ici. La première, correspondant en général à l'existence d'une hiérarchie globale entre les résultats des candidats sur les problèmes de référence, consiste à choisir le « meilleur » algorithme, sans vraiment s'intéresser aux rapports entre le problème à traiter et les problèmes de référence. Cette approche est très difficile à défendre. Les théorèmes NFL montrent même que l'algorithme qui fournit les meilleurs résultats sur un ensemble de problèmes sera en moyenne le plus mauvais sur l'ensemble de tous les autres problèmes possibles. Sans plus de connaissances reliant le problème à traiter aux problèmes de référence, l'inférence sous-jacente à cette approche est donc fausse.

En l'absence de hiérarchie globale, la deuxième approche, plus affinée, est appliquée quand l'ensemble des problèmes de référence peut être décomposé en sous-ensembles et on arrive à trouver des hiérarchies partielles entre les candidats – une hiérarchie pour chaque sous-ensemble. Le sous-ensemble le plus similaire au problème à traiter est alors déterminé et le meilleur algorithme sur ce sous-ensemble est sélectionné. En apparence, cette deuxième approche n'est pas sujette à la même critique que la première. Mais la notion de similarité entre des problèmes de référence et le problème à traiter mérite d'être mieux explorée. Il faut déterminer si les meilleurs résultats obtenus par un algorithme sur un sous-ensemble de problèmes de référence sont dus aux caractéristiques qui rendent ces problèmes similaires au problème à traiter, ou à d'autres caractéristiques, *ad hoc*. Une telle connaissance met en rapport les spécificités d'un algorithme et les caractéristiques d'un ensemble de problèmes, et doit être qualifiée d'explicite. Les connaissances implicites ont le mérite d'initier une démarche d'explicitation. Si cette démarche n'aboutit pas (en raison de l'opacité des problèmes ou des algorithmes), la critique provenant des théorèmes NFL continue à s'appliquer ; les seules connaissances implicites ne constitueront donc pas une justification valide de la deuxième approche.

Si les connaissances implicites s'avèrent insuffisantes pour appuyer le choix *a priori* d'un algorithme d'apprentissage, leur accumulation permet d'initier et éventuellement de guider une démarche d'explicitation.

2.3. Connaissances explicites

Les deux paragraphes précédents montrent que pour déterminer le domaine d'intérêt d'un algorithme d'apprentissage, ou comparer différents algorithmes, nous ne pouvons pas faire l'économie d'une recherche de connaissances mettant explicitement en relation des caractéristiques des problèmes à traiter et des spécificités des algorithmes.

Ces connaissances *a priori* concernant les problèmes sont de nature très diverse. Il peut s'agir de connaissances génériques, comme le caractère quasi-linéaire ou lisse de la dépendance cible, ou de connaissances dépendantes du problème, comme la présence de symétries, de dépendances à moyen ou long terme, etc. Des discussions intéressantes concernant la typologie des connaissances *a priori* mais aussi leur origine peuvent être trouvées dans [Abu-Mostafa, 1995] et [Buntine, 1996].

Pour un algorithme, les connaissances explicites permettant d'évaluer son adéquation à un problème proviennent idéalement de preuves formelles, ou éventuellement d'expérimentations systématiques. Par exemple, le fait que le perceptron n'est capable

d'apprendre que des classes linéairement séparables peut être obtenu par démonstration, ou par une série d'expériences bien choisies. Nous remarquerons que pour arriver à une telle conclusion à partir de résultats expérimentaux, quelques problèmes de référence ne suffisent pas ; il faut commencer par identifier la caractéristique dont dépendent le plus les résultats et ensuite vérifier cette hypothèse par de nombreux essais effectués sur des données générées artificiellement, de façon appropriée [une référence à un "générateur de données et superviseur d'expériences" publié dans Modulad en 1997 ne déparerait pas, me semble-t-il? Je laisse à ton appréciation…] . Les connaissances implicites accumulées sur des problèmes de référence peuvent guider la recherche, mais ne servent que de point de départ dans cette démarche. Si pour un algorithme donné nous avons déterminé deux ensembles de problèmes tels que l'algorithme donne de très bons résultats sur le premier et de très mauvais sur le second, nous pouvons essayer d'extraire de ces deux ensembles les caractéristiques qui semblent encourager ou décourager l'utilisation de l'algorithme. Ces hypothèses seront ensuite validées (ou invalidées) par des essais systématiques sur des données artificielles.

Une démarche expérimentale symétrique peut être employée pour déterminer les caractéristiques d'un problème inconnu, à partir des résultats obtenus sur ce problème par différents algorithmes dont les spécificités sont bien comprises.

Les connaissances explicites concernant les caractéristiques des problèmes et les spécificités des algorithmes d'apprentissage nous permettent de sélectionner un algorithme adapté (ou de paramétrer un algorithme plus générique) pour le problème à traiter, et de déterminer le domaine d'intérêt d'un algorithme. Comme un exemple de sélection ou de paramétrage approprié, regardons quelques moyens dont nous disposons pour privilégier des dépendances « lisses » au cours d'une modélisation par des réseaux de neurones artificiels (RNA) :

1° Pour [Moody et Rögnvaldsson, 1996] le lissage correspond à des bornes sur les dérivées d'ordre *k* de la fonction de transfert ; les termes de régularisation obtenus à partir d'une telle contrainte pour des RNA non récurrents sont très généraux.

2° L'injection de bruit dans les exemples d'apprentissage ou dans les poids des connexions d'un RNA produit un lissage de la fonction de transfert (voir par exemple [Canu et al., 1995]).

3° Plutôt que d'employer un seul RNA comme modèle, nous pouvons développer un ensemble de RNA sur des ensembles d'apprentissage légèrement différents et faire une moyenne entre leurs sorties pour obtenir le résultat. [Riis et Krogh, 1995] constatent que ceci correspond à un lissage de la fonction de transfert globale.

Au-delà de ces usages, les connaissances explicites peuvent nous guider dans le développement d'algorithmes qui répondent à des caractéristiques spécifiques des problèmes à traiter. S'il est en général difficile de prouver que l'algorithme est optimal par rapport à la caractéristique ciblée, il est souvent possible de montrer qu'il élimine (ou atténue) des inconvénients dont souffrent nombre de ses concurrents (voir un exemple dans [Boné et al., 2000] ←[je laisse cette référence ?] oui).

3. Un site Web comme outil de gestion des connaissances

Dans la section précédente, nous avons essayé de mettre en évidence l'utilité des connaissances implicites et surtout explicites concernant les algorithmes d'apprentissage et les problèmes traités. D'où l'intérêt que présente un outil de gestion de ces connaissances comme aide à l'exploration systématique du domaine, ainsi que dans l'inspiration de travaux théoriques. Nous avons mis au point un tel outil sous la forme d'un site Web, qui est largement accessible et rend très simples les opérations de consultation ou d'enrichissement.

Notre site Web http://www.rfai.li.univ-tours.fr/validation a été mis en place avec des objectifs multiples, parmi lesquels nous pouvons mentionner :

1° Le site doit fournir des algorithmes d'apprentissage candidats (y compris leurs implémentations documentées) pour traiter les problèmes que les utilisateurs souhaitent résoudre, ainsi que des problèmes de référence pour ceux qui mettent au point de nouveaux algorithmes.

2° Le site doit encourager les bonnes pratiques dans l'utilisation des algorithmes d'apprentissage et la présentation des résultats.

3° Le site doit servir d'outil de gestion des connaissances en assurant l'accumulation, l'organisation, la présentation lisible, le partage et la mise à jour facile des connaissances implicites ou explicites. Rappelons le rôle important d'une accumulation des connaissances implicites dans l'initiation de l'exploration théorique ou expérimentale ayant pour objectif la connaissance explicite.

4° Le site doit permettre de lancer facilement des compétitions, avec évaluation automatique et indépendante des résultats par le site.

Afin d'atteindre ces objectifs, un certain nombre de facilités ont été présentes dans le site dès sa création :

1° L'exploration du contenu peut être faite selon deux profils, à travers les algorithmes ou à travers les problèmes disponibles. Chacun des profils donne ensuite accès à la consultation des résultats obtenus par les algorithmes sur les problèmes.

2° Pour les algorithmes, les problèmes et les résultats nous trouvons d'abord un niveau où l'information est synthétique (figures 1 à 3 [les trois figures sont nécessaires ? J'aurais aimé en mettre d'autres, mais il y a pas mal de choses qui devraient changer d'ici l'été…]). Ces vues synthétiques permettent d'atteindre les connaissances détaillées disponibles.

ACCUEIL ALGORITHMES PROBLEMES

Les algorithmes évalués

Numéro	Nom algorithme	Type algorithme	Description détaillée	Références	Résultats	Implémentation
25	BPTT	numérique, base déterministe, régression et classification, problèmes temporels	descriptif	bibliographie	résultats	accès implémentation
33	LEAD	numérique, base déterministe, régression et classification, problèmes temporels	descriptif	bibliographie	résultats	accès implémentation
34	CBAD	numérique, base déterministe, régression et classification, problèmes temporels	descriptif	bibliographie	résultats	accès implémentation

Figure 1 : Vue synthétique des algorithmes d'apprentissage.

ACCUEIL ALGORITHMES PROBLEMES

Les problèmes disponibles

Numéro	Nom problème	Type problème	Nb ex app	Nb ex test	Description détaillée	Références	Les données	Résultats
23	sunspots	numérique, temporel, régression	221	59 (35, 24)	descriptif	bibliographie	données	résultats
24	Fraser	numérique, temporel, régression	750	196	descriptif	bibliographie	données	résultats
28	Saint-Jean	numérique, temporel, régression multivariée	1000	440	descriptif	bibliographie	données	résultats
29	Eau	numérique, temporel, régression multivariée	-	-	descriptif	bibliographie	données	résultats
30	Fisher	numérique, temporel, régression multivariée	1000	461	descriptif	bibliographie	données	résultats

Figure 2 : Vue synthétique des problèmes disponibles.

ACCUEIL ALGORITHMES PROBLEMES

Les résultats

Problème	Algorithme	Auteur	Erreur app.	Ecart-type app.	Erreur test	Ecart-type test	Description expérience	Justification expérience	Références
sunspots	BPTT	Romuald BONE	0.0928	0.0183	0.1021, 0.3713	0.0136, 0.0228	description	justification	bibliographie
sunspots	LEAD	Romuald BONE	0.101	0.0002	0.096, 0.320	0.0008, 0.0045	description	justification	bibliographie
sunspots	CBAD	Romuald BONE	0.101	0.0062	0.094, 0.281	0.0063, 0.0338	description	justification	bibliographie

Figure 3 : Vue synthétique des résultats obtenus sur un des problèmes.

3° Les connaissances explicites décrivent les caractéristiques des algorithmes et des problèmes de la base, ainsi que les raisons qui ont conduit à employer tel algorithme avec tels paramètres pour la résolution de tel problème. Toutes ces

connaissances sont présentes dans les justifications et descriptifs détaillés accessibles à partir des trois vues synthétiques mentionnées.

4° Des pointeurs vers des implémentations documentées des algorithmes sont inclus. Les descriptions des résultats reprennent non seulement les valeurs de tous les paramètres utilisés, mais aussi les fichiers permettant de reproduire les résultats avec les implémentations disponibles.

5° Une recherche dans le texte, par mots-clés, est disponible à partir de la page d'accueil et donne un accès direct aux informations. [à implémenter, env. 1h…]

6° Pour insérer de nouveaux algorithmes, problèmes ou résultats dans la base, des formulaires à remplir en ligne sont disponibles. La possibilité d'introduire le contenu en format source HTML (avec des indications de mise en page ou des liens externes) dans la plupart des champs de saisie des formulaires donne une flexibilité supplémentaire.

7° De nombreux guides et notices explicatives sont disponibles à tous les niveaux, à la fois pour la consultation du site et pour l'enrichissement de la base.

Bien que le site soit actuellement parfaitement fonctionnel, différents développements sont en cours, comme la mise en œuvre de liens symboliques pour mieux structurer les connaissances, ou la gestion de forums de discussion associés au site. D'autres développements peuvent être envisagés afin de répondre aux demandes des utilisateurs.

Enfin, sous la forme d'extraits très succincts des multiples guides disponibles en ligne, voici quelques exemples de ce que le site permet de faire :

1° Afin de trouver des problèmes pour valider un algorithme d'apprentissage il suffit de i) sélectionner dans la liste des problèmes ceux dont le type est adéquat par rapport à l'algorithme ; ii) pour chacun de ces problèmes, consulter les résultats obtenus avec d'autres algorithmes et notamment la « Justification de l'expérience » ; iii) sélectionner les problèmes jugés intéressants par rapport aux caractéristiques de l'algorithme.

2° Pour choisir des algorithmes candidats pour un problème à traiter, i) chercher, dans la liste des problèmes disponibles sur le site, ceux qui sont d'un même type que le problème à traiter ; ii) pour chacun de ces problèmes, regardez les résultats obtenus avec différents algorithmes ; iii) à partir de la colonne « Justification de l'expérience », sélectionner les problèmes qui possèdent des caractéristiques communes avec le problème à traiter ; iv) pour chacun des problèmes retenus, choisir l'algorithme le plus approprié (faute de connaissances explicites suffisantes, celui qui donne les meilleurs résultats), et appliquer ces algorithmes au problème à traiter.

3° Afin de déterminer quelles caractéristiques des problèmes traités donnent l'avantage à un algorithme, i) choisir l'algorithme à étudier dans la liste des algorithmes dont l'évaluation est disponible sur le site ; ii) regarder les résultats obtenus par cet algorithme sur les différents problèmes sur lesquels il a été testé ;

iii) à partir de la « Justification de l'expérience », sélectionner les explications correspondant aux bons résultats de l'algorithme et déterminer leurs points communs ; les caractéristiques résultantes sont probablement celles qui donnent l'avantage à l'algorithme étudié ; iv) à partir de la « Justification de l'expérience », sélectionner les (éventuelles) explications correspondant aux mauvais résultats de l'algorithme, déterminer leurs points communs, les caractéristiques résultantes sont probablement celles désavantagent l'algorithme étudié.

4. Conclusion

Nous nous sommes intéressés au choix d'un algorithme d'apprentissage pour traiter un problème et à la validation de ces algorithmes. Dans cette perspective, nous avons mis en évidence l'utilité des connaissances implicites et surtout explicites concernant les algorithmes candidats et les problèmes traités. Pour avancer, les experts d'un domaine d'application doivent essayer d'expliciter les caractéristiques des problèmes qu'ils traitent et les informaticiens doivent utiliser explicitement ces caractéristiques dans la conception des algorithmes qu'ils proposent.

Nous avons développé un site qui peut servir d'outil de gestion des connaissances concernant les problèmes et les algorithmes d'apprentissage à partir de données. Cette base de connaissances peut constituer un lien entre ceux qui développent des algorithmes et ceux qui les utilisent. Les uns y trouveront des problèmes intéressants à traiter et des intuitions pour entreprendre des études théoriques. Les autres arriveront à une meilleure compréhension de leurs problèmes, et pourront repérer dans la base des algorithmes d'apprentissage adaptés.

Le plus difficile reste à faire, car c'est seulement l'accumulation de connaissances dans la base, grâce aux contributions des différents utilisateurs, qui rendra le site vraiment utile. [pour faire plus gai, on peut toujours mettre une blague à la coluche du genre: "un algo d'apprentissage, c'est un truc, tu lui donnes un problème, au truc, et, quand il te fournit une solution, le truc, tu comprends plus ton problème…" bon, on restera sérieux, j'ai rien trouvé d'intermédiaire!]

Bibliographie

Abu-Mostafa, Y. (1995) Hints, *Neural Computation* 7(4): 639-671.

Boné, R., Crucianu, M., Asselin de Beauville, J.-P. (2000) An Algorithm for the Addition of Time-Delayed Connections to Recurrent Neural Networks, *European Symposium on Artificial Neural Networks*, Belgique, avril 2000.

Buntine, W. (1996) Prior Probabilities, *NATO Workshop on Learning in Graphical Models*, Erice, Italy, September 1996 (http://www.ultimode.com/wray/refs.html).

Canu, S., Ding, X., Grandvalet, Y. (1995) Réseaux de neurones pour la prévision à un pas de temps, *Ecole Modulad*, 6-8 décembre 1995, Montpellier, p. 12.1-12.23.

Dietterich, T. G. (1998) Approximate statistical tests for comparing supervised classification learning algorithms, *Neural Computation* 10: 1895-1923.

Ehrenfeucht, A., Haussler, D., Kearns, M., Valiant, L. (1989) A general lower bound on the number of examples needed for learning, *Information and Computation* 82: 247-261.

Elisseeff, A., Paugam-Moisy, H., Guermeur, Y. (1999) Confidence intervals for multiclass discriminant models, *Rapport de Recherche*, ERIC, Université de Lyon 2, France.

Hoeffding, W. (1963) Probability inequalities for sums of bounded random variables, *Journal of the American Statistical Association* 58: 13-30.

Moody, J. (1994) Prediction risk and architecture selection in neural networks, V. Cherkassy, J. H. Friedman, H. Wechsler (eds.) *NATO ASI Series F*, Springer-Verlag, 1994.

Moody, J. E., Rögnvaldsson, T. S. (1996) Smoothing regularizers for projective basis function networks, *OGI CSE Technical Report* 96-006, Dept. of Computer Science and Engineering, Oregon Graduate Institute, Portland, Oregon, USA.

Riis, K., Krogh, A. (1995) Improving prediction of protein secondary structure using structured neural networks and multiple sequence alignments, *NORDITA preprint 95/34-S*.

Vapnik, V. N., Chervonenkis, A. Y. (1971) On the uniform convergence of relative frequencies of events to their probabilities, *Theory of Probability and Its Applications* 16: 264-280, 1971.

Vapnik, V. N. (1998) *Statistical Learning Theory*. John Wiley & Sons, N.Y., 1998.

Wolpert, D. H. (1996a) The lack of *a priori* distinctions between learning algorithms, *Neural Computation* 8: 1341-1390.

Wolpert, D. H. (1996b) The existence of *a priori* distinctions between learning algorithms, *Neural Computation* 8: 1391-1420.