UNIVERSITÉ DE PROVENCE - AIX-MARSEILLE I
U.F.R. de Mathématiques, Informatique et Mécanique

École Doctorale Mathématiques et Informatique de Marseille
E.D. numéro 184

THÈSE

pour obtenir le grade de

DOCTEUR DE L'UNIVERSITÉ DE PROVENCE
Spécialité : Informatique

présentée et soutenue publiquement par

Christophe Nicolas MAGNAN

le 12 décembre 2007

APPRENTISSAGE À PARTIR DE DONNÉES DIVERSEMENT ÉTIQUETÉES POUR L'ÉTUDE DU RÔLE DE L'ENVIRONNEMENT LOCAL DANS LES INTERACTIONS ENTRE ACIDES AMINÉS

devant le jury composé de :

Florence d'Alché-Buc	Professeur	*Rapporteur*
Antoine Cornuéjols	Professeur	*Rapporteur*
François Denis	Professeur	*Directeur de thèse*
Cécile Capponi	Maître de Conférences	*Directrice de thèse*
Alain Guénoche	DR CNRS	*Examinateur*
Yann Guermeur	CR CNRS	*Examinateur*

Remerciements

Je remercie sincèrement, et en tout premier lieu, François Denis, qui m'a fait découvrir l'apprentissage automatique et les passionnantes perspectives qu'offrent ce domaine de recherche. Dès les premiers cours qu'il a dispensés durant ma quatrième année universitaire, il m'a transmis sa passion et l'envie de me spécialiser dans cette discipline.

Je remercie encore François Denis, mais avec Cécile Capponi cette fois, pour m'avoir proposé de mener une thèse à leurs côtés, pour leur encadrement sérieux pendant ces trois années, et pour m'avoir proposé un sujet aussi intéressant, qui m'a permis de rejoindre un domaine que j'affectionne tout particulièrement, celui de la biologie. J'espère qu'ils garderont comme moi, un souvenir positif de cette expérience.

Je voudrais également remercier tous les chercheurs du CMI, quel que soit leur laboratoire, pour m'avoir accepté sans condition comme l'un des leurs, et ce, dès mon arrivée, ainsi que pour les nombreux moments agréables que j'ai passé avec eux. Merci également à tous les enseignants avec qui j'ai eu le plaisir de travailler dans le cadre du monitorat ou avec qui j'ai le plaisir de travailler actuellement dans celui d'un poste d'ATER.

Je remercie Florence d'Alché-Buc et Antoine Cornuéjols pour avoir accepté de rapporter cette thèse, pour la relecture attentive qu'ils en ont faite, et pour leurs rapports détaillés, qui me seront sans aucun doute d'une grande utilité pour la suite des travaux menés durant cette thèse. Merci également à Alain Guénoche d'avoir accepté de faire partie du jury de ma thèse, comme il l'avait été lors du jury de DEA, il y a trois ans de cela. Je remercie également Yann Guermeur d'avoir lui aussi accepté de faire partie du jury et pour m'avoir accepté dans le projet Genoto3D dont il était responsable.

Je remercie les personnels administratifs du CMI, pour tous les services qu'ils nous rendent chaque jour, mais également pour leur agréabilité, et pour toutes les franches rigolades qu'on a pu avoir sur des sujets de la plus haute importance scientifique bien sûr...

Merci également aux thésards du CMI pour les bons moments passés depuis trois ans, les pauses cafés semi-scientifiques, nos échanges instructifs, et je voudrais remercier plus particulièrement Clément et Lionel, compagnons d'arme depuis de longues années, encore toutes mes félicitations pour votre thèse.

Je remercie les membres de ma famille pour leur soutien important durant ces trois années. Je m'excuse auprès d'eux pour mes absences prolongées et ma disponibilité parfois reprochable. Je les remercie d'avoir accepté tout cela pour une cause qu'ils n'ont pas choisi.

Un grand merci à lolo, mon irremplaçable ami depuis vingt ans, pour tout ce qu'il m'apporte depuis toutes ces années, et pour toutes les bonnes soirées dont il a le secret.

Enfin, je remercie chaleureusement Gwladys, pour son amour et son soutien durant ces années. Je la remercie également de m'avoir attendu pour fêter l'obtention de son doctorat cet été au début de ma rédaction. La fête qui semble arriver n'en sera que plus joyeuse...

Un dernier merci à tous ceux que j'aurais maladroitement oublié...

Résumé. Nous étudions le problème bioinformatique de la prédiction de contacts ponctuels entre résidus distants sur la séquence d'une protéine, tels que les ponts disulfures ou salins, une étape encore non résolue du problème plus général de la prédiction de la structure 3D d'une protéine à partir de sa séquence primaire. L'étude de l'état de l'art sur ce problème a fait ressortir des questions sur la modélisation de ce problème ainsi que sur le rôle de l'environnement local des acides aminés appariés dans la formation de ces contacts. Plusieurs considérations biologiques d'une part, et des expérimentations d'autres part, montrent la nécessité d'étudier des contextes d'apprentissage jusqu'ici peu connus et peu étudiés pour répondre à ces questions. Le premier est un cas particulier de l'apprentissage semi-supervisé binaire dans lequel on suppose que les exemples classés dont on dispose appartiennent uniquement à une seule classe, nous l'appelons *apprentissage semi-supervisé asymétrique*. Le second cadre d'apprentissage étudié est une extension de l'apprentissage avec bruit de classification, noté *CN*, dans lequel on suppose que les données de chacune des deux classes sont corrompues par un bruit de classification constant par classe avant d'être observées, nous notons ce modèle de bruit *CCCN*. Nous montrons que ces deux contextes d'apprentissage sont mal posés dans le cadre général de l'apprentissage statistique, mais que certaines hypothèses sur les distributions sous-jacentes permettent de les rendre bien posés, comme par exemple l'hypothèse que les distributions conditionnelles à chacune des classes sont des distributions produits. Des adaptations de méthodes connues de l'apprentissage à ces contextes sont proposées : l'algorithme naïf de Bayes et l'algorithme du perceptron. Ces nouveaux algorithmes ont été expérimentés puis utilisés pour tenter de répondre aux questions biologiques initialement posées.

Mots clés : prédiction de la structure 3D des protéines, ponts disulfures et salins, affinité locale, apprentissage statistique supervisé et semi-supervisé, bruit de classification.

Abstract. The tridimensional structure of proteins is constrained or stabilized by some interactions between distant amino acids in the primary sequences. An accurate prediction of these bonds by machine learning methods may reduce the set of feasible conformations for a protein and may be an important step forward for the prediction of the 3D structure from primary sequences. A review of the literature raises questions about the role of the neighbourhood of bonded amino acids in the formation of these bonds and about the modeling of this problem. Some biological considerations and experiments show that we have to investigate uncommon learning frameworks to answer these questions. The first one, that we call *asymmetrical semi-supervised learning*, is a particular case of binary semi-supervised learning, in which the only labelled data to learn from belong to one class, and the second one considers that the data at hand are subject to *class-conditional classification noise* (*CCCN*), a generalization of the *uniform classification noise* (*CN*). We show that learning in these frameworks leads to ill-posed problems. We give some assumptions that make these problems well-posed. We propose adaptations of well-known learning methods, namely naive Bayes algorithm and the perceptron, to these learning frameworks. We test these methods and apply them to try to answer the questions on the biological problem considered in this study.

Keywords : proteins 3D structure prediction, disulfide and salt bridges, local affinities, statistical supervised and semi-supervised learning, classification noise.

Table des matières

Table des figures

Table des algorithmes

Liste des tableaux

Introduction

Depuis plusieurs années, le domaine de la biologie fournit à plusieurs disciplines des problèmes cruciaux qui permettent aux chercheurs de ces disciplines de faire progresser à la fois la connaissance sur ces problèmes biologiques et l'état de l'art de leur propre domaine de recherche. Elle crée donc, au travers des problèmes qu'elle pose, de la connaissance dans d'autres disciplines. En particulier, la prédiction de la structure tridimensionnelle des protéines à partir de leur séquence primaire est un challenge de forte actualité qui rassemble aussi bien les biologistes que les chimistes, les mathématiciens ou les informaticiens.

Si les récentes techniques de séquençage des génomes ont permis la découverte de près de 5 millions de séquences protéiques, les techniques expérimentales de modélisation de la structure tridimensionnelle, par résonance magnétique nucléaire ou par diffraction des rayons X sur les cristaux de protéines, ne permettent pas encore d'obtenir les structures de ces protéines avec une telle rapidité puisqu'environ 45.000 structures seulement sont connues (août 2007). De plus, déterminer ces structures expérimentalement est une tâche longue, difficile, coûteuse et aux résultats non garantis. C'est la raison pour laquelle de nombreux chercheurs tentent de mettre au point de nouvelles méthodes pour modéliser la structure des protéines à partir des structures déjà déterminées expérimentalement.

Parmi les différentes approches bioinformatiques de la modélisation de la structure des protéines, une d'entre elles consiste à prédire, à partir de la séquence d'acides aminés qui constitue une protéine, et par des méthodes d'apprentissage automatique, différentes parties de la structure, comme des motifs structuraux fréquents, afin d'obtenir un ensemble réduit de conformations spatiales envisageables pour une protéine. Les protéines dont la structure est connue forment alors la base de connaissance indispensable à l'apprentissage de fonctions prédictives performantes pour la tâche considérée.

L'exemple le plus marquant de ces dernières années est sans aucun doute la prédiction de la structure secondaire des protéines, une structure intermédiaire entre la séquence des acides aminés et la structure 3D, constituée de formes régulières et fréquentes dans les protéines, par exemple les brins β et les hélices α. Elle a fait l'objet de nombreux travaux qui permettent une prédiction précise de ces motifs à partir de la séquence.

La prédiction d'autres parties de la structure forme encore aujourd'hui un ensemble de problèmes ouverts. Par exemple, la prédiction de contacts ponctuels entre deux résidus distants de la séquence d'une protéine, tels que les ponts disulfures ou les ponts salins, est toujours un problème d'actualité, même si ces contacts sont parfois davantage considérés comme une conséquence de la structure que comme une cause. Ces contacts ont un rôle dans la structure, ils permettent par exemple de la stabiliser. Savoir les prédire

précisément serait un grand pas en avant car cela réduirait l'ensemble des conformations envisageables d'une protéine en ajoutant une contrainte sur sa structure tridimensionnelle. C'est dans le cadre de ces travaux bioinformatiques, cherchant à mettre au point des méthodes pour prédire les contacts distants entre deux résidus d'une protéine, que se situent les travaux menés et présentés dans ce mémoire de thèse.

Les ponts disulfures, qui relient deux acides aminés d'une protéine, des cystéines, par une liaison covalente, la plus forte des interactions entre deux résidus, sont les contacts ponctuels les plus étudiés. Stables et conservés, ils constituent une cible de choix pour les chercheurs. Une étude approfondie de la littérature sur la prédiction des ponts disulfures indique qu'une information est systématiquement utilisée pour prédire ces ponts : l'environnement local des cystéines, c'est-à-dire les acides aminés situés autour des cystéines sur la séquence primaire de la protéine. Les premiers travaux de cette thèse (Chapitre 3) sont voués à étudier ce choix d'information pour prédire les ponts.

Pour les biologistes du domaine, il n'est pas clair que cette information joue un rôle dans la formation de ponts, en particulier de ponts disulfures [Geourjon, 2005, Gibrat, 2005]. En effet, certains d'entre eux considèrent que ces contacts ne sont qu'une conséquence de la structure et que l'environnement local des cystéines n'aurait peut-être pas d'influence sur le choix des partenaires des cystéines. Savoir si l'environnement local de résidus en contact joue un rôle dans la formation de ces interactions entre résidus distants d'une protéine est très important car c'est l'information la plus utilisée pour la prédiction de la plupart des contacts dans les protéines, et qui sera certainement utilisée pour la prédiction d'autres contacts encore non étudiés. C'est la raison pour laquelle cette question est devenue le thème central de cette thèse, ne concernant alors plus seulement les ponts disulfures, mais toute interaction ponctuelle entre deux résidus distants d'une protéine.

Considérer que les environnements locaux de deux résidus potentiellement en contact jouent un rôle dans la formation ou non de ce contact, c'est faire l'hypothèse que ces paires d'environnement ont une certaine propension, une sorte d'*affinité*, à se retrouver appariées par ces résidus. Plusieurs considérations biologiques d'une part (Chapitre 3), et une formalisation de cette notion d'affinité entre deux segments d'une protéine d'autre part (Chapitre 4), montrent que si une telle information existe et intervient dans la formation de contacts locaux, alors les observations d'acides aminés appariés et non appariés dont on dispose au travers des structures de protéines déjà déterminées expérimentalement, doivent permettre de la détecter et de l'extraire. Toutefois, cette formalisation montre également que les observations dont on dispose n'en fournissent qu'une vue indirecte. En effet, ces observations seraient alors des exemples bruités d'une fonction d'affinité entre environnements locaux. Le modèle de bruit induit par cette formalisation apparaît rarement dans la littérature [Blum and Mitchell, 1998], c'est une généralisation directe d'un modèle de bruit de classification connu dans le domaine de l'apprentissage : le *bruit de classification uniforme* [Angluin and Laird, 1988], généralement noté CN. Nous appelons ce modèle *bruit de classification conditionnel à chaque classe*, que nous notons CCCN. On montre alors que si la fonction d'affinité entre environnements locaux peut être représentée par une fonction apprenable à partir de données sujettes à du bruit de classification CCCN, alors on doit pouvoir l'approcher de manière arbitrairement proche, et donc en prouver l'existence, en supposant que l'on dispose de suffisamment d'exemples et indépendamment de toute connaissance sur le processus qui apparie les acides aminés dans une protéine.

De nombreux travaux ont proposé des études de l'apprentissage à partir de données corrompues par du bruit uniforme CN. Les méthodes d'apprentissage issues de ces travaux sont reconnues pour leur efficacité mais sont basées sur l'hypothèse de la présence d'un bruit de classification de type CN et ne permettent pas d'être utilisées dans le modèle CCCN que nous introduisons. C'est la raison pour laquelle nous avons mené trois études théoriques, algorithmiques et expérimentales dans ce contexte d'apprentissage (Chapitres 5 et 6, Annexe C). Ces études montrent que la présence d'un tel bruit rend généralement mal posé le problème d'apprentissage correspondant mais que certaines hypothèses sur les distributions sous-jacentes au problème permettent de rendre possible sa résolution. Notamment, l'apprentissage de distributions produits (Chapitre 5) et de séparateurs linéaires (Chapitre 6) peut se faire de façon efficace à partir de données affectées par du bruit CCCN.

Ces travaux ont permis de montrer que le protocole proposé pour détecter la présence d'une information locale impliquée dans la formation de contacts entre acides aminés d'une protéine peut être mis en place et expérimenté, par exemple avec les méthodes proposées lors de ces travaux. Une implémentation JAVA du protocole et des algorithmes proposés dans cette thèse a été menée pour cela (Annexe B). Les derniers travaux de cette thèse (Chapitre 7) ont donc consisté à expérimenter le protocole sur des données réelles issues de la base de données PDB (Annexe A). Ces expériences prouvent la validité du protocole et ouvrent des perspectives intéressantes aux travaux de thèse présentés dans ce mémoire.

Première partie

Notions de base et situation du sujet de recherche

Chapitre 1

La prédiction d'interactions locales dans les protéines

Sommaire

Ce chapitre décrit le contexte bioinformatique dans lequel se sont déroulés les travaux présentés dans ce mémoire : la prédiction de la structure tridimensionnelle des protéines et en particulier la prédiction d'interactions locales entre deux résidus distants d'une protéine comme les ponts disulfures ou les ponts salins.

La prédiction *de novo* de la structure tridimensionnelle des protéines regroupe de nombreux chercheurs de différents domaines. Cette tâche complexe consiste à prédire, par l'intermédiaire de méthodes bioinformatiques, certaines parties de la structure des protéines à partir de leur séquence primaire pour proposer un ensemble de conformations spatiales envisageables pour une protéine dont on ne connaît pas la structure. Certaines parties de cette structure tridimensionnelle ont fait l'objet de beaucoup d'attention et sont aujourd'hui prédites avec une grande précision. D'autres restent encore des problèmes ouverts qui continuent à retenir l'attention de plusieurs groupes de recherche. C'est notamment le cas de la prédiction de contacts locaux entre deux résidus distants comme les ponts disulfures et les ponts salins. Ces ponts contraignent ou stabilisent la structure des protéines, leur prédiction est donc un challenge aux enjeux importants pour le problème plus général de la prédiction de la structure spatiale des protéines.

La première partie de ce chapitre (Section 1.1) présente la problématique de la prédiction de la structure tridimensionnelle des protéines : pourquoi cette structure est-elle si importante ? pourquoi cherche-t-on à la prédire ? par quels moyens ? etc. Cette section décrit notamment comment l'apparition des méthodes de séquençage automatique des génomes a entraîné la nécessité de mettre en œuvre des méthodes automatiques de prédiction de la structure malgré l'existence de méthodes expérimentales qui permettent de déterminer avec précision cette structure. La structure secondaire, intermédiaire entre la séquence des acides aminés et la conformation spatiale de la protéine, est actuellement le problème qui a retenu le plus d'attention et sur lequel de nombreux travaux ont été menés. Ils permettent aujourd'hui d'en obtenir une prédiction très précise. La section 1.1.5 en retrace l'historique car ces études ont eu un impact important sur les travaux de prédiction d'autres parties de la structure et en particulier sur ceux qui portent sur la prédiction des contacts locaux ponctuels, tels que les ponts disulfures ou les ponts salins, qui sont au cœur de l'étude menée dans cette thèse.

La seconde partie de ce chapitre (Section 1.2) présente les différentes interactions locales ponctuelles entre deux acides aminés distants sur la séquence d'une protéine. Parmi ces contacts, les ponts disulfures se démarquent des autres car cette interaction entre deux cystéines, l'un des vingt acides aminés à partir desquels les protéines sont formées, est plus forte que les autres. En effet, les résidus sont liés par une liaison covalente qui nécessite beaucoup d'énergie pour être brisée, elles stabilisent donc la structure des protéines. C'est la raison pour laquelle ces ponts ont fait l'objet de beaucoup d'attention et les travaux proposant des méthodes pour les prédire sont présentés dans cette section. Ces travaux forment le point de départ de l'étude menée durant la thèse présentée dans ce mémoire. D'autres interactions sont considérées dans cette thèse et sont présentées dans ces préliminaires. C'est notamment le cas des ponts salins, des interactions ioniques faibles entre deux résidus chargés d'une protéine. A la différence des ponts disulfures, on ne trouve pas de travaux proposant des méthodes pour leur prédiction à partir de la séquence. Toutefois, ces interactions présentent de fortes similarités avec les ponts disulfures et en particulier sur la modélisation du problème. La section 1.2.2 présente brièvement ces interactions.

1.1 Prédiction *de novo* de la structure 3D des protéines

En 1972, Christian Boehmer Anfinsen obtient le prix Nobel de chimie [Anfinsen, 1973] pour « ses travaux sur la ribonucléase et spécialement pour ceux concernant la connexion entre la séquence des *acides aminés* et l'*activité biologique conformationnelle* ». C'est 11 ans avant, en 1961, qu'il démontre que la ribonucléase, une enzyme qui catalyse la rupture des liaisons entre certains nucléotides de l'ARN, pouvait revenir à sa conformation initiale après dénaturation de celle-ci tout en préservant son activité enzymatique [Anfinsen et al., 1961]. Ce résultat suggère que la majeure partie de l'information nécessaire à une protéine pour obtenir sa conformation spatiale est encodée dans sa seule *structure primaire* : la séquence en acides aminés qui la définit. Ce résultat aura de nombreuses répercutions et en particulier celle de justifier des travaux pour tenter de prédire la structure à partir de la séquence. C'est justement dans le cadre de ces travaux *bioinformatiques* qui visent à mettre au point des méthodes de prédiction de la structure des protéines à partir de la séquence primaire que s'inscrit le travail présenté dans cette thèse.

1.1.1 Protéine, structure et fonction physiologique

Les protéines sont les plus nombreuses des molécules organiques des êtres vivants. Elles sont responsables de la totalité des activités cellulaires : régulation, catalyse, transport, communication, reconnaissance et bien d'autres. Toutes les protéines sont construites sur le même modèle : ce sont des macromolécules qui résultent de la mise bout à bout de quelques dizaines à quelques centaines de résidus élémentaires, les acides aminés. En théorie, une protéine peut adopter une infinité de conformations différentes et pourtant, même dénaturée, elle reprend spontanément sa forme native lorsqu'on la place dans son milieu naturel (Exemple en Figure 1.1). Une conformation spécifique semble donc être privilégiée dans un milieu et une situation donnés, même si on sait que cette conformation n'est ni rigide ni unique car elle peut varier dans le temps sous certaines conditions.

Les fonctions remplies par une protéine sont étroitement liées à sa conformation. En effet, selon sa forme, une protéine pourra se fixer à d'autres molécules et permettre une transformation chimique ou tout autre type d'interaction moléculaire [Branden et al., 1996]. La structure d'une protéine détermine donc quelle(s) propriété(s) physiologique(s) est (sont) assurée(s) par cette protéine. Savoir quel est le rôle principal de chaque protéine serait un grand pas en avant vers la connaissance et la compréhension du vivant : son fonctionnement, son évolution, ses dysfonctionnements, comment y remédier.

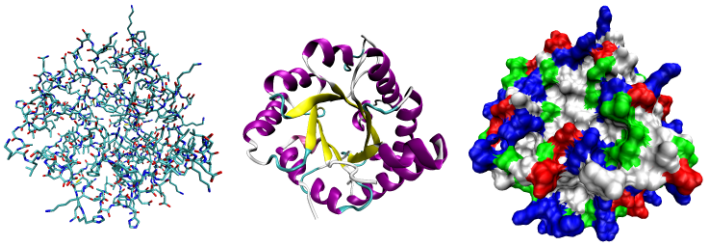


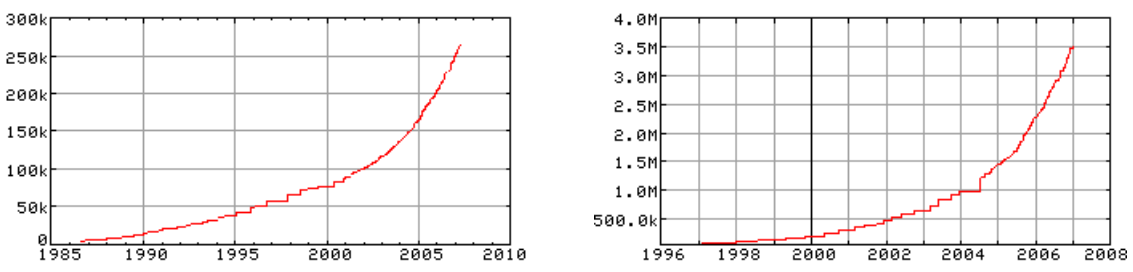
Figure 1.1. Trois vues distinctes d'une protéine (identifiant PDB : `1TIM`) obtenues par le logiciel VMD [Humphrey et al., 1996]. De gauche à droite : vue des atomes, vue de la structure secondaire (style cartoon), vue de la surface de la protéine.

1.1.2 L'incroyable évolution de la découverte de séquences protéiques

Au début des années 50, Frederick Sanger révèle la première séquence primaire d'une protéine : l'hormone insuline [Sanger and Thompson, 1953]. Il reçut pour cela le prix Nobel de Chimie en 1958. Depuis, le nombre de séquences protéiques découvertes ne cesse de croître et surtout depuis la fin des années 90 avec l'apparition des méthodes de séquençage des génomes. En 1997, le génome d'*Escherichia Coli* est le premier entièrement séquencé. En 1999, le séquençage du chromosome humain 22 s'est achevé, suivi quelques années plus tard, en 2004, par la publication de l'intégralité du génome d'une personne [Human Genome Sequencing Consortium International, 2004]. Des milliers de protéines sont découvertes à partir de ces génomes et sont ajoutées quotidiennement aux bases de données de séquences protéiques.

La base de données centrale de ces séquences protéiques est *UniProt-KB* (*Universal Protein Resource - KnowledgeBase*) [The UniProt Consortium, 2007]. Elle contient de nombreuses séquences annotées (fonction(s), domaines, structure secondaire, quaternaire, similarités à d'autres protéines, variantes, etc), permettant ainsi une utilisation pertinente de ces données. Elle est constituée de deux bases de données qui diffèrent par l'origine des annotations [Boeckmann et al., 2003] : manuelle (base de données *Swiss-Prot*), effectuées par des communautés d'experts, ou automatique (base de données *TrEMBL*), attribuées par des méthodes algorithmiques. Ces deux bases de données sont gérées par l'*Institut Suisse de Bioinformatique* (SIB) et l'*Institut Européen de Bioinformatique* (EBI).

La base *Swiss-Prot* est respectée pour sa haute qualité d'annotations et son faible niveau de redondance. La figure 1.2(a) donne l'évolution du nombre de séquences dans cette base depuis sa création en 1986. Elle contient actuellement 267.354 séquences annotées manuellement (*Swiss-Prot Release 52.5*). Cela montre l'intérêt croissant pour la constitution d'une telle base de connaissance et permet de constater les progrès effectués dans ce domaine. La base *TrEMBL* (*Translated EMBL*) est complémentaire à *Swiss-Prot*. Elle est construite à partir de la banque de séquences génétiques *EMBL* [Kulikova et al., 2007]. Les séquences codantes de cette base sont extraites puis traduites pour obtenir les protéines codées par ces séquences. Les protéines obtenues sont ensuite annotées par des méthodes algorithmiques en attendant d'être étudiées par des experts puis intégrées dans *Swiss-Prot*. La figure 1.2(b) donne l'évolution du nombre de séquences dans *TrEMBL* depuis sa création en 1997. Elle contient actuellement 4.361.897 séquences (*TrEMBL Release 35.5*). Le début de la période de séquençage intensif des génomes est très visible sur la courbe.



(a) (b)

Figure 1.2. Evolution du nombre de séquences annotées (a) manuellement dans la base *Swiss-Prot* depuis 1986 (b) automatiquement dans *TrEMBL* depuis 1997.

1.1.3 Détermination expérimentale de la structure

L'abondance des séquences protéiques découvertes montre la diversité du vivant mais également la nécessité de déterminer les fonctions de ces protéines pour comprendre ou mieux comprendre les phénomènes physiologiques et pouvoir trouver des solutions à certaines maladies. Cette nécessité se traduit par le besoin de connaître la *conformation spatiale* des protéines. Deux méthodes expérimentales sont principalement utilisées pour cela : par *diffraction des rayons X* sur les *cristaux de protéines* et par *résonance magnétique nucléaire* (RMN), le terme nucléaire faisant uniquement référence au noyau des atomes. Ces deux techniques permettent d'obtenir une description spatiale des atomes dans l'espace et donc de reconstruire la structure tridimensionnelle des protéines.

Ces méthodes ne sont pas récentes. La première voit le jour en 1912 par Max Von Laue qui découvre que les cristaux diffractent un pinceau de rayons X et donne naissance par cette découverte à la radiocristallographie. En 1926, J.B. Sumner parvint à cristalliser pour la première fois une enzyme et c'est en 1959 que John Kendrew proposa la première structure d'une petite protéine, la myoglobine. La RMN, plus récente, vient essentiellement des découvertes d'Isidor Isaac Rabi en 1938 qui découvre le phénomène de résonance magnétique sur des jets moléculaires et de Felix bloch et Edward Mills Purcell qui révèlent de façon rigoureuse la notion de fréquence de résonance en 1946. Ce n'est qu'à partir de 1973 que Paul Lauterbur propose les premières applications de la RMN en imagerie, la plus célèbre aujourd'hui étant l'IRM (imagerie par résonance magnétique).

Les structures déterminées par ces deux méthodes (et la microscopie électronique pour une faible part) sont centralisées dans la *Protein Data Bank* (PDB) [Berman et al., 2000]. Cette base a été créée en 1971 et contenait à l'origine 7 structures de protéines. A partir des années 80, le nombre de structures déterminées par ces méthodes a rapidement augmenté grâce à l'amélioration des techniques cristallographiques et à la RMN. La figure 1.3 donne l'évolution du nombre de structures contenues dans la PDB depuis 1972 qui en contient actuellement un peu plus de 45.000.

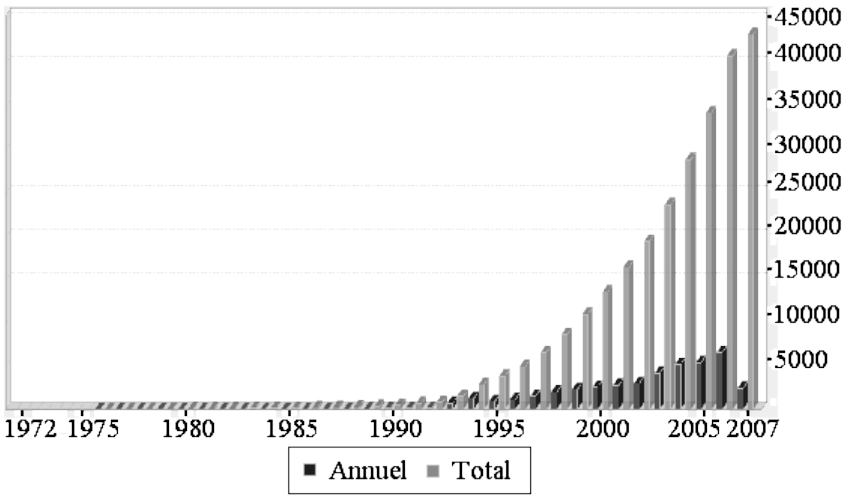


Figure 1.3. Evolution du nombre de structures tridimensionnelles de protéines contenues dans la base de données *Protein Data Bank* (PDB) depuis 1972.

Cette évolution rapide n'est toutefois pas à la hauteur de celle de la découverte des séquences protéiques. En effet, cela représente un peu plus de 16% du nombre de séquences de la base *Swiss-Prot* mais un peu moins de 1% du nombre de séquences de la base *TrEMBL*. En plus de ce problème, les deux méthodes expérimentales imposent des contraintes assez importantes :

- la diffraction des rayons X sur les cristaux de protéines est peu coûteuse et bien maîtrisée mais c'est la cristallisation des protéines qui est une tâche délicate. En effet, les cristaux sont fragiles et peuvent ne pas être parfaits à cause de la flexibilité moléculaire des protéines, le phénomène s'aggravant avec la taille des protéines. Toutefois, c'est la méthode la plus utilisée ($\simeq$ 85% des structures dans la base PDB)

- la RMN ne nécessite pas de cristaux de protéines mais présente trois inconvénients majeurs : la précision de la structure obtenue, le temps nécessaire pour son obtention et surtout, le coût de l'installation et du fonctionnement [Stevens, 2003]. Elle a néanmoins permis de déterminer un peu plus de 14% des structures de protéines de la base PDB.

1.1.4 Modélisation de la structure par d'autres moyens

Toutes ces considérations d'une part, et l'obtention d'un nombre tout de même important de structures d'autre part, ont entraîné de façon tout à fait naturelle de nouveaux travaux de recherche ayant pour objectif de proposer des méthodes différentes pour modéliser la structure spatiale des protéines.

On compte deux grandes familles de telles méthodes : celles qui tentent de déterminer la structure de nouvelles protéines en se basant sur les conformations spatiales des protéines contenues dans des bases de données telles que la PDB, et celles qui tentent de déterminer la structure par d'autres moyens et d'autres données, on parle souvent de méthodes *ab initio* (approches principalement physico-chimiques) ou *de novo* (approches bioinformatiques).

Les méthodes de recherche par homologie et les méthodes de *protein threading* [Bowie et al., 1991, Miller et al., 1996, Higgins and Taylor, 2000], essentiellement basées sur le constat que la structure est plus conservée que la séquence [Chothia and Lesk, 1986], sont des méthodes de modélisation comparative qui cherchent à détecter, par des méthodes d'alignement – séquences-séquences ou séquences-structures –, des similarités entre une nouvelle protéine ou portion de protéine et celles déjà connues. Ce type d'approche est très développé et obtient de bonnes performances même si deux problèmes principaux se posent encore aujourd'hui. Ils sont liés au principe même de la recherche par homologie : il faut avoir une base de données qui contienne un ensemble exhaustif de structures ou motifs, ce qui n'est pas encore le cas même si la découverte de nouveaux motifs structuraux ralentit depuis plusieurs années, mais également le problème des exemples qui dérogent à la règle : des séquences proches avec des structures très différentes et inversement des structures proches mais des séquences très différentes. Ces méthodes sont néanmoins très efficaces pour des protéines homologues aux protéines dont la structure est connue. Les travaux effectués dans cette thèse ne concernant pas ce type d'approche, il ne sera pas développé plus amplement.

Une autre catégorie importante de méthodes pour déterminer la structure d'une protéine est la catégorie des méthodes *ab initio* [Bonneau and Baker, 2001, Bonneau et al., 2001, Hardin et al., 2002]. Ces méthodes tentent de reproduire le processus de repliement de la protéine sous des considérations principalement géométriques et énergétiques. La protéine se stabilisant dans une conformation énergétiquement stable, il est naturel de tenter de trouver dans l'espace des configurations envisageables pour une protéine, celles qui minimisent les fonctions d'énergies et qui respectent les contraintes physiques des atomes. Ces méthodes souffrent néanmoins de la taille de l'espace de recherche particulièrement élevée et de leur forte sensibilité aux paramètres qui sont utilisés. C'est actuellement un domaine de recherche très actif et très prometteur.

Enfin, une dernière grande catégorie de méthodes pour déterminer la structure des protéines vient compléter les méthodes expérimentales, par homologie ou *ab initio* : les méthodes *de novo* de prédiction de la structure à partir de la séquence primaire. On fait souvent référence par ce terme à des méthodes bioinformatiques de reconstruction de la structure globale de la protéine à partir de différentes parties de cette structure déterminées par des méthodes d'apprentissage automatique. En effet, plusieurs motifs structuraux sont fréquents dans les protéines et comme on sait par ailleurs que la majeure partie de l'information nécessaire pour prédire ces motifs est contenue dans la séquence primaire des protéines, il est naturel de tenter de les prédire avec des méthodes d'apprentissage automatique, d'autant plus que les bases de données contiennent de nombreuses structures qui peuvent servir d'exemples d'apprentissage.

L'objectif est extrêmement ambitieux car la tâche est complexe. Elle se décompose en de nombreux sous-problèmes, correspondant à différentes informations sur la structure, qui peuvent se révéler difficiles à résoudre même individuellement. Plusieurs facteurs sont responsables de cette difficulté : la structure complexe des données, la quantité d'exemples disponibles, une part de non-déterminisme dans la tâche de prédiction, l'intégration d'informations hétérogènes (locales et globales par exemple), le manque d'informations biologiques sur certains phénomènes, des méthodes d'apprentissage parfois inadaptées, et bien d'autres facteurs. Néanmoins, cet objectif est réaliste car chaque nouvelle information que l'on peut prédire avec précision sur la structure restreint un peu plus l'espace des configurations envisageables pour une protéine dont on cherche la structure et rapproche de l'objectif de la prédiction complète de la structure à partir de sa séquence primaire.

Plusieurs approches combinées de ces méthodes de modélisation de la structure ont également été proposées. Parmi les plus connues, on peut notamment citer la méthode *Rosetta* [Simons et al., 1999, Bonneau et al., 2001, Bradley et al., 2003]. Cette méthode de prédiction de la structure des protéines combine des méthodes de *protein threading* avec des méthodes de prédiction *ab initio*. Elle a en particulier connu du succès de par ses performances aux campagnes de prédiction de la structure des protéines organisées tous les deux ans : *CASP*. C'est notamment la méthode qui a obtenu les meilleurs scores à la campagne CASP 4 (2000).

Les campagnes de prédiction de la structure 3D des protéines CASP permettent tous les deux ans d'évaluer l'évolution des différentes méthodes de prédiction de la structure en permettant aux auteurs de ces méthodes de les comparer sur un ensemble sélectionné de structures de protéines non publiées.

1.1.5 L'exemple de la prédiction de la structure secondaire des protéines

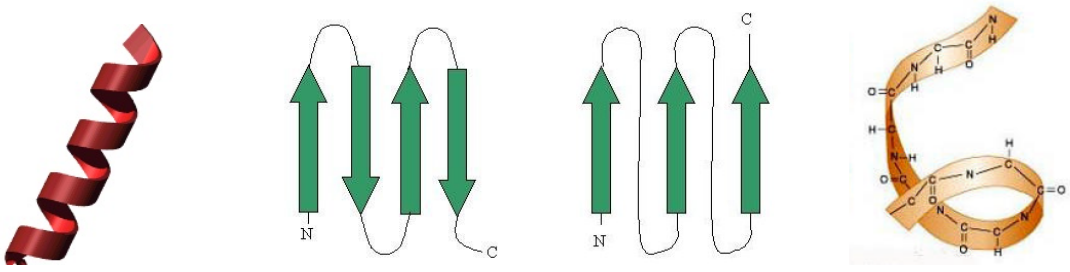
Parmi tous les travaux qui proposent des méthodes de prédiction de certaines parties de la structure, l'exemple le plus marquant est sans aucun doute la prédiction de la structure secondaire des protéines. Cette structure est intermédiaire entre la séquence primaire de la protéine et sa conformation spatiale. Elle contient des éléments structuraux réguliers et fréquents dans les protéines, en particulier les hélices α, les brins β et les coudes (*coil*). Leur forme caractéristique est illustrée par les images de la figure 1.4.

La structure secondaire (2D) fournit une empreinte de la structure spatiale d'une protéine. Sa prédiction est une étape importante pour la tâche globale, d'autant plus que la structure 3D est très conservée et donc en partie déterminée par sa structure 2D.

Les premières études menées sur la formation des brins β se sont principalement concentrées sur l'obtention de la propension de chacun des 20 acides aminés à se trouver apparié dans un brin β [Smith et al., 1994, Minor and Kim, 1994a, Minor and Kim, 1994b, Smith and Regan, 1995, Merkel and Regan, 1998]. Les expériences menées dans ces travaux consistaient à sélectionner une petite protéine contenant un feuillet β et à créer des mutants de cette protéine, en remplaçant dans chaque mutant un acide aminé impliqué dans le feuillet original pour observer si le nouveau résidu permettait la formation du feuillet dans la protéine modifiée ou non. Les résultats obtenus montrent que certains résidus favorisent ou défavorisent la formation de feuillets β.

Des expériences similaires ont été menées par la suite pour déterminer la propension de toute les paires d'acides aminés à être appariées dans un feuillet β [Wouters and Curmi, 1995, Smith and Regan, 1997, Hutchinson et al., 1998]. Les expériences menées dans ces travaux permettent de constater que certaines paires d'acides aminés sont favorisées dans les brins β, et que certaines paires défavorisent la formation de brins β.

Rapidement, la corrélation entre les paires de résidus appariés dans les brins β et leur *environnement local*, c'est-à-dire les résidus voisins sur la séquence de ces paires d'acides aminés appariés, a fait l'objet de plusieurs études [Minor and Kim, 1996, Zaremba and Gregoret, 1999, Merkel et al., 1999, Mandel-Gutfreund et al., 2001].



(a) (b) (c) (d)

Figure 1.4. Aspect géométrique des principaux motifs de structure secondaire : (a) Hélice α (b) Brins β antiparallèles (c) Brins β parallèles (d) Coude (*coil*).

Les conclusions de ces études diffèrent sur l'existence d'une corrélation entre l'environnement local des résidus et la formation de feuillets β entre ces résidus, certains mettant en évidence une corrélation évidente, d'autres montrant que cette corrélation n'est pas significative. Toutes les études citées ici sont les références des travaux qui ont proposé des méthodes de prédiction des brins β dans les protéines à partir de leur séquence primaire.

Dès la fin des années 90, les premiers travaux sur la prédiction des résidus impliqués dans un brin β à partir de la séquence primaire [Krogh and Riis, 1996, Frishman and Argos, 1996, Zhang et al., 1998] proposent déjà des méthodes obtenant des performances sur la tâche de prédiction supérieure à 70% de résidus correctement prédits. Certains prédisent également les hélices α dans le même temps.

Au début des années 2000, l'amélioration des méthodes d'apprentissage et les connaissances supplémentaires obtenues sur les brins β et les hélices α ont permis une nette progression dans les performances de prédiction de ces éléments de structure secondaire, certains proposant même la prédiction du registre - quel(s) brin(s) β s'apparie(nt) avec tel autre brin β - en plus de la prédiction des brins. La plupart des méthodes permettent également la prédiction des résidus impliqués dans une hélice α et dans des repliements spécifiques en forme de coudes. Ces méthodes obtiennent alors des performances globales sur tous ces éléments de structure secondaire proches des 85% [Baldi et al., 2000, Bradley et al., 2001, Baldi et al., 2003, Steward and Thornton, 2002, Sen, 2003].

Depuis, les méthodes n'ont cessé de s'améliorer permettant aujourd'hui une prédiction très précise ($> 90\%$) des résidus impliqués dans les brins β et les hélices α [Ruan et al., 2005, Cheng and Baldi, 2005, Cheng and Baldi, 2007].

Les travaux sur la prédiction de la structure secondaire des protéines ont joué un grand rôle dans les travaux sur la prédiction d'autres éléments de la structure des protéines, comme ceux sur la prédiction des ponts disulfures et des ponts salins. Ils ont notamment permis d'établir la liste des attributs prédictifs importants pour ce problème et l'impact de chacun d'eux sur la qualité de l'apprentissage effectué. Par exemple, tous les travaux intègrent l'environnement local (Section 1.2.1) des acides aminés potentiellement impliqués dans des éléments de structure secondaire, c'est-à-dire les k plus proches acides aminés de ces résidus sur la séquence primaire des protéines. De même, la notion de distance relative entre les acides aminés d'une séquence appariés dans un brin β est un attribut récurrent. Enfin, beaucoup de ces travaux utilisent l'information des fréquences des acides aminés et des paires d'acides aminés appariés dans les éléments de structure secondaire pour apprendre des modèles prédictifs. Ces choix s'appuient en particulier sur les articles cités en début de cette section portant sur l'étude des propensions de chaque acide aminé ou paire d'acides aminés à se retrouver impliqué dans un élément de structure secondaire.

1.2 Les interactions locales : objets d'étude de la thèse

Le travail entrepris durant la thèse se place dans la problématique bioinformatique de la prédiction d'interactions locales ponctuelles qui peuvent se former dans une protéine entre deux résidus distants de celle-ci. Il existe de nombreuses interactions de ce type, en voici les principales :

- les liaisons covalentes,
- les liens hydrogènes,
- les interactions électrostatiques,
- les forces de Wan Der Waals.

La figure 1.5 montre une protéine, où les acides aminés sont représentés par des sphères, dont la structure est contrainte par quelques-unes de ces interactions. Il est admis aujourd'hui que toutes ces interactions contraignent ou stabilisent, pour les plus faibles d'entre elles, la structure. Savoir les prédire correctement serait donc une avancée importante pour le problème plus général de la prédiction de la structure 3D des protéines.

Mettre au point des méthodes de prédiction de ces interactions locales ponctuelles est d'autant plus nécessaire que la prédiction de la structure secondaire (Section 1.1.5), aujourd'hui considérée comme une tâche très avancée, ne suffit pas à déterminer complètement la structure spatiale. Connaître en plus les résidus qui sont en contact permettrait de réduire considérablement l'espace des conformations possibles d'une protéine.

Les sections suivantes présentent en détail deux de ces interactions étudiées dans les travaux présentés dans les chapitres suivants. La première est une liaison covalente (liaison forte) qui se forme entre deux cystéines d'une protéine suite à leur oxydation, elle est appelée *pont disulfure*. La seconde est une interaction beaucoup plus faible qui intervient entre deux résidus chargés proches dans l'espace, appelée *pont salin*.

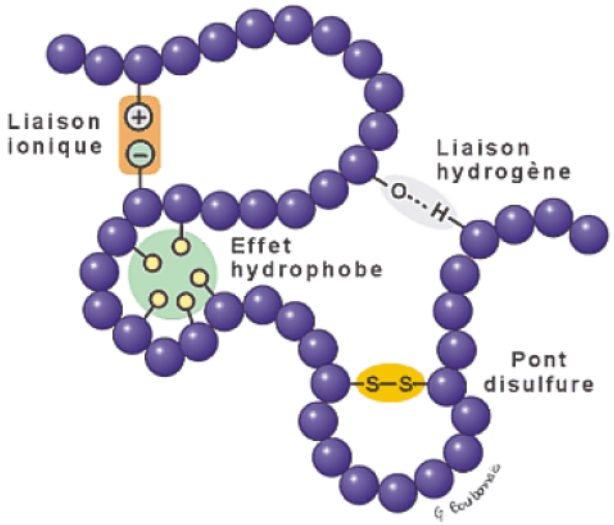


Figure 1.5. Représentation des différentes interactions locales intervenant dans la stabilisation de la structure des protéines.

1.2.1 Les ponts disulfures

Parmi toutes les interactions citées plus haut, la plus forte d'entre elles se produit entre deux *cystéines*, l'un des 20 acides aminés qui constituent l'alphabet sur lequel est formé la séquence protéique, suite à leur oxydation. Les atomes de soufre des deux cystéines ayant perdu leur atome d'hydrogène forment entre eux une liaison physique appelée covalente. Cette liaison est appelée *pont disulfure* (exemple à la figure 1.6). Une cystéine ne peut former qu'un seul pont avec une seule autre cystéine oxydée.

Ces interactions physiques sont fortes et très bien conservées, elle sont donc des contraintes importantes pour la stabilité de la structure d'une protéine. Il existe différentes méthodes expérimentales pour déterminer les ponts disulfures dans une protéine, comme la RMN et la diffraction de rayons X sur les cristaux de protéines (voir Section 1.1.3) ou encore par mutagénèse dirigée. Toutefois, ces méthodes restent coûteuses et longues. Il est donc naturel de tenter de les prédire par des méthodes bioinformatiques, tout comme la structure secondaire. La prédiction de ces ponts nécessite deux étapes :

- la prédiction des cystéines oxydées : quelles cystéines forment un pont ?
- la prédiction des ponts : quelle cystéine oxydée s'apparie avec quelle autre cystéine ?

La prédiction des cystéines oxydées d'une protéine

La première étape de la prédiction des ponts disulfures est une tâche qui a déjà été beaucoup étudiée. Les méthodes issues des travaux portant sur cette partie du problème permettent aujourd'hui une prédiction très précise des cystéines d'une protéine qui vont former un pont et de celles qui n'en formeront pas.

Les premiers travaux portant sur la prédiction de l'état d'oxydation des cystéines d'une protéine se situent autour de l'an 2000 [Fariselli et al., 1999, Fiser and Simon, 2000, Mucchielli-Giorgi et al., 2002, Frasconi et al., 2002]. Les méthodes proposées permettaient alors de prédire correctement l'état d'oxydation pour environ 80% des cystéines. Ces quatre travaux cités sont très distincts aussi bien d'un point de vue méthodologie que du point de vue de l'information utilisée pour prédire si une cystéine est oxydée ou non.

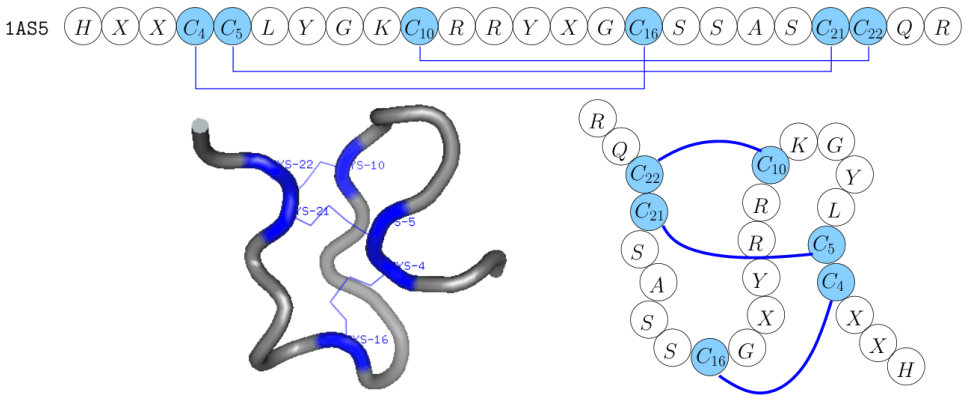


Figure 1.6. Exemple d'une protéine (identifiant PDB `1AS5`) de longueur 24 avec 6 cystéines oxydées et appariées deux à deux par un pont disulfure. (vue de gauche obtenue avec le logiciel libre PyMOL [DeLano, 2002])

Les méthodes proposées dans les travaux suivants [Martelli et al., 2002a, Martelli et al., 2002b, Ceroni et al., 2003, Song et al., 2004, Ferrè and Clote, 2005b, Baldi et al., 2005] obtiennent de bien meilleures performances. Cette amélioration très visible (entre 85 et 95% des cystéines correctement prédites) s'explique par la réunion de plusieurs facteurs : l'expérience des premiers travaux sur les attributs prédictifs pertinents, l'utilisation de méthodes plus sophistiquées comme les réseaux de neurones ou les SVM et bien sûr, le nombre de structures de protéines déterminées en constante augmentation dans les bases de données telles que la PDB.

La prédiction des appariements de cystéines oxydées

La seconde étape de la prédiction des ponts disulfures correspond à la prédiction des appariements eux-mêmes : quelle cystéine oxydée s'apparie avec quelle autre cystéine oxydée. On parle de configuration des ponts ou de registre. Contrairement à la première étape de prédiction des cystéines oxydées, ce problème reste encore un challenge pour lequel les méthodes déjà proposées ne permettent pas encore une prédiction suffisamment précise pour être intégrée à des serveurs de prédiction de la structure globale.

C'est un problème équivalent à la recherche d'un couplage parfait dans un graphe complet (Figure 1.7) où chacun des $2k$ sommets correspond à une des $2k$ cystéines oxydées d'une protéine et un arc indique un contact potentiel entre les deux cystéines portées par les sommets qu'il relie. La tâche d'apprentissage devient alors d'apprendre une fonction de pondération des arcs du graphe pour en extraire le couplage parfait de poids maximal correspondant à une configuration des ponts. C'est un problème de complexité croissante avec le nombre de ponts. Dès les premiers travaux, les auteurs décident donc de séparer ce problème d'apprentissage par le nombre de ponts contenus dans une protéine, chaque nombre de ponts formant un problème indépendant des autres.

Pour apprendre une fonction de pondération des arcs du graphe, qui relient deux cystéines d'une protéine, les auteurs des travaux sur ce problème de prédiction choisissent plusieurs informations propres à une paire de cystéines. En particulier, une information est dès le début indiquée comme essentielle : les environnements locaux des deux cystéines, c'est-à-dire les résidus voisins sur la séquences des deux cystéines.

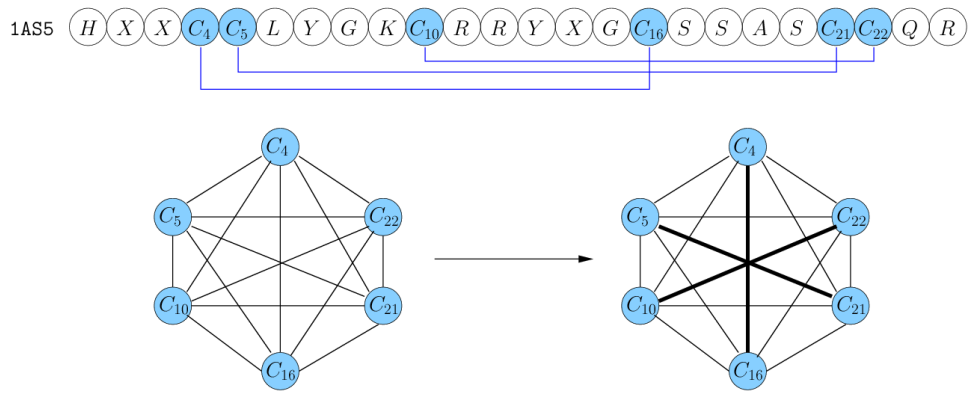


Figure 1.7. Séquence d'une protéine (Identifiant PDB `1AS5`) contenant 6 cystéines oxydées. Les trois ponts disulfures forment un couplage parfait (droite) du graphe complet (gauche).

Ce choix de l'information locale des cystéines comme principal attribut prédictif, dont un exemple est donné à la figure 1.8, s'explique par les travaux sur la prédiction de la structure secondaire (Section 1.1.5). Dans ces travaux, le contexte local des résidus impliqués dans un brin β a été démontré à plusieurs reprises comme une information pertinente pour ce problème de prédiction. C'est pour cette raison que cette information sur les cystéines est toujours utilisée pour le problème de la prédiction des ponts disulfures. Toutefois, on ne trouve aucune étude similaire à celles effectuées pour l'étude du rôle de l'environnement local des résidus impliqués dans un brin β.

D'autres informations sont également utilisées comme la distance relative entre les deux cystéines, l'information d'évolution, ou encore la structure secondaire dans laquelle se trouve une cystéine oxydée. Les fonctions ainsi apprises pondèrent chaque arc du graphe et un algorithme de couplage parfait de poids maximal sélectionne alors la configuration la plus probable. Le nombre de ponts par protéine étant fixé, le critère de performance étudié est le taux de ponts correctement prédits par une méthode d'apprentissage.

Les premiers travaux qui ont proposé cette formalisation du problème avec une méthode de prédiction sont [Fariselli and Casadio, 2001, Fariselli et al., 2002]. Les auteurs de ces travaux ont utilisé, pour attribuer un score à deux environnements locaux de cystéines, les contacts potentiels entre deux résidus proposés par Mirny et Shakhnovich et ont obtenu les meilleures performances pour un rayon d'environnement local de 5 (5 résidus de part et d'autre des cystéines). La table 1.1 donne les résultats obtenus dans ces travaux.

Nombre de ponts par protéine	Nombre de chaînes protéiques	Pourcentage de ponts correctement prédits
2	158	68
3	153	37
4	103	37
5	44	26

Table 1.1. Pourcentages de ponts disulfures correctement prédits par la méthode proposée dans les travaux [Fariselli and Casadio, 2001, Fariselli et al., 2002].

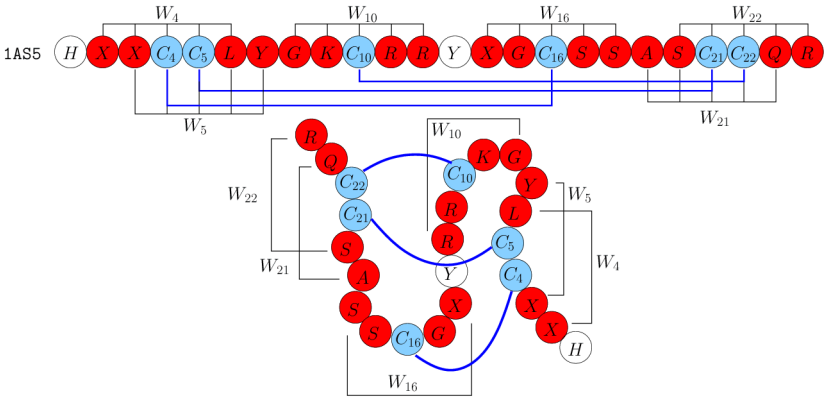


Figure 1.8. Environnements locaux des 6 cystéines oxydées de la protéine `1AS5`. Deux acides aminés de part et d'autre des cystéines sont considérés dans cet exemple.

Les deux méthodes proposées par la suite dans [Vullo and Frasconi, 2003, Vullo and Frasconi, 2004], basées sur des réseaux de neurones récursifs, ont permis d'améliorer les performances de prédiction des ponts disulfures. Le protocole expérimental était similaire à celui des travaux précédents. En particulier, le rayon optimal des environnements locaux qui ont permis d'obtenir les meilleurs résultats est encore 5. Les résultats sont indiqués dans la table 1.2.

Nombre de ponts par protéine	Nombre de chaînes protéiques	Pourcentage de ponts correctement prédits
2	156	73
3	146	56
4	99	37
5	45	30

Table 1.2. Pourcentages de ponts disulfures correctement prédits par la méthode proposée dans les travaux [Vullo and Frasconi, 2003, Vullo and Frasconi, 2004].

Ces résultats étaient, au début de cette thèse, les meilleurs sur ce problème de prédiction. On constate que dès que le nombre de ponts par protéine dépasse 2, les performances chutent de façon drastique montrant la difficulté croissante de ce problème de prédiction dès lors que le nombre de ponts augmente.

Depuis, d'autres méthodes ont été proposées pour résoudre ce problème de prédiction [Ferrè and Clote, 2005b, Zhao et al., 2005, Tsai et al., 2005, Rama et al., 2005, Baldi et al., 2005, Cheng et al., 2006]. Elles incorporent de plus en plus d'informations supplémentaires sur les paires de cystéines, profitent d'un nombre de structures déterminées en constante croissance et utilisent des méthodes toujours plus sophistiquées. Par contre, à la différence des travaux précédents, la plupart de ces méthodes tentent de traiter les deux phases de la prédiction des ponts disulfures dans le même temps, rendant difficile la comparaison avec les résultats présentés précédemment. En effet, les méthodes proposées par ces articles tentent à la fois de prédire quelles sont les cystéines oxydées, et en fonction de ces prédictions, proposent une configuration des ponts.

Toutefois, la méthode proposée dans [Tsai et al., 2005], à base de SVM, semble permettre une meilleure prédiction de la configuration des ponts pour les protéines en contenant 4 ou 5 (environ 70%). Les résultats sont donnés sur un jeu de données de redondance inférieure ou égale à 30% et dans cette étude, des profils d'environnements locaux sont utilisés à la place des environnements locaux originaux (calculés avec le logiciel d'alignements multiples PSI-Blast [Altschul et al., 1997]).

La plupart de ces groupes ont mis à disposition leurs méthodes sur des serveurs Web :

- DiANNA [Ferrè and Clote, 2005a, Ferrè and Clote, 2006],
- DISULFIND [Ceroni et al., 2006],
- DiPRO [Cheng et al., 2006],
- PreCys [Tsai et al., 2005].

La prédiction des ponts disulfures à partir de la séquence, c'est-à-dire la prédiction de l'état d'oxydation des cystéines et de leur appariement, est une tâche complexe qui reste actuellement un problème ouvert qui fait l'objet de beaucoup d'attention. De plus, des questions se posent sur ce problème. Est-ce qu'il est vraiment possible de prédire les ponts disulfures uniquement à partir de la séquence primaire de la protéine ? Une question qui peut se reformuler de la manière suivante : est-ce que les ponts ne sont pas une simple conséquence de la structure de la protéine ? Auquel cas, il faudrait connaître la structure pour prédire les ponts. Et la deuxième question qui se pose concerne l'information utilisée dans tous ces travaux : l'information contenue dans l'environnement local des cystéines. Est-ce qu'une telle information existe et contribue au fait que deux résidus soient appariés ? Si oui, dans quelle mesure ?

Tous les travaux cités dans cette section ne permettent pas cette analyse car cette information est mélangée à toutes les autres, il est donc difficile de savoir quelle est son importance dans les performances des méthodes proposées.

On peut penser à la lecture de ces travaux que les réponses à ces questions sont triviales, mais pour les biologistes et chimistes du domaine, il n'est pas évident qu'il soit possible de prédire les ponts sans connaître la structure d'une protéine et il n'est pas admis non plus que l'environnement local des cystéines joue un rôle dans ces appariements.

1.2.2 Les ponts salins

Les ponts salins sont des interactions ioniques faibles qui se forment entre deux résidus chargés d'une protéine lorsque ceux-ci se trouvent proches dans l'espace. Ils interviennent entre un résidu chargé négativement (Aspartate (lettre D) ou Glutamate (lettre E)) et un résidu chargé positivement (Arginine (R), Lysine (K) ou Histidine (H)). Ces ponts sont rarement observés dans le cœur d'une protéine (milieu hydrophobe) mais très fréquemment à la surface de celles-ci (milieu hydrophile) [Honig and Hubbell, 1984]. Ces ponts ont tendance a être accompagnés de liaisons hydrogènes, plus fortes, ce qui explique qu'on retrouve parfois ces liaisons hydrogènes dans la définition des ponts salins.

En 1984, dans les travaux présentés dans [Honig and Hubbell, 1984], Barry H. Honig et Wayne Hubbell mettent en avant des phénomènes intéressants sur ces interactions concernant leur rôle potentiel dans la stabilité des protéines. Les travaux de ces auteurs montrent que les ponts salins peuvent exister dans des milieux très hostiles à leur formation, ce qui suggère que, même très faibles, ces liaisons peuvent contribuer à la stabilité de la structure des protéines (ou groupements de protéines).

Dans les années qui ont suivi, plusieurs travaux se sont concentrés sur l'étude du rôle de ces ponts dans la stabilité de la structure. Dans [Hendsch and Tidor, 1994], les auteurs mènent des expériences qui montrent que ces ponts devraient plus être considérés comme des liaisons qui contribueraient à l'unicité, à la spécificité d'une protéine, que comme des liaisons stabilisantes. On retrouve également ce constat dans les travaux présentés dans l'article [Xu et al., 1997]. Les auteurs y indiquent que les ponts salins seraient en partie l'explication du cœur stable de la protéine et d'une surface malléable, capable de s'adapter à différents ligands (molécules qui se fixent à la protéine) grâce aux liaisons comme les ponts salins ou hydrogènes, qui peuvent se réagencer facilement.

D'autres travaux ont également étudié le rôle des ponts salins dans la stabilité de la structure [Kumar and Nussinov, 1999, Vijayakumar and Zhou, 2001, Kumar and Nussinov, 2001, Bayas et al., 2007]. Dans [Vijayakumar and Zhou, 2001], les auteurs étudient les trois ponts salins présents dans une enzyme, appelée la Barnase. Il montrent, en remplaçant à chaque expérience un résidu chargé par un résidu neutre (le pont salin ne peut plus se former), que les protéines mutantes obtenues sont beaucoup moins stables que la protéine originale, et donc que les ponts salins stabilisent la protéine.

La stabilité apportée par un pont salin à la structure 3D d'une protéine est aujourd'hui admise, tout comme il est admis que la position du pont dans la protéine influe sur sa capacité à stabiliser celle-ci [Kumar and Nussinov, 2001].

Ces ponts présentent de fortes similarités avec les ponts disulfures et en particulier du point de vue de la formalisation que l'on peut en faire : ce sont des interactions ponctuelles entre deux résidus distants d'une protéine.

Aucune méthode pour prédire ces ponts n'a été proposée actuellement. Les ponts disulfures, plus stables et plus conservés que les ponts salins, ont été privilégiés jusqu'à aujourd'hui. Bien que ces interactions soient plus faibles, elles restent des contacts intéressants à prédire puisqu'ils sont impliqués dans la structure et dans sa stabilité. Ces ponts sont concernés par l'étude menée dans cette thèse et des expérimentations sur des protéines contenant des ponts salins sont proposées.

1.3 L'A.C.I. masse de données Genoto3D

Cette thèse s'est déroulée dans le cadre d'un des nombreux projets de recherche visant à apporter une contribution à ce défi de la prédiction de la structure : le projet **Genoto3D** (*www.loria.fr/∼guermeur/ACIMD/*) de l'Action Concertée Incitative (ACI) « Masses de données ». Ce projet a démarré en avril 2003 sous la responsabilité de Yann Guermeur (*Yann.Guermeur@loria.fr*) et s'est achevé en novembre 2006.

1.3.1 Description du projet

Le projet rassemblait des équipes impliquées dans la bioinformatique structurale des protéines et dans l'apprentissage automatique de plusieurs laboratoires :

- LORIA (*Vandoeuvre-lès-Nancy*),
- IBCP (*Lyon*),
- INRA (*Jouy-en-Josas*)
- IRISA (*Rennes*),
- LIF (*Marseille*),
- LIRMM (*Montpellier*).

L'objet de ce projet, dans le contexte de la prédiction de la structure 3D des protéines, était de mettre en évidence des problèmes génériques difficiles et de proposer des méthodes qui s'appuient sur des approches provenant de l'apprentissage susceptibles de faire progresser l'état de l'art dans le domaine. La stratégie adoptée est de « diviser pour régner », c'est-à-dire d'aborder la tâche progressivement, au travers de problèmes connexes tels que la prédiction de la structure secondaire ou de liaisons comme les ponts disulfures ou salins.

Les retombées de projets tels que Genoto3D sont importantes, aussi bien en biologie qu'en apprentissage automatique. En effet, chaque information prédite de manière précise sur la structure des protéines apporte une contribution non négligeable au problème global et l'apprentissage automatique est également enrichi par ces projets qui permettent de confronter des méthodes algorithmiques et des savoir-faire différents sur un problème difficile, d'apporter une meilleure compréhension des techniques d'apprentissage, la conception de nouveaux algorithmes adaptés aux données séquentielles ou l'adaptation d'algorithmes développés dans d'autres contextes.

1.3.2 Les jeux de données développés dans le cadre du projet

Pour les besoins du groupe de travail de l'A.C.I., un ensemble de protéines annotées a été extrait de la base de données de structures de protéines PDB par Christophe Geourjon [Errami et al., 2003], un des membres du groupe de travail. Cet ensemble contient 1688 protéines dont le taux d'homologie entre les séquences de ces protéines est inférieur à 25%. A partir de cet ensemble de protéines annotées, deux jeux de données ont été préparés : le premier indique pour chacune de ces 1688 protéines les ponts disulfures observés expérimentalement dans ces protéines et le second les ponts salins. Nous nommons respectivement ces jeux de données `G3D-SS` et `G3D-SA`.

La figure 1.9 montre un extrait du fichier `G3D-SS` pour une protéine contenant trois ponts disulfures. L'identifiant de la protéine, la séquence primaire, la structure secondaire et la description des ponts disulfures présents dans la protéine constituent les informations disponibles sur chacune des 1688 protéines.

L'annexe A de ce mémoire (page 147) propose une description et des statistiques sur ces deux ensembles de données utilisés durant la thèse.

```
>/disk2/Pdb/Entries/pdb1e0f.ent Chaine=k
IRFGMGKVPCPDGEVGYTCDCGEKICLYGQSCNDGQCSGDPKPSSEFEEFEIDEEEK
*
CEECCCCCCCCCCEEECCCEECCCCEECCCEEECCCCEEECCCCCCCCCCCCCCTTCCC
*
TRUE-INTRA : CYS K 10 - CYS K 19 Dist = 2.026
TRUE-INTRA : CYS K 21 - CYS K 32 Dist = 2.031
TRUE-INTRA : CYS K 26 - CYS K 37 Dist = 2.026
```

Figure 1.9. Extrait du jeu de données `G3D-SS`. La chaîne K de la protéine d'identifiant PDB `1e0f` contient 3 ponts. Le premier lie la cystéine à la position 10 de la séquence primaire et celle à la position 19.

Chapitre 2

Préliminaires sur l'apprentissage et la classification supervisée

Sommaire

Ce chapitre présente les notions de base sur la classification supervisée nécessaires aux travaux de recherche décrits dans ce mémoire de thèse. Ce domaine de l'apprentissage statistique se concentre autour d'une problématique qui peut se définir de manière très informelle comme la construction d'une procédure permettant de classer automatiquement des données à partir d'un ensemble d'exemples déjà classés.

Parmi les principaux intérêts de ce domaine, on trouve la création d'outils automatisés de prise de décision, d'aide au diagnostic ou de prévision. Par exemple, il s'agira d'établir un diagnostic médical à partir de la description clinique du patient, de décider de l'accord d'un prêt bancaire en fonction de la situation du client, de reconnaître des formes sur une photo (lettres ou chiffres manuscrits, visages, etc), de filtrer des spams, de détecter des pannes à partir de signaux, ou encore de prédire certaines parties de la structure d'une protéine à partir de sa séquence primaire.

L'apprentissage statistique et la classification supervisée sont des domaines très vastes que ces préliminaires n'ont aucunement prétention à couvrir. Des présentations détaillées de ces deux domaines peuvent être trouvées dans [Mitchell, 1997], [Vapnik, 1998], [Hastie et al., 2001], et [Cornuéjols et al., 2002].

Les notions exposées dans les différentes sections de ce chapitre sont destinées à donner une description formelle du problème de la classification supervisée ainsi que des résultats sur lesquels les travaux de recherche menés durant la thèse se sont appuyés. C'est également au travers de ces préliminaires que le vocabulaire et les notations adoptés pour la rédaction de ce mémoire sont donnés.

Les sections sont organisées de la façon suivante :

– La section 2.1 propose une vue d'ensemble de la problématique de la classification supervisée, un des domaines de l'apprentissage à partir d'exemples. Elle décrit le problème considéré : apprendre à classer à partir de données déjà classées, et présente les principaux concepts attenants à cette problématique. Plusieurs variantes de la classification supervisée sont considérées dans les travaux présentés dans ce mémoire, cette section les présente également. Enfin, la dernière partie propose une description du cadre spécifique d'apprentissage PAC (*Probably Approximatively Correct*) dans lequel une étude est menée dans cette thèse.

– La section 2.2 présente le classifieur naïf de Bayes et son calcul dans différents contextes d'apprentissage. Ce classifieur est basé sur une hypothèse simplificatrice concernant les distributions sous-jacentes au problème considéré, cette dernière est dite naïve du fait qu'elle n'est que très rarement vérifiée en pratique. Toutefois, ce classifieur a été largement étudié dans la littérature du domaine et il est reconnu pour sa simplicité et ses bonnes performances sur certains problèmes comme la classification de textes. Cette section décrit les résultats concernant ce classifieur qui sont utilisés dans le cadre de l'étude menée dans cette thèse.

– La section 2.3 présente des travaux sur l'apprentissage de séparateurs linéaires avec l'algorithme du perceptron. Différents contextes d'apprentissage sont considérés et les variantes de l'algorithme correspondant sont décrites. Les séparateurs linéaires sont souvent présents dans la littérature du domaine car leur extension naturelle aux séparateurs non linéaires permise par les machines à noyaux est au cœur de beaucoup de travaux de forte actualité.

2.1 Notions de base sur la classification supervisée

Cette section propose une description générale de l'apprentissage à partir d'exemples et de la problématique de la classification supervisée. Elle offre un aperçu du domaine dans lequel la majeure partie des travaux menés durant la thèse s'inscrivent.

2.1.1 Apprentissage à partir d'exemples

L'apprentissage automatique est un vaste domaine hétéroclite. En effet, considéré comme un domaine de l'informatique, il est également en étroite interaction avec les domaines des mathématiques, des sciences cognitives, des neurosciences et de la biologie. Il concerne de nombreuses applications et peut être considéré plus comme une problématique qu'un problème isolé à résoudre. C'est un domaine de forte actualité car il est impliqué dans de nombreuses applications et sciences. Il est délicat de vouloir donner une définition formelle de l'apprentissage, en voici une qui en décrit le principe : l'apprentissage automatique est une discipline qui consiste à développer des méthodes qui permettent de faire évoluer une machine (au sens large) de façon à ce qu'elle accomplisse des tâches qu'il n'est pas possible de réaliser par des méthodes plus classiques.

L'apprentissage à partir d'exemples considère la situation où l'on dispose de données que l'on cherche à classer de façon automatique dans un ensemble de classes $\mathcal{Y}$. Classer des données porte deux noms différents selon le type des éléments de $\mathcal{Y}$:

- lorsque les éléments de $\mathcal{Y}$ sont *quantitatifs* (taille, poids, mesure, quantité, etc), on parle d'un problème de *régression*,

- lorsque ces éléments sont *qualitatifs* (malade, sain, positif, négatif, bleu, blanc, rouge, grand, petit, etc), on parle d'un problème de *classification*.

Les travaux présentés dans ce mémoire ne considèrent que la deuxième catégorie de problèmes : ceux qui consistent à classer des données dans un ensemble de classes qualitatives. Là encore, on distingue les deux situations où l'ensemble des classes $\mathcal{Y}$ est :

- inconnu : on parle de problèmes de *classification non supervisée*,

- connu : on parle de problèmes de *classification supervisée*.

Dans le premier cas, on attend d'une méthode de proposer un ensemble structuré de classes dans lesquelles les données dont on dispose se répartissent de façon homogène. Dans le deuxième, on connaît les classes de $\mathcal{Y}$ et on cherche à construire une procédure qui classe correctement les données à partir d'exemples déjà classés dans $\mathcal{Y}$.

Seul le deuxième ensemble de problèmes est considéré dans ce mémoire. On appellera donc *classification supervisée* la problématique qui consiste à construire une méthode capable de classer correctement des exemples dans un ensemble $\mathcal{Y}$ connu et qualitatif à partir d'un ensemble d'exemples déjà classés dans $\mathcal{Y}$. Une description formelle de cette problématique est donnée dans la section suivante. Toute méthode répondant à cette définition sera appelée *méthode de classification supervisée*.

2.1.2 Classification supervisée

Cette section donne une description formelle d'un problème de classification supervisée : comment se présente un tel problème ? quels sont les objectifs que l'on cherche à atteindre ? comment peut-on les atteindre ? et quels sont les moyens d'évaluer la qualité d'une solution à un tel problème ?

Comment se présente un problème de classification supervisée ?

Au départ de tout problème de classification supervisée, on dispose d'un ensemble de données (exemples) déjà classées – *étiquetées* – qui sera noté S_{lab}, qui se présente sous la forme $S_{lab} = \{(x_1, y_1), ..., (x_l, y_l)\}$ et tel que, $\forall i \in \{1, ..., l\}$, x_i appartient à un *espace de description* $\mathcal{X}$: on l'appelle la *description* de l'exemple, et y_i est la classe de l'exemple, elle appartient à un ensemble $\mathcal{Y}$. On suppose que les données de S_{lab} sont distribuées indépendamment et identiquement selon une distribution P sur $\mathcal{X} \times \mathcal{Y}$ fixe mais inconnue.

Voici quelques exemples de tels ensembles :

- $\mathcal{X}$ = symptômes d'un patient, $\mathcal{Y}$ = maladies diagnostiquées
- $\mathcal{X}$ = séquences protéiques, $\mathcal{Y}$ = éléments de structure secondaire
- $\mathcal{X}$ = séquences protéiques, $\mathcal{Y}$ = familles de protéines
- $\mathcal{X}$ = article scientifique, $\mathcal{Y}$ = domaines scientifiques des articles

L'espace des descriptions $\mathcal{X}$ est un domaine défini comme le produit cartésien de m attributs : $\mathcal{X} = \prod_{j=1...m} \mathcal{X}^j$. Chacun de ces attributs peut être binaire, numérique ou symbolique. Nous considérons dans cette partie préliminaire que les attributs qui décrivent les données sont symboliques afin de simplifier les écritures. Pour toute donnée $x \in \mathcal{X}$, on notera x^j la projection de x sur $\mathcal{X}^j$ et on appellera x^j la valeur de l'attribut j de x. A partir de la distribution P sur $\mathcal{X} \times \mathcal{Y}$, on définit la distribution P sur $\mathcal{X}$ qui décrit la distribution des exemples dans $\mathcal{X}$: $\forall x \in \mathcal{X}$, $P(x) = \sum_{y \in \mathcal{Y}} P(x, y)$.

L'*ensemble d'apprentissage* S_{lab} est étiqueté, c'est-à-dire déjà classé par un spécialiste du domaine ou par un algorithme de classification non supervisée. On suppose que la classe y observée de tout exemple dépend de façon statistique de l'observation x. C'est un cadre non déterministe, où chaque exemple peut potentiellement être classé dans une

Figure 2.1. Les ensembles $\mathcal{X}$ et $\mathcal{Y}$ munis respectivement d'une loi de probabilité $P(.)$ et de lois de probabilités conditionnelles $P(.|x)$ $\forall x \in \mathcal{X}$ tel que $P(x) \neq 0$.

classe ou une autre avec une certaine probabilité. En effet, $\forall x \in \mathcal{X}$ tel que $P(x) \neq 0$, on définit la loi de distribution conditionnelle $P(.|x)$ sur $\mathcal{Y}$ à partir de la distribution P sur $\mathcal{X} \times \mathcal{Y}$ et de la distribution P sur $\mathcal{X}$: $\forall y \in \mathcal{Y}$, $P(y|x) = \frac{P(x,y)}{P(x)}$. Le cas déterministe, où un exemple ne peut être classé que d'une seule façon – avec probabilité 1 – est donc un cas particulier du cadre décrit ici.

L'ensemble des distributions $P(.)$ sur $\mathcal{X}$ et $P(.|x)$ sur $\mathcal{Y}$, $\forall x \in \mathcal{X}$ tel que $P(x) \neq 0$, forme un *modèle* du problème considéré, elles le définissent entièrement. Ce modèle est a priori inconnu, mais on dispose des exemples de S_{lab}, i.i.d. selon la distribution jointe $P(x, y) = P(x) \cdot P(y|x)$, comme information partielle sur ce modèle.

Objectif de la classification supervisée

L'objectif d'un problème de classification supervisée est de calculer, à partir de l'ensemble de données étiquetées S_{lab}, une fonction $f : \mathcal{X} \to \mathcal{Y}$, appelée *classifieur* ou *modèle prédictif* (Figure 2.2), qui approche au mieux la relation qu'il y a entre les descriptions et les classes. Cette fonction doit être capable, pour toute donnée x de l'espace $\mathcal{X}$, de déterminer la classe $y \in \mathcal{Y}$ qui lui correspond le mieux. On dit que f doit posséder de bonnes capacités de *généralisation* puisqu'elle doit apprendre à classer au mieux des données qu'elle n'a, par exemple, jamais rencontrées dans la phase d'apprentissage et suivant une relation dont elle n'a qu'une description partielle par S_{lab}.

Les méthodes qui permettent de calculer un classifieur f à partir d'un échantillon de données étiquetées S_{lab} sont appelées *apprenants* ou *algorithmes d'apprentissage*.

Erreur de classification et règle de Bayes

Dans le paragraphe précédent, il est indiqué que le classifieur que l'on cherche à calculer doit attribuer à un exemple la classe qui lui correspond le mieux. Une notion essentielle en classification supervisée est l'*évaluation* des performances d'un classifieur. Cette évaluation nécessite le choix d'un critère. Très souvent, on cherche un classifieur qui fasse tout simplement le moins d'erreurs, mais on peut également vouloir maximiser une fonction de profit ou de gain ou encore pénaliser certaines erreurs plus que d'autres.

On utilise pour cela une *fonction de perte*, notée $Loss(f(x), y)$, qui attribue un coût à la décision $f(x)$ d'un classifieur f pour un exemple donné x sachant que l'on attend y comme réponse. Nous ne considérons dans ce mémoire que la *fonction de perte 0-1* classique,

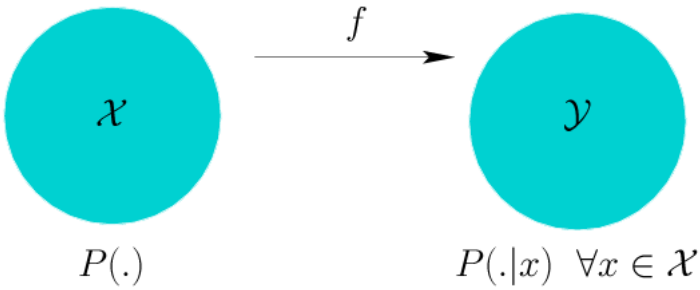


Figure 2.2. Un classifieur f définit une relation entre les descriptions et les classes.

c'est-à-dire que $Loss(f(x), y)$ vaut 1 si $f(x) \neq y$, 0 sinon. Lorsque $\mathcal{X}$ est un espace discret, on définit l'*erreur de classification* $R_f(x)$ de f sur un exemple x tiré selon une distribution P sur $\mathcal{X}$ et l'*erreur réelle de classification* $R(f)$ ou *risque de* f de la façon suivante :

Définition 2.1. *Soient $f : \mathcal{X} \to \mathcal{Y}$ un classifieur et x_0 un exemple de $\mathcal{X}$ tiré selon une distribution P sur $\mathcal{X}$. L'erreur $R_f(x_0)$ de f pour la description x_0 est la probabilité que x_0 soit mal classé par f : $R_f(x_0) = P(x, y|x = x_0, y \neq f(x_0))$.*

Définition 2.2. *L'erreur réelle $R(f)$ d'un classifieur f est la moyenne pondérée des erreurs de f sur toutes les descriptions $x \in \mathcal{X}$: $R(f) = \sum_{x \in \mathcal{X}} R_f(x) \cdot P(x)$.*

Cette erreur n'est pas calculable dans la plupart des problèmes du fait du manque de connaissance sur la distribution originale P. Diverses méthodes sont néanmoins utilisées pour l'estimer. Un deuxième ensemble de données déjà classées, tiré de façon indépendante de l'ensemble d'apprentissage S_{lab} et noté S_{test}, permet d'estimer le risque $R(f)$ de f en calculant l'*erreur empirique* ou *erreur apparente* $\hat{R}(f)$ de f sur S_{test} :

Définition 2.3. *L'erreur empirique $\hat{R}(f)$ de f sur un ensemble de données classées $S_{test} = \{(x_1, y_1), ..., (x_l, y_l)\}$ est le taux d'erreur de f sur S_{test} : $\hat{R}(f) = \frac{1}{l} \cdot \sum_{x_i \in S_{test}} Loss(f(x_i), y_i)$ où $Loss(f(x_i), y_i) = 1$ si $f(x_i) \neq y_i$ et 0 sinon (fonction de perte 0-1).*

La règle de décision optimale pour assigner une classe à un objet $x \in \mathcal{X}$ est appelée *règle de Bayes*, elle consiste à attribuer à tout objet $x \in \mathcal{X}$ la classe $y \in \mathcal{Y}$ la plus probable sachant x. Le classifieur C_{Bayes} induit par cette règle est appelé *classifieur de Bayes* :

$$C_{Bayes}(x) = \underset{y}{argmax}\ P(y|x) \quad (x \in \mathcal{X}, y \in \mathcal{Y}) \tag{2.1}$$

Néanmoins, ce classifieur optimal requiert la connaissance complète des distributions sous-jacentes. Comme indiqué dans les paragraphes précédents, ces distributions ne sont pas connues et les données dont on dispose ne suffisent généralement pas à les estimer. C'est la raison pour laquelle ce classifieur ne peut, en règle générale, être calculé sans information complémentaire sur les distributions sur lesquelles il s'appuie. Il faut remarquer que l'erreur réelle minimale que permet d'obtenir ce classifieur ne vaut 0 que dans un cadre déterministe où chaque exemple ne peut être classé que d'une seule façon, avec probabilité 1.

Choisir une catégorie de classifieurs

De nombreuses catégories de classifieurs existent, ainsi que d'algorithmes pour les apprendre. Par exemple, les *arbres de décision* (Figure 2.3(a)) sont des ensembles de règles structurées faciles à interpréter et à appliquer. L'algorithme ID3 [Quinlan, 1979] amélioré en C4.5 dans [Quinlan, 1993], ou encore l'algorithme CART [Breiman et al., 1984], sont des algorithmes pour construire ces classifieurs à partir de données dont la classe est connue. De même, les *séparateurs linéaires* (Figure 2.3(b)) sont des classifieurs ; l'algorithme du perceptron [Rosenblatt, 1958] permet de les calculer à partir d'exemples étiquetés. Bien d'autres catégories de classifieurs et algorithmes existent, et sont proposés encore régulièrement de nos jours, prouvant le caractère très actif du domaine de la classification.

Le choix de la catégorie de classifieurs dans laquelle on travaille pour résoudre un problème de classification est très important, ainsi que le choix de l'apprenant chargé de calculer le classifieur. Cela demande une bonne connaissance de la nature du problème (déterministe ou non, complexité, etc) et dépend également du choix de la description des données (par exemple, le type des attributs).

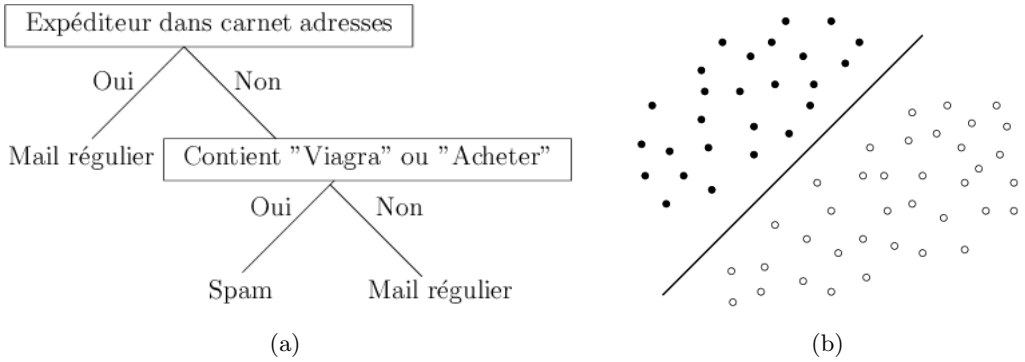


Figure 2.3. Exemples de classifieurs : (a) un arbre de décision, (b) un séparateur linéaire. Dans les deux cas, l'espace de description $\mathcal{X}$ est de dimension 2.

Synthèse

La figure 2.4 propose un schéma de synthèse de ce qui est exposé dans cette section sur le problème de la classification supervisée. L'ensemble d'apprentissage S_{lab} sert d'entrée à un algorithme d'apprentissage (l'apprenant) qui calcule à partir de ces exemples un classifieur f dont on peut estimer le risque $R(f)$ par un ensemble de données S_{test} indépendant de S_{lab}. Ce classifieur sert à classer de façon automatique de nouvelles données dont on ne connaît pas la classe.

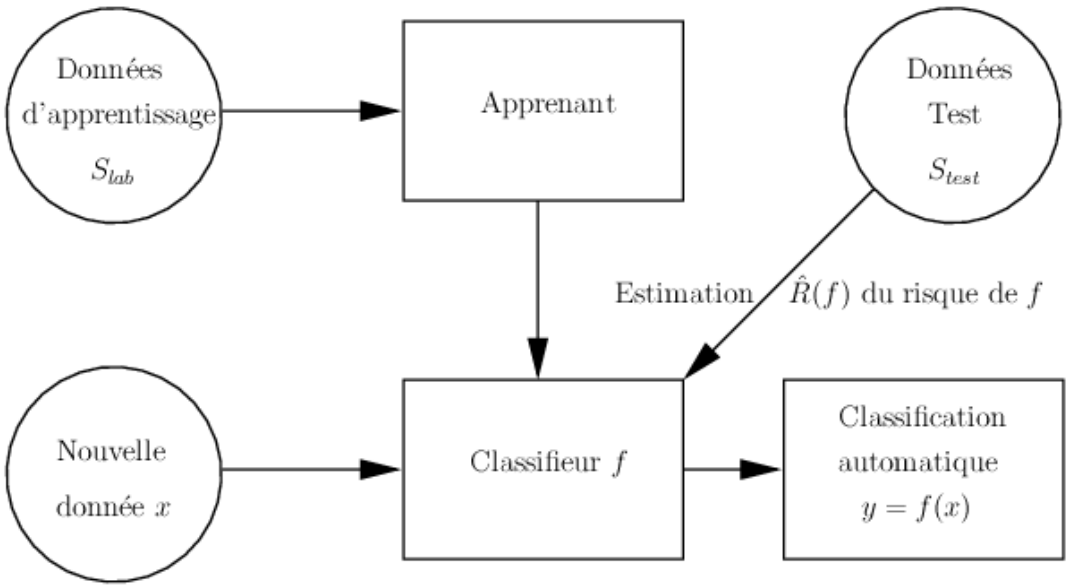


Figure 2.4. Schéma de synthèse sur la classification supervisée.

Notations spécifiques à la classification supervisée binaire

Lorsque le nombre de classes de $\mathcal{Y}$ vaut 2, on parle de classification supervisée binaire. Le problème biologique étudié dans cette thèse a entraîné des travaux de recherche qui considèrent exclusivement ce cas. Le vocabulaire spécifique utilisé est présenté ici.

Les deux classes sont appelées respectivement *classe positive* et *classe négative*. On utilise alors indifféremment les notations $\mathcal{Y} = \{+, -\}$ ou $\mathcal{Y} = \{1, -1\}$ selon le contexte. Les exemples (x, y) tels que $y = +$ (respectivement $y = -$) sont appelés des *exemples positifs* (resp. *exemples négatifs*). Ces notations, propres au cas binaire, ne présentant aucune ambiguïté, différents abus de notation sont fréquemment utilisés dans ce mémoire pour simplifier certaines parties techniques. Par exemple, la notation « $y =$ » est omise dans les expressions où elle intervient : $P(y = +)$ devient $P(+)$, $P(y = +|x)$ devient $P(+|x)$. De même, les distributions $P(.|y = +)$ et $P(.|y = -)$ sur $\mathcal{X}$ sont régulièrement notées $P_+(.)$ et $P_-(.)$. Ces notations sont toutefois rappelées dans certaines sections.

2.1.3 Variantes de la classification supervisée

Plusieurs variantes de la classification supervisée sont considérées dans ce mémoire. Cette section des préliminaires les présente brièvement.

Classification en contexte semi-supervisé

Classer des données manuellement ou expérimentalement peut être une tâche longue, complexe, coûteuse, voire même parfois impossible. Par ailleurs, avec peu de données déjà classées, il est difficile d'obtenir un classifieur ayant toute la capacité de généralisation qu'on en attend. Pourtant, il arrive souvent que beaucoup de données non étiquetées soient disponibles comme par exemple des documents textuels, des pages html ou encore des séquences protéiques. Ces données apportent une information sur la distribution des exemples qui doit pouvoir être utilisée lors de la phase d'apprentissage.

On appelle *classification semi-supervisée* la problématique qui consiste à utiliser des données non étiquetées, en plus de données classées, pour le calcul d'un classifieur tel que défini dans la section précédente. L'objectif est toujours d'apprendre un classifieur f qui minimise le risque $R(f)$ et pour cela, on dispose donc de deux ensembles de données d'apprentissage : l'ensemble S_{lab} tel que défini à la section 2.1.2 et un ensemble de données non étiquetées $S_{unl} = \{x'_1, ..., x'_{l'}\}$ supposées i.i.d. selon la distribution $P(.)$ sur $\mathcal{X}$. Généralement, la taille de S_{unl} est beaucoup plus grande que celle de S_{lab} ($l' >> l$).

On trouve différentes sortes de travaux sur ce contexte d'apprentissage. Certains proposent des méthodes ou adaptations de méthodes supervisées pour apprendre un classifieur à partir de ces deux sources de données [Nigam et al., 1998, Blum and Mitchell, 1998, Corduneanu and Jaakkola, 2001, Chapelle et al., 2003, Zhu et al., 2003a]. D'autres références en fournissent une étude plus théorique [Cozman et al., 2003] ou plus expérimentale [Nigam et al., 2000, Moreno and Agarwal, 2003]. Généralement, les algorithmes proposés tentent d'inférer des modèles qui maximisent la vraisemblance (Section 2.2.2) des deux ensembles de données grâce à des méthodes de type *Expectation Maximization* (E.M.) [Dempster et al., 1977]. Une description de cette méthode est proposée à la section 2.2.3. Les travaux cités ici constatent qu'utiliser l'ensemble S_{unl} en plus de S_{lab} pour apprendre un classifieur améliore le plus souvent [Cozman et al., 2003] les performances de celui-ci.

Cas particulier asymétrique

Le thème de l'*apprentissage semi-supervisé asymétrique* ou *apprentissage à partir de données positives et non étiquetées* a été introduit dans [Denis, 1998]. C'est une variante de l'apprentissage semi-supervisé binaire dans laquelle l'ensemble des données classées dont on dispose est uniquement constitué d'exemples positifs, distribués selon la loi $P(.|+) = \frac{P(+|.)\cdot P(.)}{P(+)}$ sur $\mathcal{X}$. On note cet ensemble S_{pos}.

Plusieurs méthodes sont proposées pour apprendre des classifieurs malgré l'absence de données négatives. Certaines de ces méthodes permettent de construire des arbres de décision [De Comité et al., 1999b, Letouzey et al., 2000], ou encore de calculer les paramètres des classifieurs naïfs de Bayes (défini à la section 2.2) [Denis et al., 2002a, Liu et al., 2002, Liu et al., 2003, Denis et al., 2003, Li and Liu, 2005] dans ce contexte. Une série de travaux propose également une méthode algorithmique itérative qui permet d'utiliser les machines à vecteurs supports (SVM) [Yu et al., 2002, Yu et al., 2003, Liu et al., 2003, Li and Liu, 2003, Yu et al., 2004, Fung and Lu, 2006] dans ce cadre d'apprentissage. La plupart de ces méthodes sont basées sur des hypothèses supplémentaires (paramètres connus du système d'apprentissage) ou sur des heuristiques liées au problème considéré : la classification de textes pour la plupart des travaux.

Classification supervisée avec bruit de classification

Il arrive pour certains problèmes de classification supervisée que les données observées soient corrompues par du *bruit de classification*. On appelle bruit de classification tout processus qui, pour une raison ou une autre, modifie la classe des exemples. Dans ce cas, les étiquettes observées des exemples de S_{lab} ne sont plus fiables. L'objectif reste toutefois de construire un classifieur f qui minimise le risque $R(f)$ pour la distribution originale du problème. Pour pouvoir atteindre cet objectif, on est alors contraint de faire l'hypothèse que le processus qui corrompt les classes des données observées possède une structure particulière qu'on appelle un *modèle de bruit*.

Plusieurs modèles de bruit ont été proposés et étudiés dans la littérature. Lorsque $\mathcal{Y} = \{+, -\}$, le modèle de bruit CN [Angluin and Laird, 1988], appelé *bruit de classification uniforme*, suppose que tout exemple classé (x, y) voit sa classe échangée avec une certaine probabilité η constante et inférieure à 0.5. La généralisation de ce bruit, noté CPCN [Decatur, 1997] et appelé *bruit de classification constant par morceaux*, suppose qu'une partition de $\mathcal{X} \times \mathcal{Y}$ est telle que sur chaque morceau de la partition, le bruit est uniforme (CN) avec un taux de bruit η qui dépend de la partition. Ces deux modèles de bruit ont été particulièrement étudiés dans le cadre d'apprentissage PAC (*Probably Approximatively Correct*) de Valiant [Valiant, 1984] présenté en section 2.1.4.

Le bruit de classification est un processus qui rend inefficace la plupart des algorithmes standards de classification supervisée. C'est la raison pour laquelle des adaptations de méthodes qui ont fait leurs preuves dans le cas supervisé sans bruit de classification sont proposées dans la littérature [Bylander, 1994, Blum et al., 1996, Zhu et al., 2003b, Yang et al., 2003]. Diverses stratégies sont adoptées : tenter de détecter les exemples bruités des données d'apprentissage (stratégie généralement peu efficace), prendre directement en compte le modèle de bruit dans le processus d'apprentissage, identifier la distribution originale à partir de la distribution corrompue lorsque cela est possible, etc.

2.1.4 Apprentissage PAC de Valiant

Cette section présente le cadre d'apprentissage PAC (Probably Approximatively Correct) introduit par L.G. Valiant en 1984 dans [Valiant, 1984]. Ce cadre d'apprentissage considère des problèmes de classification supervisée déterministes binaires (nous utilisons la notation $\mathcal{Y} = \{1, -1\}$). Ce cadre d'apprentissage est généralement décrit par des notations qui lui sont spécifiques. La première partie de cette section en donne la signification.

Soient $\mathcal{X}$ un espace quelconque de description, et P une distribution fixe mais inconnue appartenant à l'ensemble $\mathcal{P}$ de toutes les distributions sur $\mathcal{X}$. On appelle *concept* un sous-ensemble de $\mathcal{X}$. Avec les mêmes notations que celles utilisées précédemment, on dira qu'un élément $x \in \mathcal{X}$ est positif si x appartient au concept c, et qu'il est négatif sinon. De plus, on appelle *classe de concepts*, noté $\mathcal{C}$, un ensemble de concepts. La figure 2.5 fournit une illustration de la notion de concept.

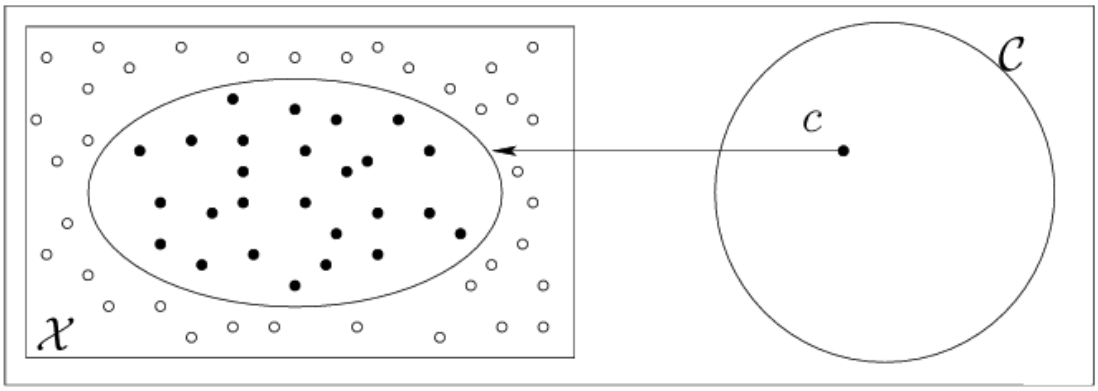


Figure 2.5. Illustration de la notion de concept dans le cadre PAC.

Avec ces notations, la tâche d'apprentissage peut donc se décrire comme l'identification de c à partir d'exemples pour lesquels on sait s'ils appartiennent à c ou non. On modélise la présence des exemples classés en supposant l'existence d'un *oracle* $EX(c, P)$ tel qu'à chaque appel à cet oracle, celui-ci fournit une paire $(x, c(x))$ telle que $x \in \mathcal{X}$ est tiré selon la distribution P et $c(x) = 1$ si $x \in c$ et $c(x) = -1$ sinon. La notation de la classe $c(x)$ d'un exemple x remplace donc la notation y utilisée dans les sections précédentes.

Le cadre d'apprentissage PAC proposé par Valiant fournit un cadre théorique qui permet de caractériser les classes de concepts apprenables. De façon informelle, une classe de concepts $\mathcal{C}$ sera dite PAC-apprenable si $\forall c \in \mathcal{C}$ et $\forall P \in \mathcal{P}$, on peut calculer à partir de $EX(c, P)$ une bonne approximation de c avec une forte confiance, d'où le terme d'apprentissage *probablement approximativement correct*. En voici la définition formelle :

Définition 2.4. *Une classe de concepts $\mathcal{C}$ est dite efficacement PAC-apprenable s'il existe un algorithme $\mathcal{A}$ tel que $\forall c \in \mathcal{C}$, $\forall P$ sur $\mathcal{X}$, $\forall \epsilon, \delta > 0$, $\mathcal{A}$ calcule en un nombre polynomial $p(\frac{1}{\epsilon}, \frac{1}{\delta})$ d'appels à $EX(c, P)$ et en temps polynomial $q(\frac{1}{\epsilon}, \frac{1}{\delta})$ une hypothèse h dans un espace de représentation $\mathcal{H}$ sur $\mathcal{X}$ telle que $P(R(h) > \epsilon) < \delta$ avec $R(h) = P(h(x) \neq c(x))$. ϵ est appelé la précision, δ le paramètre de confiance et $R(h)$ le risque ou l'erreur de h.*

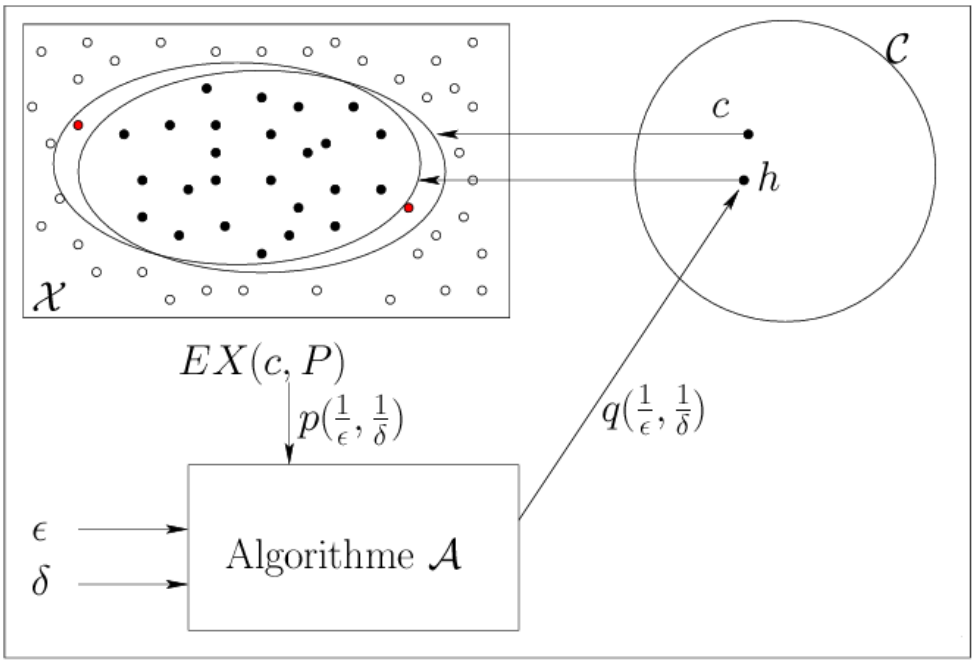


Figure 2.6. Illustration d'une classe de concepts efficacement PAC-apprenable avec $\mathcal{H} = \mathcal{C}$.

Les deux polynômes p et q ont également un argument supplémentaire : la taille du concept c à apprendre [Valiant, 1984]. Ce paramètre n'entre toutefois pas en jeu dans les travaux présentés dans ce mémoire, il n'apparaîtra donc pas dans les notations afin de les simplifier. Ces deux polynômes permettent de contrôler que l'apprentissage d'un concept c de $\mathcal{C}$ peut se faire en temps raisonnable et à partir d'un nombre d'exemples pas trop élevé.

L'espace de représentation $\mathcal{H}$ signifie que la recherche d'une bonne approximation de c peut, pour une raison ou une autre, ne pas être effectuée dans l'ensemble de concepts $\mathcal{C}$. La précision ϵ et la confiance $1 - \delta$ contrôlent la qualité de l'hypothèse h.

La définition de la PAC-apprenabilité d'une classe de concepts connaît plusieurs extensions. Par exemple, les travaux présentés dans [Balcan and Blum, 2005] proposent l'extension du cadre PAC pour l'apprentissage semi-supervisé. Les deux sections suivantes en présentent l'extension lorsque les exemples fournis par l'oracle sont corrompus par du bruit de classification uniforme (CN) et constant par morceaux (CPCN).

PAC-apprenabilité avec bruit de classification uniforme CN

Le cadre de l'apprentissage PAC de classes de concepts avec bruit de classification uniforme CN a été introduit dans [Angluin and Laird, 1988]. L'oracle EX^{η}_{CN} auquel l'algorithme d'apprentissage a accès est alors défini par :

Définition 2.5. *Soient $\eta \in [0,1]$, $c \in \mathcal{C}$ et $P \in \mathcal{P}$. L'oracle $EX^{\eta}_{CN}(c,P)$ fournit à chaque appel une paire $(x, c^{\eta}(x))$ telle que x est tiré selon P et $c^{\eta}(x)$ est défini par :*

$$c^{\eta}(x) = \begin{cases} c(x) \text{ avec probabilité } 1-\eta \\ -c(x) \text{ avec probabilité } \eta \end{cases}$$

La notion de CN-apprenabilité est alors définie de la façon suivante :

Définition 2.6. *Une classe de concepts $\mathcal{C}$ est efficacement CN-apprenable par une classe de représentation $\mathcal{H}$ ssi il existe un algorithme $\mathcal{A}$ et deux polynômes p et q tels que $\forall c \in \mathcal{C}$, $\forall P \in \mathcal{P}$, $\forall \epsilon, \delta > 0$ et $\forall \eta \in [0, 0.5[$, en ayant accès à $EX^{\eta}_{CN}(c, P)$, ϵ, δ et à une borne η_0<0.5 de η, $\mathcal{A}$ calcule avec une probabilité $> 1 - \delta$ une hypothèse $h \in \mathcal{H}$ telle que $R(h) \leq \epsilon$. $\mathcal{A}$ requiert au plus $p(\frac{1}{\epsilon}, \frac{1}{\delta}, \frac{1}{1-2\eta_0})$ appels à $EX^{\eta}_{CN}(c, P)$ et s'arrête en temps $q(\frac{1}{\epsilon}, \frac{1}{\delta}, \frac{1}{1-2\eta_0})$.*

L'hypothèse de la connaissance d'une borne η_0 du taux de bruit η n'est pas une condition restrictive car, comme indiqué dans [Angluin and Laird, 1988] ou encore dans [Kearns and Vazirani, 1994], il est possible de calculer une valeur de η_0 lorsqu'aucune n'est connue. La figure 2.7 illustre, comme pour le cas sans bruit, cette définition.

PAC-apprenabilité avec bruit de classification constant par morceau CPCN

L'extension de l'apprentissage PAC au bruit de classification constant par morceau a été proposée dans [Decatur, 1997]. On considère dans ce contexte un ensemble de fonctions de partitions $\Pi = \{\pi_1, ..., \pi_k\}$ définies sur $\mathcal{X} \times \mathcal{Y}$ et à valeurs dans $\{0, 1\}$ telles que $\sum_{i=1}^{k} \pi_i(x, y) = 1$ pour toute paire $(x, y) \in \mathcal{X} \times \mathcal{Y}$. On suppose alors que sur chaque morceau π_i de la partition Π, le taux de bruit η_i est constant (Figure 2.8). L'oracle $EX^{\overrightarrow{\eta}}_{CPCN}$ est alors défini de la façon suivante :

Définition 2.7. *Soient $\Pi = \{\pi_1, ..., \pi_k\}$ un ensemble de fonctions de partitions sur $\mathcal{X} \times \mathcal{Y}$ et $\overrightarrow{\eta} = [\eta_1, ..., \eta_k]$ avec $\eta_i \in [0, 1]$. Soient $c \in \mathcal{C}$ et $P \in \mathcal{P}$, l'oracle $EX^{\overrightarrow{\eta}}_{CPCN}(c, P)$ fournit à chaque appel une paire $(x, c^{\overrightarrow{\eta}}(x))$ telle que x est tiré selon P et $c^{\overrightarrow{\eta}}(x)$ est défini par :*

$$c^{\overrightarrow{\eta}}(x) = \begin{cases} c(x) \text{ avec probabilité } 1 - \eta_i \\ -c(x) \text{ avec probabilité } \eta_i \\ \text{avec } i \text{ tel que } \pi_i(x, c(x)) = 1 \end{cases}$$

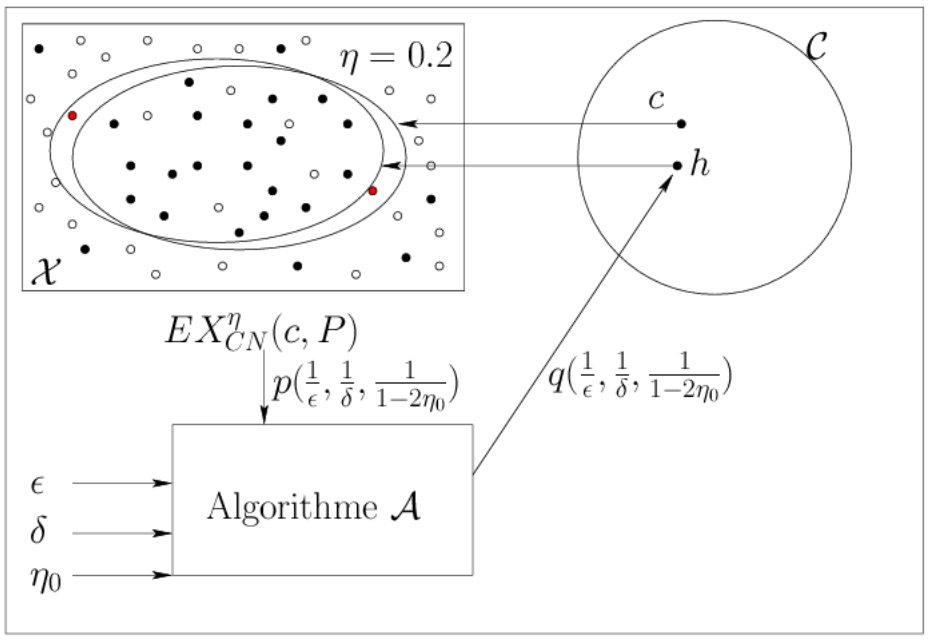


Figure 2.7. Illustration d'une classe de concepts efficacement CN-apprenable avec $\mathcal{H} = \mathcal{C}$.

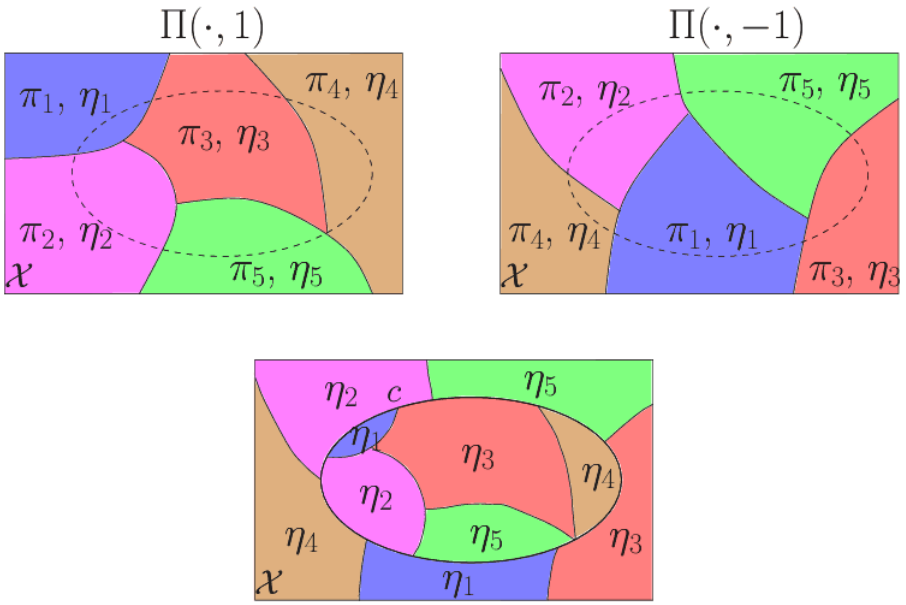


Figure 2.8. Illustration d'une partition Π sur $\mathcal{X} \times \{1, -1\}$ (images du haut) puis de la partition résultante correspondant à un concept c donné (image du bas).

La définition suivante donne alors les conditions de la CPCN-apprenabilité :

Définition 2.8. *Une classe de concepts $\mathcal{C}$ est efficacement CPCN-apprenable par une classe de représentation $\mathcal{H}$ ssi il existe un algorithme $\mathcal{A}$ et deux polynômes p et q tels que $\forall c \in \mathcal{C}$, $\forall P \in \mathcal{P}$, $\forall\ \Pi = \{\pi_1, ..., \pi_k\}$, $\forall\ \overrightarrow{\eta} = [\eta_1, ..., \eta_k]$ avec $\eta_i \in [0, 0.5[$, $\forall \epsilon, \delta > 0$, en ayant accès à $EX^{\overrightarrow{\eta}}_{CPCN}(c, P)$, ϵ, δ et à une borne $\eta_0 < 0.5$ des η_i, $\mathcal{A}$ calcule avec une probabilité supérieure à $1 - \delta$ une hypothèse $h \in \mathcal{H}$ telle que $R(h) \leq \epsilon$. $\mathcal{A}$ requiert au plus $p(\frac{1}{\epsilon}, \frac{1}{\delta}, \frac{1}{1-2\eta_0})$ appels à $EX^{\overrightarrow{\eta}}_{CPCN}(c, P)$ et s'arrête en temps $q(\frac{1}{\epsilon}, \frac{1}{\delta}, \frac{1}{1-2\eta_0})$.*

Nous avons montré (Annexe C) que, sous certaines conditions sur les taux de bruit qui affectent les données, les classes de concepts apprenables avec bruit de classification CPCN sont les mêmes que celles apprenables avec bruit de classification CN.

Beaucoup de travaux proposent des études de l'apprentissage dans le cadre PAC avec la présence de bruit de classification [Angluin and Laird, 1988, Kearns, 1993, Kearns and Vazirani, 1994, Bylander, 1994, Aslam and Decatur, 1994, Blum et al., 1996, Decatur, 1997, Bylander, 1998, Dunagan and Vempala, 2004, Goldberg, 2006]. Les résultats de certains de ces travaux sont décrits dans ce mémoire. En particulier ceux qui concernent l'apprentissage PAC des séparateurs linéaires avec bruit de classification [Bylander, 1994, Blum et al., 1996, Bylander, 1998, Dunagan and Vempala, 2004] sont donnés en section 2.3. Cette classe de concepts est très étudiée en grande partie parce que son extension naturelle aux séparateurs non linéaires avec les machines à noyaux est connue pour être particulièrement performante sur beaucoup de problèmes réels.

2.2 Distributions produits et classifieur naïf de Bayes

Cette section des préliminaires présente le *classifieur naïf de Bayes* et les algorithmes permettant de le calculer dans différents contextes d'apprentissage.

Un des principaux intérêts de ce classifieur est qu'il est calculable simplement et efficacement à partir d'un ensemble de données. Il a fait l'objet de beaucoup d'attention en apprentissage automatique. Intensivement utilisé, par exemple, pour des problèmes de classification de textes [Lewis and Ringuette, 1994, Mc Callum and Nigam, 1998, Nigam and Ghani, 2000, Eyheramendy et al., 2003], il a régulièrement montré des performances surprenantes sur la tâche de classification au vu de l'hypothèse forte sur laquelle il s'appuie et qui ne se vérifie pourtant généralement pas en pratique. Plusieurs études de ce classifieur [Langley et al., 1992, Domingos and Pazzani, 1996, Domingos and Pazzani, 1997, Rish, 2001] justifient pourquoi et à quelles conditions ce modèle prédictif très simple conserve de très bonnes performances en classification alors que l'hypothèse sur laquelle il est construit n'est généralement pas vérifiée.

2.2.1 L'hypothèse simplificatrice du classifieur naïf de Bayes

Soit $\mathcal{X} = \prod_{j=1}^{m} \mathcal{X}^j$ un espace discret de description défini par m attributs symboliques et soit $\mathcal{Y}$ un ensemble fini de classes. La règle de Bayes présentée en section 2.1.2 est la règle de décision optimale pour attribuer une classe $y \in \mathcal{Y}$ à tout objet $x \in \mathcal{X}$. Le classifieur associé, appelé *classifieur de Bayes* s'écrit :

$$C_{Bayes}(x) = \underset{y}{argmax}\ P(y|x) = \underset{y}{argmax}\ P(x|y) \cdot P(y) \quad (x \in \mathcal{X}, y \in \mathcal{Y}) \tag{2.2}$$

L'application de cette règle nécessite une connaissance complète du problème considéré, c'est-à-dire de toutes les distributions sous-jacentes à celui-ci. Or l'espace des descriptions $\mathcal{X}$ est souvent de taille très importante et l'ensemble de données étiquetées dont on dispose ne permet pas, dans la plupart des cas réels, d'estimer tous les paramètres du modèle.

En revanche, si on fait l'hypothèse que les attributs descriptifs de $\mathcal{X}$ sont indépendants deux à deux conditionnellement à chacune des classes, on fait alors l'hypothèse que les distributions $P(.|y)$ sur $\mathcal{X}$ pour tout $y \in \mathcal{Y}$ sont des *distributions produits* et le problème se simplifie car $\forall x = (x^1, ..., x^m)$ de $\mathcal{X}$ $(x^j \in \mathcal{X}^j)$, les probabilités $P(x|y)$ s'expriment simplement comme le produit des probabilités $P(x^j|y) : P(x|y) = \prod_{j=1}^{m} P(x^j|y)$.

Dans ce cas, le nombre de paramètres à estimer devient raisonnable. Lorsque les classes et les attributs sont binaires, le nombre de paramètres à estimer passe de $O(2^m)$ à $O(m)$. Que cette hypothèse sur les attributs soit vérifiée ou non, le classifieur de Bayes défini en (2.2) devient alors le *classifieur naïf de Bayes*, noté C_{NB}, et il est défini par :

$$C_{NB}(x) = \underset{y}{argmax}\ P(y) \cdot \prod_{j=1}^{m} P(x^j|y) \quad (x \in \mathcal{X}, y \in \mathcal{Y}) \tag{2.3}$$

L'hypothèse d'indépendance n'est pas vérifiée dans la plupart des problèmes réels, ce qui explique l'utilisation du terme naïf. Malgré cela, ce classifieur est connu pour donner de bons résultats en classification [Domingos and Pazzani, 1996].

Lorsque $\mathcal{Y} = \{+, -\}$, les classifieurs naïfs de Bayes sont entièrement spécifiés par l'ensemble des paramètres suivants :

- $p = P(y = +)$,
- $p_{jk} = P(x^j = k|y = +)\ \forall j \in \{1, ..., m\}$ et $\forall k \in \mathcal{X}^j$,
- $q_{jk} = P(x^j = k|y = -)\ \forall j \in \{1, ..., m\}$ et $\forall k \in \mathcal{X}^j$.

Une instance de l'ensemble de ces paramètres est appelée un *modèle*, noté θ.

2.2.2 Estimateurs analytiques des paramètres du classifieur

Lorsqu'on cherche à estimer les paramètres d'un modèle à partir d'un ensemble de données, on attend de ce modèle qu'il rende compte au mieux des données. Le principe du maximum de vraisemblance définit un critère pour choisir un tel modèle.

Estimation de paramètres par le principe du maximum de vraisemblance

Soient $S = \{z_1, ..., z_l\}$ un ensemble de données i.i.d. et θ un modèle, on appelle *vraisemblance* (respectivement *log-vraisemblance*) de S pour le modèle θ et on note $L(\theta, S)$ (resp. $l(\theta, S)$), les valeurs :

$$L(\theta, S) = \prod_{i=1}^{l} P(z_i|\theta) \quad \text{et} \quad l(\theta, S) = log\ L(\theta, S) \tag{2.4}$$

Le *principe du maximum de vraisemblance* recommande de trouver un modèle θ tel que $L(\theta, S)$ – et donc aussi $l(\theta, S)$ – soit maximale.

Application à l'estimation du classifieur naïf de Bayes en contexte supervisé

Le principe du maximum de vraisemblance permet d'obtenir en contexte supervisé des estimateurs analytiques des paramètres qui spécifient les classifieurs naïfs de Bayes.

Soit $S_{lab} = \{(x_1, y_1), ..., (x_l, y_l)\}$ l'ensemble de données d'apprentissage étiquetées. La vraisemblance de S_{lab} pour un modèle donné θ s'écrit :

$$\begin{aligned} L(\theta, S_{lab}) &= \prod_{i=1}^{l} P(x_i, y_i|\theta) \\ &= \prod_{i=1}^{l} P(y_i|\theta) \Big[\prod_{j=1}^{m} P(x_i^j|y_i, \theta) \Big] \end{aligned} \tag{2.5}$$

Lorsque $\mathcal{Y} = \{+, -\}$ et en notant n_p le nombre de données positives de S_{lab} (soit $l - n_p$ données négatives), n_{jk} (respectivement m_{jk}) le nombre de données positives (resp. négatives) de S_{lab} telles que $x^j = k$, la vraisemblance de S_{lab} pour un modèle donné $\theta = \{p, p_{jk}, q_{jk}, j \in \{1, ..., m\}, k \in \mathcal{X}^j\}$ définie en (2.5) s'écrit :

$$L(\theta, S_{lab}) = p^{n_p} \cdot (1 - p)^{l - n_p} \cdot \prod_{\substack{j \in \{1, ..., m\} \\ k \in \mathcal{X}^j}} p_{jk}^{n_{jk}} \cdot q_{jk}^{m_{jk}} \tag{2.6}$$

De même, la log-vraisemblance de S_{lab} pour le modèle θ s'écrit :

$$l(\theta, S_{lab}) = n_p \cdot log\ p + (l - n_p) \cdot log\ (1-p) + \sum_{\substack{j \in \{1,...,m\} \\ k \in \mathcal{X}^j}} n_{jk}\ log\ p_{jk} + m_{jk}\ log\ q_{jk} \quad (2.7)$$

Cette fonction trouve son maximum pour le modèle $\hat{\theta}$ suivant :

$$\hat{p} = \frac{n_p}{l} \quad (2.8)$$

$$\hat{p}_{jk} = \frac{n_{jk}}{\sum\limits_{s \in \mathcal{X}^j} n_{js}} = \frac{n_{jk}}{n_p} \quad (2.9)$$

$$\hat{q}_{jk} = \frac{m_{jk}}{\sum\limits_{s \in \mathcal{X}^j} m_{js}} = \frac{m_{jk}}{l - n_p} \quad (2.10)$$

Les paramètres du modèle $\hat{\theta}$ sont des estimateurs analytiques consistants des paramètres du modèle θ qui spécifient le classifieur naïf de Bayes recherché. On appelle *algorithme naïf de Bayes* (Algorithme 1) et on note `NB` la méthode ainsi définie pour estimer les paramètres des classifieurs naïfs de Bayes.

En pratique, les estimations des paramètres p_{jk} et q_{jk} nécessitent toutefois de prendre des précautions. En effet, si pour une classe $y \in \mathcal{Y}$, une valeur k d'un attribut j n'est jamais observée sur l'ensemble des données de classe y de S_{lab}, alors l'estimation de la probabilité $P(x^j = k|y)$ vaut 0 et cela entraîne l'exclusion de la classe y pour tout exemple $x \in \mathcal{X}$ tel que $x^j = k$. Pour éviter cette situation, on applique par exemple un *lissage* des paramètres estimés. Nous considérons le *lissage de Laplace*, les formules (2.9) et (2.10) deviennent alors, en notant $|\mathcal{X}^j|$ le nombre de valeurs que peut prendre l'attribut j :

$$\hat{p}_{jk} = \frac{1 + n_{jk}}{|\mathcal{X}^j| + n_p} \quad (2.11)$$

$$\hat{q}_{jk} = \frac{1 + m_{jk}}{|\mathcal{X}^j| + l - n_p} \quad (2.12)$$

Ce lissage permet de ne pas rencontrer le problème décrit ci-dessus sans trop biaiser les estimations des paramètres. Tous les algorithmes qui tentent d'estimer les paramètres des classifieurs naïfs de Bayes contenus dans ce mémoire sont concernés par ce problème. Nous considérons donc, comme c'est le cas ici, que ce lissage est appliqué par défaut pour tous ces algorithmes : il n'apparaîtra donc pas dans leur description (par l'exemple, l'algorithme `NB` donné ici indique les estimateurs (2.9) et (2.10) au lieu de (2.11) et (2.12)).

Algorithme 1 `NB` - Algorithme naïf de Bayes supervisé pour $\mathcal{Y} = \{+, -\}$

Entrée: $S_{lab} = \{(x_1, y_1), ..., (x_l, y_l)\}$ i.i.d. selon $P(x, y) = P(x) \cdot P(y|x)$

Calculer n_p, n_{jk} et m_{jk} $\forall j \in \{1, ..., m\}$, $\forall k \in \mathcal{X}^j$ sur S_{lab}
Calculer $\hat{p}$, $\hat{p}_{jk}$ et $\hat{q}_{jk}$ $\forall j \in \{1, ..., m\}$, $\forall k \in \mathcal{X}^j$ avec les formules (2.8), (2.9) et (2.10)

Sortie: $\hat{\theta} = \{\hat{p}, \hat{p}_{jk}, \hat{q}_{jk}\}$ qui maximise la vraisemblance de S_{lab}

Cas semi-supervisé

Dans un contexte d'apprentissage semi-supervisé (Section 2.1.3), on dispose de deux ensembles de données : l'ensemble de données classées $S_{lab} = \{(x_1, y_1), ..., (x_l, y_l)\}$ et l'ensemble de données non classées $S_{unl} = \{x'_1, ..., x'_{l'}\}$. On peut modéliser la présence de ces deux échantillons par l'existence d'un *oracle* qui, avec une probabilité β fournit un exemple étiqueté et avec la probabilité $1 - \beta$ procure un exemple non étiqueté. Le paramètre β complète le modèle θ vu précédemment : $\theta' = \theta \cup \{\beta\}$.

Lorsque $\mathcal{Y} = \{+, -\}$, les probabilités d'avoir un exemple $(x, y) \in S_{lab}$ ou un exemple $x \in S_{unl}$ dans ce nouveau modèle θ' se calculent de la manière suivante :

$$\begin{aligned} P(x, y|\theta') &= \beta \cdot P(x, y|\theta) && (2.13)\\ P(x|\theta') &= (1 - \beta) \cdot P(x|\theta) && (2.14)\\ \text{avec } P(x|\theta) &= P(+|\theta) \cdot P(x, y|y = +, \theta) + P(-|\theta) \cdot P(x, y|y = -, \theta) && (2.15) \end{aligned}$$

La vraisemblance de S_{lab} et S_{unl} pour le modèle θ' s'écrit alors :

$$\begin{aligned} L(\theta', S_{lab}, S_{unl}) &= \prod_{i=1}^{l} \beta \cdot P(x_i, y_i|\theta) \cdot \prod_{i=1}^{l'} (1 - \beta) \cdot P(x'_i|\theta) && (2.16)\\ &= \beta^l \cdot L(\theta, S_{lab}) \cdot (1 - \beta)^{l'} \cdot L(\theta, S_{unl}) && (2.17) \end{aligned}$$

avec $L(\theta, S_{lab})$ telle que définie en (2.6) et, avec les mêmes notations :

$$L(\theta, S_{unl}) = \prod_{i=1}^{l'} \Big(p \cdot \prod_{\substack{j \in \{1,...,m\} \\ k/x'^j_i = k}} p_{jk} + (1 - p) \cdot \prod_{\substack{j \in \{1,...,m\} \\ k/x'^j_i = k}} q_{jk} \Big) \qquad (2.18)$$

La valeur de β qui maximise la vraisemblance est $\beta = \frac{l}{l+l'}$, soit la proportion d'exemples classés dans l'ensemble d'apprentissage. En revanche, l'expression de la vraisemblance obtenue ne permet pas de trouver des formules analytiques pour le calcul des paramètres p, p_{jk} et q_{jk} de façon à ce qu'ils maximisent la vraisemblance comme dans le cas supervisé.

2.2.3 Maximum local de la vraisemblance dans le cas semi-supervisé

Lorsque les paramètres du modèle qui maximise la vraisemblance des données d'apprentissage ne peuvent pas être obtenus de façon analytique, il est possible d'utiliser des méthodes qui permettent d'obtenir un maximum local de cette vraisemblance comme par exemple la méthode E.M. (*Expectation Maximization*). Cette section présente la méthode ainsi que son application au calcul du classifieur naïf de Bayes en contexte semi-supervisé.

La méthode E.M. (*Expectation Maximization*)

La méthode E.M. a été proposée dans [Dempster et al., 1977] pour l'inférence de modèles de mélange de densités. Considérons par exemple le modèle de mélange $P(.) = p \cdot P(.|+)+(1-p) \cdot P(.|-)$ sur $\mathcal{X}$ puisque c'est celui concerné ici (avec $\mathcal{Y} = \{+, -\}$). Si les observations x sont accompagnées de la distribution $P(.|+)$ ou $P(.|-)$ dont elles proviennent, on a vu dans la section précédente que l'inférence des paramètres de P est

Algorithme 2 `EM` - Description de la méthode E.M.

Entrée: S_{obs}, les données observées

1 - Choisir un modèle initial θ_0
2 - Calculer $Q(\theta_n, \theta_n)$ pour le n courant (phase d'estimation)
3 - Trouver θ_{n+1} tel que $Q(\theta_{n+1}, \theta_n) > Q(\theta_n, \theta_n)$ (phase de maximisation)
4 - Itérer à l'étape 2 jusqu'à convergence

Sortie: le modèle θ_c obtenu après convergence

simple. Lorsque ce n'est pas le cas, la méthode E.M. tente itérativement d'inférer à la fois les paramètres de P et la distribution qui a généré x de manière à maximiser la vraisemblance. La description de la méthode E.M. qui suit est extraite de [Hastie et al., 2001].

Soient θ' un modèle, S_{obs} l'ensemble des données observées, S_{mis} les données manquantes et S l'ensemble des données complètes d'un problème : $S = S_{obs} \cup S_{mis}$. On note :

- $l(\theta', S)$ la log-vraisemblance de S dans le modèle θ',
- $l(\theta', S_{mis}|S_{obs})$, la log-vraisemblance de S_{mis} dans le modèle θ' sachant S_{obs},
- $l(\theta', S_{obs})$ la log-vraisemblance de S_{obs} dans le modèle θ'.

Avec ces notations, on a $l(\theta', S_{obs}) + l(\theta', S_{mis}|S_{obs}) = l(\theta', S)$, soit :

$$l(\theta', S_{obs}) = l(\theta', S) - l(\theta', S_{mis}|S_{obs}) \tag{2.19}$$

En supposant que les données sont générées selon un modèle θ fixé et que S_{obs} est observé, les termes de l'égalité précédente sont des variables aléatoires dépendantes de S_{mis}. On peut donc calculer leur espérance (notée E) :

$$E(l(\theta', S_{obs})|S_{obs}, \theta) = E(l(\theta', S)|S_{obs}, \theta) - E(l(\theta', S_{mis})|S_{obs}, \theta) \tag{2.20}$$

Soit, en posant $Q(\theta', \theta) = E(l(\theta', S)|S_{obs}, \theta)$ et $R(\theta', \theta) = E(l(\theta', S_{mis})|S_{obs}, \theta)$ et en remarquant que $E(l(\theta', S_{obs})|S_{obs}, \theta) = l(\theta', S_{obs})$:

$$l(\theta', S_{obs}) = Q(\theta', \theta) - R(\theta', \theta) \tag{2.21}$$

Le principe du maximum de vraisemblance recommande de chercher un modèle θ' qui maximise $l(\theta', S_{obs})$. La méthode E.M. est une heuristique, basée sur le résultat suivant qui énonce que maximiser Q ne peut pas faire décroître la vraisemblance.

Théorème 2.1. *Si $Q(\theta_2, \theta_1) > Q(\theta_1, \theta_1)$, $l(\theta_2, S_{obs}) > l(\theta_1, S_{obs})$ [Dempster et al., 1977]*

La méthode E.M. est donc basée sur l'heuristique de trouver un modèle qui maximise Q. La méthode proposée pour cela est décrite par l'algorithme 2. C'est une méthode itérative qui assure de converger vers un maximum local de la vraisemblance. En effet, elle est sensible aux conditions initiales (le modèle θ_0). Dans [Dempster et al., 1977], les auteurs proposent de répéter cet algorithme avec différents modèles initiaux θ_0 puis de choisir parmi les modèles obtenus θ_c celui qui maximise $l(\theta_c, S_{obs})$.

Application au classifieur naïf de Bayes en contexte semi-supervisé

Dans [Nigam et al., 1998, Nigam et al., 2000], les auteurs utilisent la méthode E.M. pour estimer les paramètres d'un classifieur naïf de Bayes en contexte semi-supervisé. La méthode proposée dans leurs travaux, et décrite ici, permet donc d'obtenir un modèle qui maximise localement la vraisemblance.

Le contexte semi-supervisé suppose que $S_{obs} = \{S_{lab}, S_{unl}\}$ avec, comme définis précédemment, $S_{lab} = \{(x_1, y_1), ..., (x_l, y_l)\}$ et $S_{unl} = \{x'_1, ..., x'_{l'}\}$, et que les données manquantes de S_{mis} sont les étiquettes des données de S_{unl}.

Les auteurs de [Nigam et al., 1998] proposent de choisir comme modèle initial θ_0, le modèle calculé sur l'ensemble de données étiquetées S_{lab} avec l'algorithme `NB` (Algorithme 1). Ils proposent ensuite, étant donné un modèle θ_n courant, de calculer les paramètres du modèle θ_{n+1} de la façon suivante (avec les notations définies en section 2.2.2 pour les paramètres du modèle quand $\mathcal{Y} = \{+, -\}$, ainsi que pour les caractéristiques de S_{lab}) :

$$p = \frac{n_p + \sum_{i=1}^{l'} \hat{P}(y'_i = +|x'_i, \theta_n)}{l + l'} \tag{2.22}$$

$$p_{jk} = \frac{n_{jk} + \sum_{x'_i \in S_{unl}/x'^{j}_i = k} \hat{P}(y'_i = +|x'_i, \theta_n)}{n_p + \sum_{i=1}^{l'} \hat{P}(y'_i = +|x'_i, \theta_n)} \tag{2.23}$$

$$q_{jk} = \frac{m_{jk} + \sum_{x'_i \in S_{unl}/x'^{j}_i = k} \hat{P}(y'_i = -|x'_i, \theta_n)}{l - n_p + \sum_{i=1}^{l'} \hat{P}(y'_i = -|x'_i, \theta_n)} \tag{2.24}$$

où les $\hat{P}$ sont calculées sur les données de S_{unl} en fonction du modèle courant θ_n.

Cela revient à considérer que, pour un modèle θ_n donné (obtenu à l'itération correspondante), chaque exemple x'_i de S_{unl} est classé avec le modèle θ_n pour une fraction $\hat{P}(y'_i = +|x'_i, \theta_n)$ (resp. $\hat{P}(y'_i = -|x'_i, \theta_n)$) comme exemple positif (resp. négatif) et que le modèle θ_{n+1} est calculé en prenant en compte ces étiquettes de la même manière qu'en contexte supervisé. Et c'est justement le fait que toutes les données de S_{unl} soient temporairement classées par le modèle θ_n qui justifie que le modèle θ_{n+1} appris sur ces nouvelles données (celles de S_{lab} et celles de S_{unl} avec leurs étiquettes temporaires) soit nécessairement tel que $Q(\theta_{n+1}, \theta_n) \geq Q(\theta_n, \theta_n)$. Nous notons `NBSS-EM` l'algorithme correspondant (Algorithme 3).

Les expériences sur un problème de classification de textes menées par les auteurs de l'algorithme `NBSS-EM` [Nigam et al., 1998] montrent une amélioration sensible des performances du classifieur naïf de Bayes lorsqu'il est calculé avec des données non étiquetées en plus des données étiquetées. Ces résultats mettent en avant l'apport pertinent d'information des données non étiquetées dans un problème de classification.

Algorithme 3 `NBSS-EM` - Algorithme naïf de Bayes semi-supervisé pour $\mathcal{Y} = \{+, -\}$
Entrée: S_{lab}, S_{unl}

1 - $\theta_0 = \texttt{NB}(S_{lab})$
2 - Calculer $\hat{P}(y' = +|x', \theta_n)$ et $\hat{P}(y' = -|x', \theta_n)$ $\forall x' \in S_{unl}$ avec le modèle θ_n courant
3 - Calculer θ_{n+1} avec les formules (2.22), (2.23) et (2.24)
4 - Itérer à l'étape 2 jusqu'à convergence

Sortie: le modèle θ_c obtenu après convergence

2.2.4 Calcul du classifieur dans d'autres contextes d'apprentissage

Cette section présente d'autres travaux sur le calcul du classifieur naïf de Bayes pour des contextes d'apprentissage différents de ceux présentés dans les sections précédentes. Il en existe d'autres, mais seuls ceux qui sont utilisés ou cités à plusieurs reprises dans certaines parties de ce mémoire sont décrits ici.

Contexte d'apprentissage semi-supervisé asymétrique

Ce cas particulier de l'apprentissage semi-supervisé suppose, comme indiqué à la section 2.1.3, que l'ensemble de données classées S_{lab} dont on dispose n'est constitué que d'exemples positifs, il est noté S_{pos}. Ce cadre d'apprentissage est d'ailleurs fréquemment rencontré sous le nom d'*apprentissage à partir de données positives et non étiquetées*. Plusieurs travaux se sont penchés sur la question du calcul des paramètres d'un classifieur naïf de Bayes dans ce cas d'apprentissage. La possibilité de calculer un modèle θ à partir de S_{pos} et S_{unl} revient à se poser la question de la possibilité d'obtenir une bonne estimation du paramètre $p{=}P(y = +)$ à partir de ces données (voir Section 3.2.1). Ce problème a été évoqué en premier lieu dans [De Comité et al., 1999a, De Comité et al., 1999b]. Une des solutions proposées est de supposer que cette estimation est connue du système d'apprentissage [Denis et al., 2002b, Denis et al., 2003].

Dans [Liu et al., 2002], deux autres solutions sont proposées. La première, appelée `I-EM`, consiste à adapter l'algorithme `NBSS-EM` présenté en section 2.2.3 en utilisant comme modèle initial le modèle appris par l'algorithme `NB` sur les données positives de S_{pos} et les données non classées de S_{unl} en considérant que ces dernières sont négatives. Les performances de l'algorithme `I-EM` sont très limitées du fait de la qualité très basse du modèle initial choisi. Aussi, les auteurs proposent une deuxième méthode, nommée `S-EM`, qui consiste à infiltrer dans S_{unl} des « espions » positifs (des exemples de S_{pos}) qui permettent, après l'obtention d'un modèle avec l'algorithme `I-EM`, de fixer des règles de sélection d'exemples fortement négatifs dans S_{unl} afin de constituer un ensemble S_{neg} d'exemples négatifs. Cet ensemble obtenu, l'algorithme `NBSS-EM` peut alors être utilisé.

Plusieurs variantes de `S-EM` existent dans la littérature, toutes basées sur ce principe d'extraire de S_{unl} un ensemble S_{neg} d'exemples négatifs pour revenir à un cadre d'apprentissage semi-supervisé classique. Toutefois, les méthodes proposées dans tous ces travaux sont basées sur des heuristiques. Nous avons montré (Chapitre 3) que tous les paramètres du classifieur naïf de Bayes sont en réalité déterminés par les données sans hypothèse supplémentaire ni heuristique.

Identification des paramètres du classifieur à partir de la distribution $P(.)$

Lorsque $\mathcal{Y} = \{+,-\}$, la distribution $P(.)$ sur $\mathcal{X}$ est un mélange des deux distributions $P(.|+)$ et $P(.|-)$. En effet, $P(.) = p \cdot P(.|+) + (1-p) \cdot P(.|-)$. Une condition nécessaire et suffisante de l'identifiabilité de mélanges finis de distributions est donnée dans [Yakowitz and Spragins, 1968]. L'identifiabilité des *2-mélanges* peut se définir ainsi :

Définition 2.9. *Soit $\mathcal{P}$ un ensemble de distributions sur $\mathcal{X}$. Les 2-mélanges d'éléments de $\mathcal{P}$ sont identifiables si $\forall P_1, P_2, P'_1, P'_2 \in \mathcal{P}$ et $\forall \alpha, \alpha' \in [0,1]$, l'implication suivante est vérifiée :* $\alpha P_1 + (1-\alpha)P_2 = \alpha' P'_1 + (1-\alpha')P'_2 \Rightarrow \alpha = \alpha', P_1 = P'_1, P_2 = P'_2$ *ou* $\alpha' = 1-\alpha, P'_1 = P_2, P'_2 = P_1$.

Quand $\mathcal{P}$ est constitué de distributions produits, les auteurs de [Geiger et al., 2001, Whiley and Titterington, 2002] montrent que les mélanges finis de distributions de $\mathcal{P}$ sont identifiables sous quelques conditions indiquées plus bas. Ce résultat théorique implique que les paramètres du mélange $p \cdot P(.|+) + (1-p) \cdot P(.|-)$ qui spécifient les classifieurs naïfs de Bayes (lorsque $\mathcal{Y} = \{+,-\}$) sont identifiables uniquement à partir de la distribution $P(.)$ sur $\mathcal{X}$ et donc estimables uniquement à partir de données non étiquetées (à condition que le nombre d'attributs m soit au moins égal à 3 et à la détermination des classes près).

Dans [Geiger et al., 2001], les auteurs fournissent les formules qui permettent de calculer les paramètres du classifieur naïf de Bayes lorsque $\mathcal{X} = \{0,1\}^m$. Nous utilisons ces formules dans le chapitre 3 pour construire un algorithme d'apprentissage des classifieurs naïfs de Bayes en contexte non supervisé, nous les présentons donc ici pour $\mathcal{Y} = \{+,-\}$ en occultant preuves et explications.

Soient $p = P(+)$, $p_j = P(x^j = 1|y = +)$ et $q_j = P(x^j = 1|y = -)$ les paramètres du classifieur naïf de Bayes lorsque $\mathcal{X} = \{0,1\}^m$ et soit $z_{ij\ldots r} = P(x^i = 1, x^j = 1, ..., x^r = 1)$. Notons que les z peuvent être estimés à partir d'un ensemble de données non étiquetées. L'hypothèse d'indépendance des attributs conditionnellement aux classes implique avec ces notations que $z_{ij\ldots r} = p \cdot p_i p_j \ldots p_r + (1-p) \cdot q_i q_j \ldots q_r$. Soient $s, t_1, ..., t_m, u_1, ..., u_m$ les paramètres définis par la transformation suivante :

$$p = \frac{s+1}{2},\ p_i = t_i + (1-s)u_i,\ q_i = t_i - (1+s)u_i \tag{2.25}$$

Considérons également la transformation définie récursivement sur les paramètres z par $z_{ij} \leftarrow z_{ij} - z_i z_j$, $z_{ijr} \leftarrow z_{ijr} - z_{ij} z_r - z_{ir} z_j - z_{jr} z_i - z_i z_j z_r$, etc. Les nouveaux paramètres s, t_i, u_i peuvent alors être calculés de la façon suivante :

$$t_i = z_i \tag{2.26}$$

$$u_1 = \pm \frac{1}{z_{23}} \sqrt{z_{12} z_{13} z_{23} + (z_{123})^2/4} \tag{2.27}$$

$$s = -\frac{z_{123}}{2 u_1 z_{23}} \tag{2.28}$$

$$u_i = \frac{z_{1i}}{(1-s^2)u_1} \text{ pour } i > 1 \tag{2.29}$$

et les paramètres du classifieur peuvent être calculés en utilisant les formules (2.25). Le problème de permutation des classes implique que deux modèles peuvent être calculés (en fonction du signe de u_1). Aucune étude expérimentale des estimations que permettent d'obtenir ces formules et en particulier de la vitesse de convergence n'est proposée dans ces travaux. Nous proposons, au chapitre 3, un algorithme basé sur ces formules permettant d'estimer, à partir d'un ensemble S_{unl}, les paramètres du classifieur naïf de Bayes.

2.3 Séparateurs linéaires et algorithme du perceptron

Cette section préliminaire présente les séparateurs linéaires et un algorithme permettant de les apprendre à partir d'un ensemble de données classées : le perceptron. Différentes façons de mettre en pratique cet algorithme sont présentées. De même, l'apprentissage de ces séparateurs à partir de données corrompues par du bruit de classification uniforme CN avec des versions adaptées de l'algorithme du perceptron proposées dans la littérature, est présenté. Nous proposons alors dans les chapitres suivants une extension du bruit CN et une adaptation de l'algorithme du perceptron basée sur ces algorithmes.

2.3.1 Séparation et séparateur linéaire

Soient $\mathcal{X} = \mathbb{R}^m$, $\mathcal{Y} = \{1, -1\}$ et $S_{lab} = \{(x_1, c(x_1)), ..., (x_l, c(x_l))\}$ un ensemble d'exemples classés tel que pour tout $i \in \{1, ..., l\}$, $x_i \in \mathcal{X}$ et $c(x_i) \in \mathcal{Y}$. On dit qu'un échantillon S_{lab} est *séparable* par un hyperplan H de $\mathbb{R}^m$ si les exemples positifs ($x \in \mathcal{X}$ tels que $c(x) = 1$) et négatifs ($c(x) = -1$) de S_{lab} se trouvent de part et d'autre de cet hyperplan (exemple Figure 2.9). On fera toujours référence à un ensemble S_{lab} *linéairement séparable* si S_{lab} est séparable par un hyperplan H passant par l'origine. Notons que tout ensemble S_{lab} séparable par un hyperplan H qui ne passe pas par l'origine peut être transformé en un ensemble S'_{lab} linéairement séparable par un hyperplan H' passant par l'origine en ajoutant une coordonnée égale à 1 à tous les exemples de S_{lab}.

Un hyperplan H de $\mathbb{R}^m$ passant par l'origine peut être identifié par un de ses vecteurs normaux $w \in \mathbb{R}^m$ tel que $||w|| = 1$ et $\forall x \in \mathcal{X}$, $x \in H$ si et seulement si $w \cdot x = 0$. Nous ferons toujours référence au vecteur normal w tel que tout exemple positif $(x, 1)$ satisfait $w \cdot x > 0$. On se passe alors de la notation H pour un hyperplan pour celle de w.

On dit qu'un hyperplan w sépare les exemples de S_{lab} avec une marge $\sigma \geq 0$ si $min_{x \in S_{lab}} \ |cos(w, x)| = min_{x \in S_{lab}} \ |w \cdot \bar{x}| = min_{x \in S_{lab}} \ w \cdot c(x)\bar{x} = \sigma$ avec $\bar{x} = \frac{x}{||x||}$ et en notant que $\forall (x, c(x)) \in S_{lab}$, $c(x)x$ est un exemple positif. On notera w^* l'hyperplan qui sépare S_{lab} avec une marge maximale, w^* est l'hyperplan optimal séparant S_{lab}. Dans ce contexte, l'objectif de l'apprentissage supervisé est d'inférer, à partir d'un échantillon S_{lab} linéairement séparable avec une marge $\sigma > 0$, un hyperplan w qui sépare les exemples positifs et négatifs de S_{lab}, c'est-à-dire $\forall (x, 1) \in S_{lab}$, $w \cdot x > 0$ et $\forall (x, -1) \in S_{lab}$, $w \cdot x < 0$, qui peut être synthétisé par $\forall (x, c(x)) \in S_{lab}$, $w \cdot c(x)x > 0$.

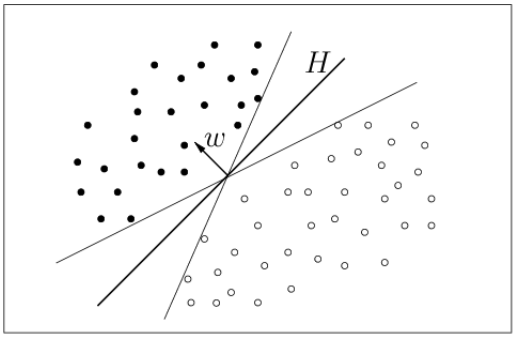


Figure 2.9. Séparation d'un ensemble de données classées de $\mathbb{R}^2$ par un hyperplan H.

2.3.2 Algorithme du perceptron [Rosenblatt, 1958]

Le perceptron est un modèle de réseau de neurones formels avec un algorithme d'apprentissage associé proposé en 1958 par Franck Rosenblatt [Rosenblatt, 1958]. Il permet d'apprendre un séparateur linéaire à partir d'un ensemble de données S_{lab} linéairement séparables. La figure 2.10 illustre ce simple réseau de neurones et son fonctionnement basé sur celui d'un neurone biologique. Les entrées (éléments de description d'une donnée) d'un neurone, pondérées individuellement, permettent à celui-ci, par une combinaison de ces entrées, de calculer une valeur de sortie qui active ou non un message de sortie (qui classe positivement ou négativement la donnée en entrée) en fonction d'un seuil fixé.

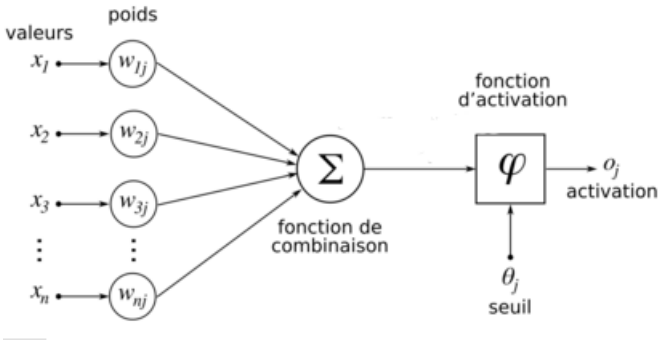


Figure 2.10. Le perceptron à seuil.

L'algorithme itératif qui lui est associé permet d'apprendre le poids associé à chaque entrée à partir d'un ensemble de données classées S_{lab} linéairement séparables de telle façon que les poids appris permettent au perceptron de classer correctement chacune des données d'apprentissage. Ces poids sont les coefficients d'un hyperplan w qui sépare les données. On considère ici les séparateurs linéaires passant par l'origine, le seuil vaut dans ce cas zéro et le perceptron classe alors positivement toutes les données x qui lui sont présentées en entrée telles que $w \cdot x > 0$ et classe toutes les autres négativement.

L'algorithme 4 donne un schéma de cet algorithme. A partir d'un vecteur w initialement nul, l'algorithme du perceptron met à jour itérativement les coordonnées de w à l'aide d'un vecteur de mise à jour x_{upd} positif ($w^* \cdot x_{upd} > 0$), mal classé par l'hyperplan courant w ($w \cdot x_{upd} < 0$), et suffisamment éloigné de w^* ($cos(w^*, x_{upd}) > 0$) pour assurer que la convergence de l'algorithme se fait en temps raisonnable.

Algorithme 4 Schéma d'algorithme du perceptron [Rosenblatt, 1958]

Entrée: $S_{lab} = \{(x_1, c(x_1)), ..., (x_l, c(x_l))\}$ linéairement séparé par un hyperplan w^*

$w = \overrightarrow{0}$
Tant que $\exists\ (x, c(x)) \in S_{lab}$ tel que $w \cdot c(x)x < 0$ **faire**
  Soit x_{upd} un vecteur mal classé par w, *i.e.* tel que $w^* \cdot x_{upd} > 0$ et $w \cdot x_{upd} < 0$
  $w = w + x_{upd}$
Fin Tant que

Sortie: Un hyperplan w tel que $\forall (x, c(x)) \in S_{lab}$, $w \cdot c(x)x > 0$

La figure 2.11 illustre la partition de S_{lab} induite par l'hyperplan optimal w^* et par l'hyperplan courant w du perceptron lorsque $\mathcal{X} = \mathbb{R}^2$. On note :

- $S^+ = \{x \in \mathbb{R}^m \mid w^* \cdot x > 0\}$, les points de $\mathbb{R}^m$ classés + par w^*
- $S^- = \{x \in \mathbb{R}^m \mid w^* \cdot x < 0\}$, les points de $\mathbb{R}^m$ classés − par w^*
- $S_+ = \{x \in \mathbb{R}^m \mid w \cdot x > 0\}$, les points de $\mathbb{R}^m$ classés + par w
- $S_- = \{x \in \mathbb{R}^m \mid w \cdot x < 0\}$, les points de $\mathbb{R}^m$ classés − par w
- $S^\alpha_\beta = S^\alpha \cap S_\beta \ \ (\alpha, \beta \in \{+, -\})$, les points de $\mathbb{R}^m$ classés α par w^* et β par w

Avec ces notations, le vecteur x_{upd} de mise à jour du perceptron doit donc se situer dans l'ensemble $S^+_- = \{x \in \mathbb{R}^m \mid w^* \cdot x > 0 \text{ et } w \cdot x < 0\}$. Sous sa forme usuelle, le vecteur de mise à jour x_{upd} de l'algorithme du perceptron vaut $c(x)\bar{x}$ avec $(x, c(x)) \in S_{lab}$ mal classé par le perceptron courant et $\bar{x} = \frac{x}{||x||}$. Il faut remarquer que l'ensemble des points x de S_{lab} incorrectement classés par w (dénotés à partir d'ici par x_B) se situent dans les ensembles S^+_- (lorsque $c(x_B) = 1$) et S^-_+ (lorsque $c(x_B) = -1$) mais que si x_B appartient à l'ensemble $S^+_- \cup S^-_+$, alors $c(x_B)x_B$ appartient à S^+_-. En notant σ la marge associée à l'hyperplan séparateur optimal w^* et en supposant $\sigma > 0$, alors x_{upd} satisfait toutes les conditions évoquées précédemment. Cet algorithme requiert alors au plus $\frac{1}{\sigma^2}$ itérations (voir par exemple [Blum et al., 1996] pour la démonstration). Il est exponentiel dans le pire des cas mais généralement efficace dans la plupart des problèmes réels.

D'autres vecteurs x_{upd} remplissent également les conditions évoquées pour la convergence de l'algorithme. Par exemple, le choix pour x_{upd} de la somme des $c(x_B)x_B$ sur tous les $x_B \in S_{lab} \cap \{S^+_- \cup S^-_+\}$ (mal classés par le perceptron courant), ou tout autre vecteur colinéaire (mais de même sens) à cette somme, comme la moyenne ou la somme normalisée, sont des bons vecteurs de mise à jour du perceptron. Il est trivial que ces vecteurs x_{upd} appartiennent à S^+_-, et que si $\forall x \in S_{lab}$, $|cos(w^*, x)| \geq \sigma$ alors $cos(w^*, x_{upd}) \geq \sigma$.

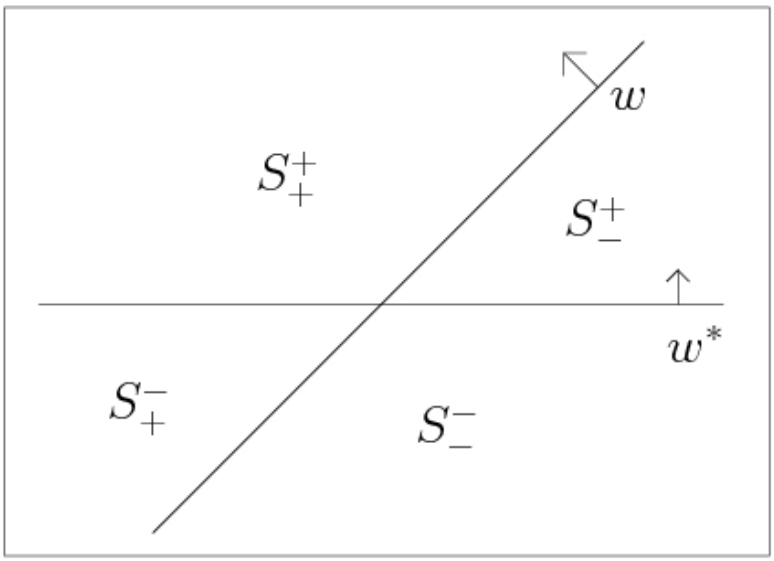


Figure 2.11. Partition définie par l'hyperplan séparateur optimal w^* et par un des hyperplans courants w de l'algorithme du perceptron ($\mathcal{X} = \mathbb{R}^2$).

Les séparateurs linéaires obtenus par l'algorithme du perceptron ne sont pas optimaux. En effet, cet algorithme assure d'obtenir un hyperplan qui sépare les données positives et négatives de S_{lab}, mais cet hyperplan n'a aucune garantie de maximiser la marge (au sens de la distance de tout point à l'hyperplan), et donc d'optimiser la séparation. Les machines à vecteurs de support (SVM) [Vapnik, 1995, Shawe-Taylor and Cristianini, 2000] permettent cette optimisation. L'extension de cette méthode à l'apprentissage de séparateurs non linéaires grâce à l'utilisation de fonctions noyaux est actuellement une méthode considérée comme l'une des plus performantes. Elle n'est pas présentée dans ces préliminaires car aucune étude ne porte sur cette méthode dans les travaux présentés dans ce mémoire. Les raisons de ce choix sont données dans les chapitres concernés de ce mémoire.

2.3.3 Algorithmes du perceptron tolérants au bruit de classification CN

Lorsque les données d'un ensemble S_{lab} linéairement séparable sont sujettes à un bruit de classification uniforme CN (Section 2.1.3), l'étiquette originale $c(x)$ de chaque exemple $x \in S_{lab}$ a été inversée avec une probabilité constante η ($\eta < 0.5$). Les exemples observés ne sont donc plus séparables dans ce cas. De plus, il n'est pas possible de savoir si un exemple est mal classé par l'hyperplan courant w de l'algorithme du perceptron puisque l'étiquette observée n'est pas fiable. La présence de bruit de classification rend donc inutilisable l'algorithme du perceptron tel que défini à la section précédente.

Plusieurs travaux ont proposé des adaptations de cet algorithme pour le rendre tolérant au bruit de classification uniforme CN [Bylander, 1994, Blum et al., 1996, Bylander, 1998]. Chacun d'entre eux propose de garder le même schéma d'algorithme mais propose un vecteur de mise à jour x_{upd} qui prend en compte la présence du bruit de classification et qui permet d'assurer la consistance de l'algorithme correspondant.

Tom Bylander est le premier à proposer une telle adaptation [Bylander, 1994]. Il propose comme vecteur de mise à jour x_{upd} pour l'algorithme du perceptron une moyenne pondérée des exemples mal classés par l'hyperplan courant w et de l'ensemble des exemples de S_{lab}. Il montre alors que ce vecteur répond aux conditions de convergence de l'algorithme évoquées dans la section précédente.

Avrim Blum, avec d'autres auteurs, montrent dans [Blum et al., 1996] que les séparateurs linéaires sont PAC-apprenables avec bruit de classification CN. Ils proposent à cette fin une adaptation de l'algorithme du perceptron de complexité polynomiale, à la différence des implémentations élémentaires de cet algorithme. L'algorithme est précédé d'un pré-traitement des données basé sur un lemme appelé *« Outlier Removal Lemma »* qui permet de supprimer les *outliers*, c'est-à-dire les points distants de la distribution de la majorité des exemples, qui pénalisent l'algorithme du perceptron.

Dans ce même article, les auteurs montrent que lorsque le taux de bruit η qui corrompt les données est connu, il est alors possible d'estimer directement la somme $\sum_{x_B \in S_{lab}} c(x_B) x_B$ des exemples mal classés de S_{lab} par l'hyperplan courant w, $c(x_B)$ étant la classe originale de x_B (attribuée par w^*) et non la classe observée. Comme indiqué à la section précédente, ce vecteur est un bon vecteur de mise à jour du perceptron. Lorsque le taux de bruit η n'est pas connu, les auteurs recommandent de scanner l'intervalle $[0, 0.5[$ pour la valeur de η avec un pas d'incrémentation petit, de lancer pour chacune des valeurs l'algorithme du

perceptron avec le vecteur de mise à jour proposé, et de sélectionner parmi les hypothèses obtenues celle qui minimise l'erreur apparente. Lorsque le bruit de classification qui corrompt les données est de type CN (avec $\eta < 0.5$), alors la sélection d'une hypothèse par le principe de minimisation du risque empirique est la meilleure stratégie envisageable. Ce résultat est donné dans l'article [Angluin and Laird, 1988] dans lequel a été introduit le bruit CN.

Deuxième partie

La question de l'existence d'une information locale impliquée dans l'appariement de résidus distants d'une protéine

Chapitre 3

Une première hypothèse sur la formulation du problème biologique

Sommaire

Nous présentons dans ce chapitre une première étude sur le problème de la prédiction de l'appariement de résidus distants dans une protéine (Section 1.2). Seul le cas des ponts disulfures (Section 1.2.1) est considéré ici. Ces travaux font suite à une étude approfondie de la littérature sur la prédiction des ponts disulfures à partir de la séquence primaire. Dans ces travaux, on trouve de nombreuses similitudes sur lesquelles nous nous sommes appuyés pour émettre et tenter de valider une hypothèse liée au choix de l'environnement local des cystéines pour la prédiction des ponts et de l'impact de ce choix sur les données d'apprentissage considérées. En effet, parmi les similarités de ces travaux, on trouve :

- les paires de cystéines appariées et non appariées comme données d'apprentissage,
- l'*environnement local* des cystéines comme principal attribut prédictif,
- la *formulation du problème* de la prédiction des ponts disulfures comme un problème de classification supervisée binaire : les paires de cystéines appariées forment les exemples positifs d'apprentissage et les paires non appariées les exemples négatifs.

Tout en respectant la plupart des choix communs à ces travaux, l'étude présentée dans ce chapitre vise à s'interroger sur le seul point de la formulation du problème comme un problème de classification supervisée. En effet, plusieurs considérations biologiques (Section 3.1.1) liées au choix de l'environnement local des cystéines appariées comme principale information pour prédire les ponts, laissent penser que si une telle information existe, le statut (la classe) des paires de cystéines non appariées est ambigu et qu'il est tout à fait envisageable qu'il ne soit pas possible de décider si ces exemples sont négatifs dans ces conditions. Dans ce cas, les seuls exemples disponibles seraient donc positifs (paires de cystéines appariées) et de classes indéterminées (paires non appariées). L'apprentissage à partir de données positives et non étiquetées est un contexte déjà étudié dans la littérature, c'est un cas particulier de l'apprentissage semi-supervisé (Section 2.1.3) dans lequel aucune information sur la distribution d'une des deux classes n'est disponible, nous l'appelons *apprentissage semi-supervisé asymétrique*.

Notre hypothèse est que si l'environnement local des cystéines est impliqué dans la formation des ponts, ce qui est toutefois loin de faire l'unanimité chez les biologistes du domaine, alors considérer les paires d'environnements locaux non appariés par leurs cystéines centrales comme des exemples de classe indéterminée devrait permettre d'améliorer les performances des classifieurs appris pour ce problème de prédiction. Afin de tester notre hypothèse, nous présentons une courte étude théorique de l'apprentissage semi-supervisé asymétrique (Section 3.2) qui montre que, dans le cadre très général de l'apprentissage statistique, le problème est mal posé : les distributions $P(.|y = +)$ et $P(.)$ sur $\mathcal{X}$ d'où proviennent les exemples positifs et non étiquetés ne déterminent pas la distribution P sur $\mathcal{X} \times \mathcal{Y}$ sous-jacente au problème considéré. Comme conséquence, même à la limite, les données positives et non étiquetées ne permettent pas de calculer un modèle du problème. Toutefois, certaines hypothèses sur les distributions permettent au problème de devenir bien posé et donc d'en déterminer un modèle.

Les distributions produits (Section 2.2.1), qui respectent l'hypothèse d'indépendance des attributs relativement à chacune des classes, sont des distributions pour lesquelles le problème de l'apprentissage semi-supervisé asymétrique est bien posé. La distribution P sur $\mathcal{X} \times \mathcal{Y}$ est donc identifiable dans ce contexte. Ce résultat n'est pas original car on sait par les travaux de [Geiger et al., 2001] (présentés à la section 2.2.4) que la distribution $P(.)$ sur $\mathcal{X}$ suffit à elle seule à déterminer la distribution P sur $\mathcal{X} \times \mathcal{Y}$ lorsque les distributions $P(.|y)$ sont des distributions produits. Néanmoins, l'information sur la distribution $P(.|y = +)$ apportée par les exemples positifs doit pouvoir être utilisée efficacement pour améliorer de façon significative les estimations des paramètres du classifieur naïf de Bayes, ses performances de classification, et pour identifier les classes.

Nous donnons à la section 3.2.2 les formules permettant de calculer tous les paramètres du classifieur naïf de Bayes à partir des distributions $P(.)$ et $P(.|y = +)$ sur $\mathcal{X}$. Ces formules permettent d'établir des estimateurs consistants de ces paramètres à partir d'un jeu de données positives et non étiquetées. Afin de comparer les estimations obtenues avec ces estimateurs et celles que l'on obtient à partir de données non étiquetées seulement, en utilisant les formules données dans [Geiger et al., 2001], nous proposons en section 3.2.3 un algorithme pour estimer les paramètres des classifieurs naïfs de Bayes à partir d'un ensemble de données non classées.

Une première série d'expériences sur des données artificielles (Section 3.2.4) permet de constater que l'utilisation des exemples positifs en plus des données non étiquetées pour estimer les paramètres du classifieur est pertinente et que ces estimations, ainsi que les performances du classifieur qu'elles définissent, sont proches de celles obtenues en contexte semi-supervisé classique avec l'algorithme `NBSS-EM` (Section 2.2.3)

Ces deux constats confirment que l'on peut apprendre efficacement les classifieurs naïfs de Bayes à partir de données positives et non étiquetées. Grâce à cette conclusion, et sans prétendre que ce classifieur simple soit suffisant pour résoudre le problème de la prédiction des ponts disulfures, il a été choisi pour tester notre première hypothèse sur la formalisation de ce problème. La section 3.3 présente une série d'expériences sur des données biologiques qui compare les performances du classifieur appris en contexte supervisé (les paires de cystéines non appariées sont considérées comme des exemples négatifs) et en contexte semi-supervisé asymétrique (aucune hypothèse sur la classe des paires non appariées de cystéines n'est faite).

3.1 Statut des paires de résidus non appariés

La plupart des travaux portant sur la prédiction des ponts disulfures à partir de la séquence primaire (Section 1.2.1) considèrent ce problème d'apprentissage comme un problème de classification supervisée : les paires de cystéines appariées forment les exemples positifs d'apprentissage et les paires non appariées les exemples négatifs. Nous émettons dans cette section, à partir de plusieurs considérations biologiques (Section 3.1.1), l'hypothèse qu'au vu de l'information utilisée pour prédire ces ponts, ce problème devrait être considéré comme un problème de classification semi-supervisé asymétrique dans lequel seuls des exemples positifs et non étiquetés (paires non appariées) sont disponibles.

3.1.1 Considérations biologiques

Les travaux sur la prédiction des appariements de cystéines oxydées à partir de la séquence ont beaucoup de points communs, par exemple le choix de l'extraction des données à partir des protéines. En effet, pour une protéine contenant $2n$ cystéines oxydées (n ponts), les auteurs considèrent les $n(2n-1)$ paires de cystéines que l'on peut former avec ces $2n$ cystéines. Par exemple, une protéine avec 3 ponts disulfures contient 15 paires de cystéines distinctes (Figure 3.1). Les 3 paires de cystéines appariées forment les exemples positifs de ponts, les 12 autres les exemples négatifs.

Les attributs prédictifs choisis pour coder toutes ces paires de cystéines oxydées varient selon les travaux. Pourtant, une information est présente dans chacun d'entre eux comme principale information sur la paire de cystéines : leurs environnements locaux (Section 1.2.1), c'est-à-dire les quelques résidus voisins des cystéines sur la séquence. Sans considérer ici la façon de coder une paire d'environnements locaux, nous nous intéressons uniquement au choix de cette information comme principal attribut descriptif des paires de cystéines servant à prédire si la paire forme un pont ou non.

Si on fait l'hypothèse que c'est une source d'information indispensable pour prédire les ponts, c'est indéniablement dans l'idée que les environnements locaux sont en partie responsables des ponts, c'est-à-dire qu'ils influent sur la formation ou non d'un pont. On considère alors que ces paires d'environnements ont une certaine *propension* à se retrouver appariés par leurs cystéines centrales respectives, une sorte d'*affinité* entre les deux segments. Sous ces considérations, les paires de cystéines appariées formeraient alors bien des exemples positifs de paires d'environnements locaux « compatibles ».

Or, une fois appariée, une cystéine ne peut plus former de pont avec une autre cystéine oxydée que celle avec qui elle est déjà appariée. Cette contrainte d'unicité implique qu'il est très difficile de statuer sur le fait que les paires d'environnements locaux non appariés par leurs cystéines centrales aient ou non une forte *propension* à se retrouver proches dans l'espace. Il est fort probable qu'une telle paire d'environnements puisse également avoir de fortes affinités l'un pour l'autre, mais qu'un ensemble de contraintes, locales et globales, comme par exemple l'unicité d'appariement d'une cystéine, ne permettent pas la formation du pont. Cette information étant utilisée comme l'information principale pour prédire les ponts, la classe de ces exemples devient alors également ambiguë et difficile à déterminer. C'est pourquoi nous pensons qu'il est préférable de ne pas considérer ces paires comme des exemples négatifs mais comme des exemples n'apportant pas d'information sur la classe.

3.1.2 Classification supervisée ou semi-supervisée asymétrique ?

La section précédente explique pourquoi utiliser les environnements locaux d'une paire de cystéines oxydées comme principale information pour déterminer si cette paire forme un pont ou non, rend le statut des paires observées non appariées ambigu (Figure 3.1). Cette ambiguïté a des conséquences sur la catégorie de problème auquel on est confronté :

- si on considère que les paires non appariées sont des exemples négatifs, on est confronté à un problème traditionnel de classification supervisée binaire,
- si on considère que les paires non appariées sont des exemples dont on ne peut pas statuer sur la classe, on est alors face à un problème de classification semi-supervisée asymétrique (Section 2.1.3).

Pour tester l'hypothèse selon laquelle il serait préférable de ne pas statuer sur la classe des paires de cystéines non appariées, notre idée est de comparer les performances de prédiction d'un classifieur appris sur un jeu de données réelles, dans un premier temps en contexte supervisé, et dans un deuxième temps en contexte semi-supervisé asymétrique.

Toutefois, la classification semi-supervisée asymétrique est un domaine de l'apprentissage assez peu étudié et qui nécessite en premier lieu de se poser la question de savoir si c'est un problème bien posé. Une étude de ce domaine dans le cadre général de l'apprentissage statistique est mené dans la section suivante. Elle montre que sans hypothèse sur les distributions qui génèrent les données positives et non étiquetées, celles-ci ne déterminent pas un modèle du problème.

Par contre, si on fait l'hypothèse d'indépendance des attributs conditionnellement à chaque classe décrite en section 2.2.1, alors le problème devient bien posé. Les paramètres du classifieur naïf de Bayes peuvent donc être calculés dans ce cas. Nous donnons les formules pour les calculer et pour les estimer à partir de données en section 3.2.2. L'hypothèse émise dans cette section peut donc être testée avec ce classifieur, sans pour autant sous-entendre que celui-ci puisse résoudre le problème de la prédiction des ponts disulfures.

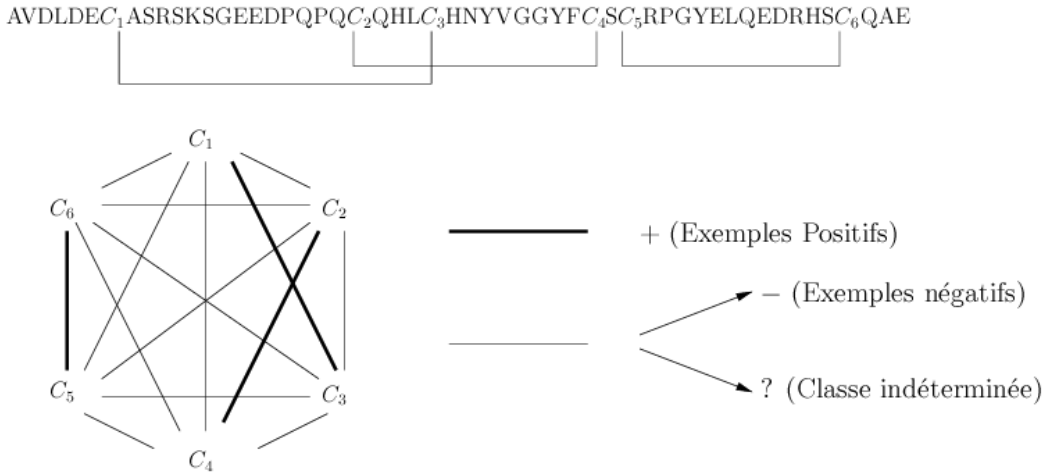


Figure 3.1. Statut des paires de résidus non appariés.

3.2 Etude de l'apprentissage semi-supervisé asymétrique

On trouve dans la littérature du domaine plusieurs travaux sur le thème de l'apprentissage supervisé binaire à partir de données positives et non étiquetées [Denis, 1998, Letouzey et al., 2000, Li and Liu, 2003, Denis et al., 2003, Yu et al., 2004, Li and Liu, 2005, Fung and Lu, 2006] (liste complète en section 2.1.3). Nous montrons dans un premier temps (Section 3.2.1) que c'est un problème mal posé sans hypothèse supplémentaire. Nous montrons ensuite que tous les paramètres qui spécifient les classifieurs naïfs de Bayes sont identifiables dans ce contexte sous des conditions très faibles, nous donnons les formules permettant leur calcul et leur estimation à partir de données positives et non étiquetées en section 3.2.2. Enfin, nous proposons plusieurs adaptations de l'algorithme naïf de Bayes dans ce contexte en section 3.2.3 et une étude expérimentale de ces algorithmes est menée sur des données artificielles (Section 3.2.4).

3.2.1 Cas général : un problème mal posé

Soient $\mathcal{X}$ un espace de description discret et $\mathcal{Y} = \{+, -\}$ l'ensemble des classes. Le cadre général de l'apprentissage statistique fait l'hypothèse de l'existence de distributions $P(.)$ sur $\mathcal{X}$ et $P(.|x)$ sur $\mathcal{Y}$ pour tout $x \in \mathcal{X}$ fixes mais inconnues (Section 2.1.2). Un problème d'apprentissage est dit bien posé si les distributions d'où proviennent les exemples d'apprentissage déterminent la totalité de ces distributions. Quand $\mathcal{Y} = \{+, -\}$, les distributions $P(.)$ sur $\mathcal{X}$ et $P(.|x)$ sur $\mathcal{Y}$, $\forall x \in \mathcal{X}$, sont déterminées par la donnée de :

- la distribution $P(.)$ sur $\mathcal{X}$,
- la distribution $P(.|y = +)$ sur $\mathcal{X}$ (notée $P(.|+)$),
- la probabilité $P(y = +)$ (notée $P(+)$).

En effet, $\forall x \in \mathcal{X}$, $P(y = +|x) = P(+|x) = \frac{P(x|+)\cdot P(+)}{P(x)}$ et $P(-|x) = 1 - P(+|x)$. L'apprentissage semi-supervisé asymétrique suppose que l'on dispose d'exemples :

- positifs, tirés selon la distribution $P(.|+)$ sur $\mathcal{X}$,
- non étiquetés, tirés selon la distribution $P(.)$ sur $\mathcal{X}$.

On en déduit que si le paramètre $P(+)$ est déterminé par ces distributions, on peut identifier les distributions $P(.)$ sur $\mathcal{X}$ et $P(.|x)$ sur $\mathcal{Y}$ pour tout $x \in \mathcal{X}$, l'apprentissage semi-supervisé asymétrique est un problème bien posé et un échantillon d'exemples positifs et non étiquetés permet alors d'estimer un modèle du problème.

Or, même en supposant que l'on dispose de toutes les informations sur les distributions $P(.)$ et $P(.|+)$ sur $\mathcal{X}$, on constate qu'elles ne déterminent pas la probabilité d'observer un exemple positif $P(+)$ sans information supplémentaire sur ces distributions.

Proposition 3.1. *Sans information supplémentaire, le paramètre $P(+)$ n'est pas déterminé par la donnée des distributions $P(.)$ sur $\mathcal{X}$ et $P(.|+)$ sur $\mathcal{X}$.*

Démonstration.

Soit $r = \min\{ \frac{P(x)}{P(x|+)} \mid x \in \mathcal{X} \text{ et } P(x|+) \neq 0\}$.

Pour tout $\lambda \in]0, r]$, il existe une distribution P' définie sur $\mathcal{X} \times \mathcal{Y}$ par :

- $P'(x, y) = \lambda \cdot P(x|+)$ si $P(x) \neq 0$ et $y = +$,
- $P'(x, y) = P(x) - \lambda \cdot P(x|+)$ si $P(x) \neq 0$ et $y = -$,
- $P'(x, y) = 0$ sinon.

En effet, $\sum_{(x,y)\in\mathcal{X}\times\mathcal{Y}} P'(x, y) = \sum_{P(x)\neq 0} \lambda \cdot P(x|+) + P(x) - \lambda \cdot P(x|+) = \sum_{P(x)\neq 0} P(x) = 1.$

Avec cette définition de P', on a alors :

- $P'(+) = \sum_{x\in\mathcal{X}} P'(x, +) = \lambda$,
- $P'(x|+) = \frac{P'(x,+)}{\lambda} = P(x|+)$,
- $P'(x|-) = \frac{P(x)-\lambda\cdot P'(x|+)}{1-\lambda}$,
- $P'(x) = \lambda \cdot P'(x|+) + (1 - \lambda) \cdot P'(x|-) = P(x)$.

Cette définition permet de montrer que $\forall \lambda_1,\ \lambda_2 \in]0, r]$ tels que $\lambda_1 \neq \lambda_2$, les distributions P'_1 et P'_2 associées vérifient :

- $P'_1(x) = P'_2(x)\ \forall x \in \mathcal{X}$,
- $P'_1(x|+) = P'_2(x|+)\ \forall x \in \mathcal{X}$,
- $P'_1(+) \neq P'_2(+)$.

L'ensemble des λ valides étant l'intervalle $]0, r]$, le paramètre $P(+)$ n'est donc pas déterminé par les distributions $P(.)$ et $P(.|+)$ sur $\mathcal{X}$. En d'autres termes, une infinité de modèles distincts n'entrent pas en contradiction avec les seules distributions dont on dispose en apprentissage semi-supervisé asymétrique : le problème est mal posé. □

Remarque : dans certains cas, le paramètre $P(+)$ peut être déterminé par les distributions $P(.)$ et $P(.|+)$ sur $\mathcal{X}$. Deux exemples suffisent à s'en convaincre :

- Dans un cadre déterministe où pour tout $x \in \mathcal{X}$, $P(+|x) = 1$ ou $P(+|x) = 0$, on a :

$$P(+) = \sum_{x\in\mathcal{X}} P(x) \cdot P(+|x) = \sum_{x\in\mathcal{X}/P(x|+)\neq 0} P(x)$$

- De même, si on fait l'hypothèse d'indépendance des attributs conditionnellement à chaque classe (hypothèse naïve de Bayes), le problème devient bien posé. D'une part, ce résultat peut être considéré comme une conséquence d'un résultat donné dans les travaux [Geiger et al., 2001], dans lesquels les auteurs montrent que, sous certaines conditions, la seule distribution $P(.)$ sur $\mathcal{X}$ suffit à déterminer tous les paramètres des modèles naïfs de Bayes, et d'autre part, nous montrons dans la section suivante que les distributions $P(.)$ et $P(.|+)$ sur $\mathcal{X}$ dont on dispose permettent d'obtenir ce résultat sous des conditions moins fortes que celles imposées dans les travaux de [Geiger et al., 2001] (Section 2.2.4).

3.2.2 Calcul des paramètres du classifieur naïf de Bayes

Plusieurs auteurs [Denis et al., 2002a, Liu et al., 2002, Denis et al., 2003, Li and Liu, 2005] ont proposé des adaptations de l'algorithme naïf de Bayes en contexte semi-supervisé asymétrique. Le problème de l'estimation du paramètre $P(+)$ évoqué à la section précédente est évidemment indiqué. Certains des auteurs le considèrent connu et d'autres tentent de ramener le problème en contexte semi-supervisé par des heuristiques (Section 2.2.4). Or, ce paramètre est déterminé par les distributions $P(.)$ et $P(.|+)$ sur $\mathcal{X}$ (et donc tous les autres paramètres du modèle), nous fournissons ici les formules pour le calculer à partir de ces distributions et l'estimer à partir d'un ensemble de données lorsque le nombre d'attributs est au moins égal à 2. Ces formules permettent d'intégrer l'information apportée par les exemples positifs et sont valides pour des conditions plus faibles que celles imposées par les travaux de [Geiger et al., 2001].

Théorème 3.1. *Soient $\mathcal{Y} = \{+, -\}$ et P une distribution sur un espace discret $\mathcal{X} = \prod_{j=1}^{m} \mathcal{X}^j$ telle que P satisfait l'hypothèse naïve de Bayes. Alors les distributions $P(.)$ et $P(.|+)$ sur $\mathcal{X}$ déterminent la probabilité $P(+)$ dès lors qu'il existe au moins deux attributs x^i et x^j tels que $P(x^i = .|+) \neq P(x^i = .)$ et $P(x^j = .|+) \neq P(x^j = .)$.*

Démonstration.

Considérons tout d'abord les valeurs extrêmes de $P(+)$:

- $P(+)$=0 est impossible du fait de l'existence de données positives,
- si $P(+)$=1, alors $\forall j \in \{1, ..., m\}$ et $\forall k \in \mathcal{X}^j$, $P(x^j = k|+) = P(x^j = k)$, ce qui contredit les hypothèses.

Considérons maintenant le cas général $0 < P(+) < 1$ et notons que :

$$P(x^j = k|+) \neq P(x^j = k) \Leftrightarrow P(x^j = k|+) \neq P(x^j = k|-)$$

De la même façon qu'à la section 2.2 qui présente le classifieur naïf de Bayes, nous notons p_{jk} les probabilités $P(x^j = k|+)$ (connues d'après les hypothèses) et $q_{jk} = P(x^j = k|-)$. De plus, nous notons respectivement α_{ik} et $\alpha_{ik,jl}$ les probabilités $P(x^i = k)$ et $P(x^i = k \cap x^j = l)$, également connues d'après les hypothèses.

Pour toute paire (i, j) d'attributs distincts et pour toute paire (k, l) de valeurs de ces attributs ($k \in \mathcal{X}^i, l \in \mathcal{X}^j$), le système suivant peut être posé :

$$\begin{cases} \alpha_{ik} = p_{ik} \cdot P(+) \ + \ q_{ik} \cdot (1 - P(+)) \\ \alpha_{jl} = p_{jl} \cdot P(+) \ + \ q_{jl} \cdot (1 - P(+)) \\ \alpha_{ik,jl} = p_{ik} \cdot p_{jl} \cdot P(+) \ + \ q_{ik} \cdot q_{jl} \cdot (1 - P(+)) \end{cases} \tag{3.1}$$

Soient (i, j) une paire d'indices d'attributs distincts et (k, l) une paire de valeurs de ces attributs telle que $p_{ik} \neq q_{ik}$ et $p_{jl} \neq q_{jl}$ (condition du théorème). D'après le système (3.1), on déduit :

$$q_{ik} = \frac{\alpha_{ik} - p_{ik} \cdot P(+)}{1 - P(+)} \quad \text{et} \quad q_{jl} = \frac{\alpha_{jl} - p_{jl} \cdot P(+)}{1 - P(+)} \tag{3.2}$$

En remplaçant ces expressions de q_{ik} et q_{jl} dans la troisième équation du système (3.1), on obtient, après simplification, l'équation du premier degré suivante :

$$P(+) \cdot (p_{ik} \cdot p_{jl} - \alpha_{ik} \cdot p_{jl} - \alpha_{jl} \cdot p_{ik} + \alpha_{ik,jl}) = \alpha_{ik,jl} - \alpha_{ik} \cdot \alpha_{jl} \tag{3.3}$$

Le facteur multiplicatif de $P(+)$ peut également s'écrire, en remplaçant α_{ik}, α_{jl} et $\alpha_{ik,jl}$ par leurs définitions et après simplification :

$$p_{ik} \cdot p_{jl} - \alpha_{ik} \cdot p_{jl} - \alpha_{jl} \cdot p_{ik} + \alpha_{ik,jl} = (1 - P(+)) \cdot (p_{ik} - q_{ik}) \cdot (p_{jl} - q_{jl}) \tag{3.4}$$

On a supposé $P(+) \neq 1$, $p_{ik} \neq q_{ik}$ et $p_{jl} \neq q_{jl}$, ce terme ne peut donc pas être nul et :

$$P(+) = \frac{\alpha_{ik,jl} - \alpha_{ik} \cdot \alpha_{jl}}{p_{ik} \cdot p_{jl} - \alpha_{ik} \cdot p_{jl} - \alpha_{jl} \cdot p_{ik} + \alpha_{ik,jl}} \tag{3.5}$$

□

La formule (3.5) clôt la démonstration du théorème. La condition de ce théorème est très faible puisqu'elle exige uniquement qu'au moins deux attributs soient pertinents (différemment distribués dans les classes) pour le problème de classification considéré.

Cette formule conduit à une estimation naturelle du paramètre $P(+)$ lors d'un apprentissage à partir d'un ensemble S_{pos} de données positives et d'un ensemble S_{unl} de données non étiquetées. Pour une paire (i, j) d'attributs distincts et une paire (k, l) de valeurs de ces attributs, nous notons $\hat{\alpha}_{ik,jl}$, $\hat{\alpha}_{ik}$, et $\hat{\alpha}_{jl}$ les estimations des paramètres $\alpha_{ik,jl}$, α_{ik}, et α_{jl} obtenues sur les données de l'ensemble S_{unl}, et $\hat{p}_{ik}$, $\hat{p}_{jl}$ les estimations des paramètres p_{ik}, p_{jl} obtenues à partir des données positives de l'ensemble S_{pos}.

L'estimation de $P(+)$ avec une paire (i, j) d'attributs (pour une paire de valeurs (k, l) de ces attributs) n'étant pas raisonnable, dû à la variance de ces estimations, nous proposons de sommer, pour toutes les paires d'attributs (i, j) $(i, j \in \{1, ..., m\}, i \neq j)$ et pour toutes les paires de valeurs (k, l) de ces attributs $(k \in \mathcal{X}^i, l \in \mathcal{X}^j)$, les équations (3.3) correspondantes. On obtient un estimateur consistant $\hat{P}(+)$ de $P(+)$ défini par :

$$\hat{P}(+) = \frac{\sum\limits_{\substack{i,j \in \{1,...,m\}, i \neq j \\ k \in \mathcal{X}^i, l \in X^j}} \hat{\alpha}_{ik,jl} - \hat{\alpha}_{ik} \cdot \hat{\alpha}_{jl}}{\sum\limits_{\substack{i,j \in \{1,...,m\}, i \neq j \\ k \in \mathcal{X}^i, l \in X^j}} \hat{p}_{ik} \cdot \hat{p}_{jl} - \hat{\alpha}_{ik} \cdot \hat{p}_{jl} - \hat{\alpha}_{jl} \cdot \hat{p}_{ik} + \hat{\alpha}_{ik,jl}} \tag{3.6}$$

D'autres estimateurs de $P(+)$ peuvent être construits à partir de la formule (3.5). Une étude expérimentale a été menée et montre que celui-ci est plus précis que la moyenne des différents estimateurs que l'on peut obtenir sur toutes les paires d'attributs et de valeurs d'attributs ou encore du choix médian de ces estimateurs. Nous n'avons pas mené plus loin ces expériences car cet estimateur permet déjà d'obtenir de très bonnes estimations.

Une étude expérimentale de cet estimateur est menée sur des données artificielles à la section 3.2.4. Ces expériences semblent montrer que les estimations calculées avec cet estimateur sont précises, même pour un nombre de données assez faible. Toutefois, l'étude théorique de cet estimateur, et en particulier de sa vitesse de convergence, n'a pas été menée et reste une question ouverte de cette étude.

3.2.3 Propositions d'algorithmes pour ce contexte d'apprentissage

A partir des résultats présentés dans la section précédente et de ceux présentés dans la section préliminaire 2.2 (page 56) sur le calcul du classifieur naïf de Bayes dans différents contextes d'apprentissage, nous proposons ici plusieurs adaptations de l'algorithme naïf de Bayes pour calculer les paramètres de ces classifieurs à partir de données positives et non étiquetées ou à partir de données non étiquetées seulement.

Calcul direct des paramètres du classifieur : algorithme `NBSSA`

Tous les paramètres des classifieurs naïfs de Bayes sont estimables directement à partir d'un ensemble S_{pos} de données positives et d'un ensemble S_{unl} de données non classées.

En effet, les estimateurs $\hat{\alpha}_{ik,jl}$, $\hat{\alpha}_{ik}$ et $\hat{p}_{ik}$ des paramètres $\alpha_{ik,jl}$, α_{ik} et p_{ik} peuvent être obtenus sur les ensembles S_{pos} et S_{unl}. La formule (3.6) permet d'obtenir une estimation $\hat{p} = \hat{P}(+)$ du paramètre $p = P(+)$. Les derniers paramètres manquants q_{ik} du modèle peuvent être estimés selon la formule (3.2) :

$$\hat{q}_{ik} = \frac{\hat{\alpha}_{ik} - \hat{p}_{ik} \cdot \hat{p}}{1 - \hat{p}} \quad \forall i \in \{1, ..., m\}, \forall k \in \mathcal{X}^i \tag{3.7}$$

Le modèle obtenu $\hat{\theta} = \{\hat{p}, \hat{p}_{ik}, \hat{q}_{ik}\}$ est donc une estimation consistante du modèle $\theta = \{p, p_{ik}, q_{ik}\}$ qui spécifie le classifieur naïf de Bayes. L'algorithme correspondant à cette estimation directe des paramètres du classifieur est noté `NBSSA` (Algorithme 5).

Algorithme 5 `NBSSA` - Algorithme naïf de Bayes semi-supervisé asymétrique
Entrée: S_{pos}, S_{unl}

1 - Calculer $\hat{\alpha}_{ik,jl}$, $\hat{\alpha}_{ik}$ et $\hat{p}_{ik}$ $\forall i, j \in \{1, ..., m\}$, $k \in \mathcal{X}^i$, et $l \in \mathcal{X}^j$ à partir de S_{unl} et S_{pos}
2 - Calculer $\hat{p} = \hat{P}(+)$ avec la formule (3.6)
3 - Calculer $\hat{q}_{ik}$ $\forall i \in \{1, ..., m\}$, $k \in \mathcal{X}^i$, avec la formule (3.7)

Sortie: un modèle $\hat{\theta} = \{\hat{p}, \hat{p}_{ik}, \hat{q}_{ik}\}$

Maximisation locale de la vraisemblance avec E.M. : algorithme `NBSSA-EM`

L'algorithme `NBSSA` fournit un moyen direct d'estimer tous les paramètres des classifieurs naïfs de Bayes à partir d'un ensemble de données positives et non étiquetées. Néanmoins, on peut souhaiter trouver un modèle qui maximise la vraisemblance de ces deux ensembles de données.

De façon similaire au cas semi-supervisé classique (Section 2.2.2), on définit β comme la probabilité d'observer un exemple positif ($1 - \beta$ est donc la probabilité d'observer un exemple non étiqueté), et θ' le modèle $\theta = \{p, p_{jk}, q_{jk}\}$ complété du paramètre β ($\theta' = \theta \cup \{\beta\}$). Soient $S_{pos} = \{(x_1, +), ..., (x_l, +)\}$ un ensemble d'exemples positifs, n_{jk} le nombre de données de S_{pos} telles que $x^j = k$, et $S_{unl} = \{x'_1, ..., x'_{l'}\}$ un ensemble de données non étiquetées, la vraisemblance $L(\theta', S_{pos}, S_{unl})$ des ensembles S_{pos} et S_{unl} dans le modèle θ' s'écrit alors :

$$\begin{aligned} L(\theta', S_{pos}, S_{unl}) &= \prod_{i=1}^{l} \beta \cdot P(x_i, +|\theta) \cdot \prod_{i=1}^{l'} (1-\beta) \cdot P(x'_i|\theta) \\ &= \beta^l \cdot L(\theta, S_{pos}) \cdot (1-\beta)^{l'} \cdot L(\theta, S_{unl}) \end{aligned} \tag{3.8}$$

avec :

$$L(\theta, S_{pos}) = \prod_{i=1}^{l} P(+|\theta) \Big[\prod_{j=1}^{m} P(x_i^j | y = +, \theta) \Big] = p^l \cdot \prod_{\substack{j \in \{1,\ldots,m\} \\ k \in \mathcal{X}^j}} p_{jk}^{n_{jk}} \tag{3.9}$$

et

$$L(\theta, S_{unl}) = \prod_{i=1}^{l'} \Big(p \cdot \prod_{\substack{j \in \{1,\ldots,m\} \\ k / x_i'^j = k}} p_{jk} + (1-p) \cdot \prod_{\substack{j \in \{1,\ldots,m\} \\ k / x_i'^j = k}} q_{jk} \Big) \tag{3.10}$$

Pour les mêmes raisons que celles évoquées en section 2.2.2 pour un contexte semi-supervisé classique, les paramètres du classifieur naïf de Bayes qui maximisent la vraisemblance des données ne sont pas déductibles de l'expression de la vraisemblance de (S_{pos},S_{unl}) dans le modèle θ' (Formule 3.8). Il faut donc recourir à des méthodes comme E.M. pour tenter de maximiser la vraisemblance.

La section 2.2.3 présente la méthode E.M. ainsi que son application, proposée par [Nigam et al., 1998], au calcul des paramètres de ce classifieur en contexte semi-supervisé (à partir des deux ensembles S_{lab} et S_{unl}). L'algorithme `NBSS-EM` issu de ces travaux permet d'obtenir, à partir d'un modèle initial calculé sur l'ensemble des données étiquetées S_{lab}, et en utilisant les données non étiquetées de S_{unl}, un modèle qui maximise localement la vraisemblance de l'ensemble des données de S_{lab} et S_{unl}.

Dans [Liu et al., 2002], les auteurs proposent pour le cas asymétrique une adaptation de cet algorithme, notée `I-EM`, qui considère comme modèle initial le modèle appris par l'algorithme `NB` supervisé sur les données positives de S_{pos} et sur les données de S_{unl} alors considérées comme négatives. Les auteurs indiquent que cette méthode n'est pas performante et que l'initialisation de la méthode E.M. avec le modèle choisi en est responsable.

Or, nous venons de montrer que l'on peut estimer tous les paramètres du classifieur naïf de Bayes à partir de données positives et non étiquetées. Le modèle calculé par l'algorithme `NBSSA` peut donc être utilisé comme modèle initial dans l'algorithme `NBSS-EM` en remplacement de celui calculé sur l'ensemble S_{lab} considéré en contexte semi-supervisé. De plus, les formules données dans [Nigam et al., 1998] pour recalculer tous les paramètres du modèle à chaque itération de l'algorithme restent valides (formules (2.22), (2.23) et (2.24), section 2.2.3 page 61). Notons toutefois des simplifications dans les formules dues à l'absence d'exemples négatifs : $n_p = l$, $m_{jk} = 0 \ \forall j \in \{1, ..., m\}, \forall k \in \mathcal{X}^j$. Les formules (2.22), (2.23) et (2.24) s'écrivent alors :

Algorithme 6 `NBSSA-EM` - Algorithme naïf de Bayes semi-supervisé asymétrique + E.M.
Entrée: S_{pos}, S_{unl}

1 - $\theta_0 =$ `NBSSA`(S_{pos},S_{unl})
2 - Calculer $\hat{P}(y' = +|x', \theta_n)$ et $\hat{P}(y' = -|x', \theta_n)$ $\forall x' \in S_{unl}$ avec le modèle θ_n courant
3 - Calculer θ_{n+1} avec les formules (3.11), (3.12) et (3.13)
4 - Itérer à l'étape 2 jusqu'à convergence

Sortie: le modèle θ_c obtenu après convergence

$$p = \frac{l + \sum_{i=1}^{l'} \hat{P}(y'_i = +|x'_i, \theta_n)}{l + l'} \tag{3.11}$$

$$p_{jk} = \frac{n_{jk} + \sum_{x'_i \in S_{unl}/x_i'^{j}=k} \hat{P}(y'_i = +|x'_i, \theta_n)}{l + \sum_{i=1}^{l'} \hat{P}(y'_i = +|x'_i, \theta_n)} \tag{3.12}$$

$$q_{jk} = \frac{\sum_{x'_i \in S_{unl}/x_i'^{j}=k} \hat{P}(y'_i = -|x'_i, \theta_n)}{\sum_{i=1}^{l'} \hat{P}(y'_i = -|x'_i, \theta_n)} \tag{3.13}$$

Nous appelons `NBSSA-EM` (Algorithme 6) l'adaptation de l'algorithme `NBSS-EM` en contexte semi-supervisé asymétrique décrite ici.

Calcul des paramètres en contexte non supervisé : algorithme `NB-UNL`

Les formules proposées dans [Geiger et al., 2001] (présentées à la section 2.2.4) permettent de calculer tous les paramètres du classifieur naïf de Bayes à partir de la distribution $P(.)$ sur $\mathcal{X}$ lorsque $\mathcal{X} = \{0,1\}^m$ et à la détermination des classes près (deux modèles sont calculés lorsque $\mathcal{Y} = \{+,-\}$). Ces formules ne permettent pas de prendre en compte l'information apportée par les exemples positifs sur la distribution $P(.|+)$ sur $\mathcal{X}$, mais elles permettent d'estimer tous les paramètres du classifieur uniquement à partir de données non étiquetées. Ces estimations sont donc pertinentes dans le contexte étudié ici.

Nous proposons un algorithme, noté `NB-UNL`, (Algorithme 7), basé sur ces formules, pour estimer les paramètres du classifieur à partir d'un ensemble de données non étiquetées S_{unl} lorsque $\mathcal{X} = \{0,1\}^m$ et $\mathcal{Y} = \{+,-\}$. La définition des notations utilisées dans cet algorithme peut être trouvée en section 2.2.4. La condition $\mathcal{X} = \{0,1\}^m$ est très restrictive. Néanmoins, les expériences menées dans la section suivante sur des données artificielles montrent que, même dans ce cas, les estimateurs nécessitent un très grand nombre de données pour fournir de bonnes estimations, rendant l'algorithme inutilisable en pratique. C'est pourquoi nous ne proposons pas de généralisation de celui-ci.

Algorithme 7 `NB-UNL` - Algorithme naïf de Bayes non supervisé ($\mathcal{X}$={0,1}^{m}, $\mathcal{Y}$={+,−})
Entrée: S_{unl}

1 - Calculer les estimations des paramètres z_i, z_{ij} et z_{ijk} $\forall i,j,k \in \{1,...,m\}$ sur l'ensemble S_{unl} et leur appliquer la transformation décrite en section 2.2.4

2 - Calculer $u_1^+ = \frac{\sum_{\substack{1\leq i,j\leq m \\ i\neq j\neq 1}} \sqrt{z_{1i}z_{1j}z_{ij}+\frac{(z_{1ij})^2}{4}}}{\sum_{\substack{1\leq i,j\leq m \\ i\neq j\neq 1}} z_{ij}}$ (formule (2.27) page 63)

3 - $\forall k>1$, calculer $u_k^+ = signe(\frac{z_{1k}}{u_1^+}) \frac{\sum_{\substack{1\leq i,j\leq m \\ i\neq j\neq k}} \sqrt{z_{ki}z_{kj}z_{ij}+\frac{(z_{kij})^2}{4}}}{|\sum_{\substack{1\leq i,j\leq m \\ i\neq j\neq k}} z_{ij}|}$ (formules (2.27) et (2.29))

4 - Calculer $s^+ = -\frac{\sum_{\substack{1\leq i,j,k\leq m \\ i\neq j\neq k}} z_{ijk}}{\sum_{\substack{1\leq i,j,k\leq m \\ i\neq j\neq k}} 2u_i^+ z_{jk}}$ (formule (2.28) page 63)

5 - Calculer les paramètres du premier modèle θ^+ avec les z_i, u_i^+ et s^+ ((2.25) page 63)
6 - Calculer les paramètres du second modèle θ^- avec les z_i, $u_i^- = -u_i^+$ et $s^- = -s^+$

Sortie: les modèles θ^+ et θ^-

Variantes de l'algorithme `NBSSA-EM`

L'algorithme `NBSSA-EM` utilise comme modèle initial θ_0 le modèle calculé par l'algorithme `NBSSA` sur les données des ensembles S_{pos} et S_{unl}. Nous avons considéré deux variantes de ce choix de modèle initial en entrée de la méthode E.M. :

- la première variante consiste à tirer des modèles θ_0 au hasard et à répéter l'algorithme `NBSSA-EM` avec ces modèles θ_0 comme modèles initiaux. La sélection d'un des modèles θ_c peut ensuite se faire en sélectionnant le modèle θ_c qui maximise la vraisemblance des données de S_{pos} et S_{unl}. Les expériences menées dans la section suivante sur des données artificielles ne présentent pas les résultats de cette variante car ses performances en estimation et en classification sont identiques à celles de l'algorithme `NBSSA-EM`, mais elle exige de nombreuses répétitions pour arriver aux mêmes résultats et donc un temps de calcul supérieur. Le calcul du modèle θ_0 avec l'algorithme `NBSSA` reste donc un choix préférable,

- la seconde variante de l'algorithme `NBSSA-EM` consiste à utiliser le modèle θ appris par l'algorithme `NB-UNL` sur les données de S_{unl} comme modèle initial θ_0. Cette variante obtient des performances similaires en estimation et en classification à l'algorithme `NBSSA-EM` sur des données artificielles. Toutefois, elle n'apporte pas d'intérêt en comparaison de la méthode `NBSSA-EM` car les expériences de la section suivante montrent que les modèles appris par l'algorithme `NB-UNL` sont moins précis que ceux appris par l'algorithme `NBSSA`. La méthode E.M. étant sensible à ses paramètres initiaux, il est plus risqué de lui fournir ce modèle en entrée. De plus, le temps de calcul nécessaire à l'algorithme `NB-UNL` pour calculer les paramètres du modèle est

supérieur au temps pris par l'algorithme NBSSA.

3.2.4 Etude expérimentale comparative des algorithmes

Cette section présente une étude expérimentale, sur des données artificielles, des algorithmes proposés dans la section précédente (NBSSA, NBSSA-EM et NB-UNL) ainsi que de l'algorithme NBSS-EM présenté en section 2.2.3. Ces algorithmes permettent tous de calculer l'ensemble des paramètres d'un classifieur naïf de Bayes à partir de données.

Les algorithmes NBSSA et NBSSA-EM utilisent deux ensembles S_{pos} et S_{unl} de données respectivement positives et non étiquetées pour calculer les paramètres d'un modèle. C'est moins d'information qu'en contexte semi-supervisé dans lequel l'algorithme NBSS-EM dispose également de données négatives, et plus qu'en contexte non supervisé dans lequel l'algorithme NB-UNL ne dispose pas de données classées.

L'objectif de ces expériences est donc de pouvoir comparer les performances, sur la tâche d'estimation des paramètres et sur la tâche de classification, des quatre algorithmes afin d'évaluer si l'information apportée par les exemples positifs en contexte asymétrique est pertinente et permet d'apprendre efficacement les classifieurs naïfs de Bayes.

Protocole expérimental

Du fait des conditions d'application de l'algorithme NB-UNL, nous considérons uniquement des attributs binaires ($\mathcal{X} = \{0,1\}^m$). Les paramètres des modèles cibles $\theta_c = \{p, p_{ik}, q_{ik}\}$ qui génèrent les données sont tirés aléatoirement à partir d'une distribution uniforme. Les distributions $P(.|+)$ et $P(.|-)$ de θ_c sont des distributions produits sur $\mathcal{X}$. Les ensembles d'apprentissage et de test sont tirés selon le modèle θ_c.

Pour chaque nombre d'attributs $m \in \{20, 50\}$, 10 modèles cibles θ_c sont générés aléatoirement. Pour chacun de ces modèles, et pour tout $l \in \{100, 300, 500\}$, 20 ensembles indépendants S_{lab} (resp. S_{unl}) de l données classées (resp. $10l$ données non classées) sont générés, ainsi qu'un ensemble S_{test} de 10000 exemples classés. Les résultats présentés dans les tables suivantes sont donc des moyennes calculées sur 200 expériences. Pour chacune des expériences, l'ensemble S_{pos} utilisé par les algorithmes NBSSA et NBSSA-EM est extrait de l'ensemble S_{lab}, son cardinal vaut donc approximativement $p \cdot |S_{lab}|$.

Le problème de savoir comment choisir parmi les deux modèles θ^+ et θ^- calculés par l'algorithme NB-UNL (identiques à permutation des classes près) n'a pas été traité. N'étant pas dans les objectifs de ces expériences, ce problème n'est pas considéré ici et le choix est simplement fait en se basant sur les paramètres des modèles cibles connus.

Comparaison des estimations du paramètre $P(+)$

Comme le montrent les sections 3.2.1 et 3.2.2, l'estimation du paramètre $p = P(+)$ est très importante en contexte semi-supervisé asymétrique. Il est donc naturel de comparer les estimations de ce paramètre de θ_c calculées par les quatre algorithmes. La table 3.1 donne l'erreur quadratique moyenne des estimations de ce paramètre, pour chacun des algorithmes, calculées sur la série d'expérience décrite précédemment. Les résultats

obtenus permettent deux conclusions intéressantes :

– Quel que soit le nombre d'exemples d'apprentissage, les estimations du paramètre $P(+)$ calculées par l'algorithme `NBSSA` (sans utiliser la méthode E.M.), sont beaucoup plus précises que celles obtenues par l'algorithme `NB-UNL`. Ce résultat montre que, même en faible quantité, les exemples positifs permettent d'estimer beaucoup plus précisément les paramètres du modèle. L'estimateur (3.6) de $P(+)$ proposé en section 3.2.2 exploite donc utilement l'information apportée par les exemples positifs.

– De plus, les estimations calculées avec l'algorithme `NBSSA-EM` sont aussi précises que celles calculées en contexte supervisé avec l'algorithme `NBSS-EM`. Cela implique que le modèle initial calculé avec l'algorithme `NBSSA` à partir de S_{pos} et S_{unl} est un très bon modèle initial car il permet à la méthode E.M. de converger vers le même modèle qu'en contexte semi-supervisé, et donc d'en atteindre les mêmes performances.

	20 attributs binaires			50 attributs binaires		
$\lvert S_{lab}\rvert$ ($\lvert S_{unl}\rvert = 10 \cdot \lvert S_{lab}\rvert$)	100	300	500	100	300	500
Algorithme 7 : `NB-UNL` S_{unl}	0.127 *±0.081*	0.088 *±0.081*	0.069 *±0.069*	0.080 *±0.053*	0.048 *±0.038*	0.032 *±0.025*
Algorithme 5 : `NBSSA` S_{pos} et S_{unl}	0.047 *±0.013*	0.023 *±0.006*	0.012 *±0.004*	0.022 *±0.009*	0.012 *±0.004*	0.007 *±0.003*
Algorithme 6 : `NBSSA-EM` S_{pos} et S_{unl}	**0.014** *±0.003*	**0.008** *±0.002*	**0.006** *±0.002*	**0.010** *±0.002*	**0.006** *±0.001*	**0.005** *±0.000*
Algorithme 3 : `NBSS-EM` S_{lab} et S_{unl}	**0.014** *±0.003*	**0.008** *±0.002*	**0.006** *±0.002*	**0.010** *±0.002*	**0.006** *±0.001*	**0.005** *±0.000*

Table 3.1. Racine carrée de l'erreur quadratique moyenne des estimations du paramètre $P(+)$ calculées par les méthodes `NBSS-EM`, `NBSSA`, `NBSSA-EM`, `NB-UNL`. Dans chacune des expériences, la taille de l'ensemble S_{pos} est approximativement égale à $P(+|\theta_c) \cdot |S_{lab}|$.

Performances pour la tâche de classification

Nous présentons ici des résultats concernant la tâche de classification (Table 3.2). Ces résultats sont issus de la même série d'expériences qui a permis d'obtenir les résultats de la table 3.1. Les modèles appris lors de ces expériences par les différents algorithmes sont maintenant utilisés pour tester les performances des classifieurs correspondants sur l'ensemble S_{test} défini dans le protocole expérimental. La table 3.2 indique les moyennes des erreurs empiriques $\hat{P}(f(x) \neq y)$ des classifieurs sur S_{test}, chacune calculée sur les 200 expériences menées par nombre d'attributs et par nombre de données d'apprentissage. Le taux d'erreur du classifieur naïf de Bayes défini par le modèle cible θ_c sur les ensembles S_{test} est également indiqué dans la table pour chaque nombre d'attributs.

Ces résultats confirment les deux conclusions des expériences précédentes. En effet, les performances des classifieurs appris avec l'algorithme `NBSSA` sont nettement meilleures que celles des classifieurs appris en contexte non supervisé avec l'algorithme `NB-UNL`. Ce résultat était prévisible car l'obtention de bonnes estimations des paramètres du classifieur en

	20 attributs binaires			50 attributs binaires		
$\lvert S_{lab}\rvert$ ($\lvert S_{unl}\rvert = 10 \cdot \lvert S_{lab}\rvert$)	100	300	500	100	300	500
Classifieurs définis par θ_c	0.049 $\pm$*0.022*			0.001 $\pm$*0.002*		
Algorithme 7 : `NB-UNL` S_{unl}	0.225 $\pm$*0.082*	0.181 $\pm$*0.086*	0.164 $\pm$*0.091*	0.113 $\pm$*0.087*	0.108 $\pm$*0.078*	0.106 $\pm$*0.075*
Algorithme 5 : `NBSSA` S_{pos} et S_{unl}	0.106 $\pm$*0.036*	0.068 $\pm$*0.024*	0.057 $\pm$*0.022*	0.015 $\pm$*0.021*	0.004 $\pm$*0.005*	0.002 $\pm$*0.003*
Algorithme 6 : `NBSSA-EM` S_{pos} et S_{unl}	**0.051** $\pm$*0.022*	**0.050** $\pm$*0.022*	**0.049** $\pm$*0.022*	**0.002** $\pm$*0.002*	**0.001** $\pm$*0.002*	**0.001** $\pm$*0.002*
Algorithme 3 : `NBSS-EM` S_{lab} et S_{unl}	**0.051** $\pm$*0.022*	**0.049** $\pm$*0.022*	**0.049** $\pm$*0.022*	**0.002** $\pm$*0.002*	**0.001** $\pm$*0.001*	**0.001** $\pm$*0.001*

Table 3.2. Moyenne et écart-type de l'erreur empirique des classifieurs appris par les algorithmes `NBSS-EM`, `NBSSA`, `NBSSA-EM` et `NB-UNL`, sur les ensembles S_{test}.

contexte non supervisé nécessite un très grand nombre de données. La deuxième conclusion est également confirmée, les deux algorithmes `NBSS-EM` et `NBSSA-EM` convergent vers le même modèle et obtiennent donc des performances similaires sur la tâche de classification.

Discussion

Plusieurs séries d'expériences supplémentaires ont été menées avec ces algorithmes :

- d'autres nombres m d'attributs binaires ont été testés. Les résultats obtenus suivent la même tendance que ceux exposés dans cette section. Les conclusions restent identiques quel que soit le nombre d'attributs,

- d'autres tailles des ensembles S_{lab} (et donc S_{unl} et S_{pos}) ont également été testées. Pour un nombre de données supérieur à ceux utilisés dans les expériences présentées dans cette section, aucun changement significatif n'est à signaler car tous les algorithmes ont pratiquement convergé dans ces expériences, à l'exception de l'algorithme `NB-UNL` qui nécessite un très grand nombre de données pour converger vers le modèle cible (de l'ordre de plusieurs milliers d'exemples).

Les classifieurs naïfs de Bayes sont donc efficacement apprenables en contexte asymétrique : d'une part, l'absence d'exemples négatifs ne pénalise pas les performances des classifieurs appris en comparaison de ceux appris en contexte semi-supervisé, et, d'autre part, la présence des exemples positifs permet d'apprendre plus efficacement les classifieurs naïfs de Bayes qu'en leur absence.

De plus, les algorithmes `NBSSA` et `NBSSA-EM` sont des méthodes simples, consistantes, faciles à implémenter et peu coûteuses en temps de calcul, ce qui les rend facilement utilisables en pratique. Enfin, contrairement aux méthodes proposées pour apprendre ces classifieurs en contexte asymétrique (Section 2.2.4), elles ne reposent sur aucune heuristique ou hypothèse supplémentaire.

3.3 Expériences sur les données ponts disulfures

Cette section présente une série d'expériences menée sur des données réelles, avec pour objectif de tester l'hypothèse posée en section 3.1 sur le statut (la classe) des paires de cystéines observées non appariées. Dans cette section, nous présentions quelques considérations biologiques qui laissent penser que choisir l'environnement local des cystéines comme information principale pour prédire les ponts disulfures, ce qui est le choix de la plupart des travaux sur ce problème, pourrait rendre ambiguë la classe des paires de cystéines non appariées. Si on considère ces exemples comme négatifs, on peut alors appliquer un algorithme de classification supervisée, mais si on considère qu'on ne peut pas décider quelle est la classe de ces exemples, on doit alors appliquer un algorithme de classification semi-supervisée asymétrique (Section 3.2). Notre hypothèse est que le deuxième choix est plus cohérent et améliorerait la prédiction des ponts disulfures.

Les résultats obtenus en section 3.2 montrent que l'apprentissage semi-supervisé asymétrique est un problème mal posé dans le cadre de l'apprentissage statistique si on ne fait pas d'hypothèse sur les distributions sous-jacentes au problème. Toutefois, ils montrent également que l'on peut apprendre efficacement les classifieurs naïfs de Bayes dans ce contexte. Ce classifieur peut donc servir à tester notre hypothèse, même si on ne peut pas objectivement attendre qu'il permette d'obtenir d'aussi bonnes performances sur la prédiction des ponts disulfures que des méthodes plus sophistiquées et plus expressives. L'algorithme `NB` (Algorithme 1) est donc choisi pour prédire les ponts en contexte supervisé, et l'algorithme `NBSSA-EM` (Algorithme 6) pour les prédire en contexte semi-supervisé. Les expériences menées à la section 3.2.4 montrent que cet algorithme est plus performant que l'algorithme `NBSSA` (Algorithme 5) seul.

3.3.1 Protocole expérimental

Les résultats présentés dans la table 3.4 ont été obtenus en suivant le protocole expérimental décrit dans cette section. Ce protocole est très proche de ceux utilisés dans les travaux décrits en section 1.2.1 sur la prédiction des ponts disulfures dans les protéines. Comme indiqué en introduction de ce chapitre, nous ne remettons pas en cause les choix faits par ces auteurs concernant la majeure partie du protocole expérimental suivi. Seule la question du statut des paires de cystéines non appariées est étudiée ici.

Séparation des protéines par nombre de ponts

Les expériences ont été menées sur l'ensemble de protéines annotées `G3D-SS`, décrit en annexe A. Lors de ces travaux, seules ces données étaient disponibles, l'ensemble `SPX` ayant été rendu disponible plus d'un an après la publication des travaux présentés dans ce chapitre dans [Magnan, 2005] et quelques mois après la publication d'une version longue de ces travaux dans [Magnan, 2006].

Les protéines de cet ensemble sont séparées selon le nombre n de ponts disulfures qu'elles contiennent, pour n allant de 2 à 5 (Table A.2). Le cas des protéines à $n = 1$ pont est trivial, et pour $n > 5$, trop peu de données sont disponibles dans l'ensemble `G3D-SS` pour qu'elles permettent d'obtenir des résultats significatifs. Chaque ensemble de protéines ainsi obtenu constitue la base de données d'un problème d'apprentissage indépendant des autres, ce qui implique que quatre problèmes indépendants sont considérés ici.

Extraction et modélisation des données d'apprentissage

Une protéine qui contient un nombre n de ponts disulfures ($2n$ cystéines oxydées) permet d'observer $n(2n-1)$ paires de cystéines potentiellement appariées. Les n paires appariées forment donc les exemples positifs d'apprentissage, et les $n(2n-2)$ paires restantes les exemples négatifs dans un premier temps puis les exemples non étiquetés d'apprentissage dans un second temps. La figure 3.1 (Section 3.1.2 p.75) propose une illustration de cela pour une protéine à 3 ponts (15 paires de cystéines distinctes).

Seuls les acides aminés voisins des cystéines oxydées sur la séquence primaire de la protéine (environnements locaux) sont utilisés comme information pour prédire si une paire de ces cystéines forme un pont disulfure ou non. Aucune autre information n'est utilisée car l'hypothèse testée ici repose sur le choix de cette seule information. Cela assure ainsi que les résultats obtenus sont pertinents pour l'évaluation de cette hypothèse (Section 3.1).

Pour chaque cystéine oxydée C d'une même protéine p, on extrait donc un fragment $W = a_{-r}, a_{-r+1}, ..., a_{-1}, C, a_1, ..., a_r$ de la séquence primaire de p centré sur C, appelé également *fenêtre*. Ce fragment représente l'*environnement local* de C et le nombre d'acides aminés situés de chaque côté de C est appelé le *rayon* (noté r) de la fenêtre.

L'alphabet naturel Σ^{nat} des acides aminés contient 20 éléments, *i.e.* $\forall i \in \{-r, ..., r\}$, $a_i \in \Sigma^{nat}$. Toutefois, certains acides aminés des séquences protéiques ne sont pas identifiés, ils sont généralement notés par le caractère $'X'$, et la définition des fenêtres donnée plus haut pose problème lorsqu'une cystéine est proche du bout de la chaîne protéique. Un caractère supplémentaire est donc ajouté à l'alphabet Σ^{nat} pour signaler une de ces deux situations. On utilise donc l'alphabet $\Sigma^{nat+} = \Sigma^{nat} \cup 'X'$ et les segments extraits peuvent alors être considérés comme des mots de longueur $2r+1$ sur cet alphabet.

Une paire de cystéines oxydées (C, C') est donc modélisée par la paire (W, W') d'environnements locaux respectifs de ces deux cystéines. Pour faire ressortir la notion d'affinité entre deux segments d'une protéine discutée en section 3.1.1, trois codages de ces paires d'environnements locaux sont testés. Chacun de ces codages est basé sur l'idée que l'affinité entre deux segments est en réalité une affinité entre les acides aminés de ces segments. En notant $W = a_{-r}, ..., a_{-1}, a_0, a_1, ..., a_r$ et $W' = a'_{-r}, ..., a'_{-1}, a'_0, a'_1, ..., a'_r$ les fenêtres de rayon r centrées respectivement sur deux cystéines C et C' ($a_0 = C$ et $a'_0 = C'$) d'une même protéine, les trois codages proposés sont :

- Codage simple : $(W, W') = \{(a_i, a'_i)\}$, $i \in \{-r, ..., r\}$, $i \neq 0$
- Codage double : $(W, W') = \{(a_i, a'_i)\} \cup \{(a_i, a'_{-i})\}$, $i \in \{-r, ..., r\}$, $i \neq 0$
- Codage croisé : $(W, W') = \{(a_i, a'_j)\}$, $i, j \in \{-r, ..., r\}$

Ces codages modélisent tous une paire de fenêtres (W, W') comme un sac de mots où chaque mot est une paire d'acides aminés formée avec un résidu de W et un résidu de W'. Les paires d'acides aminés sont des éléments de l'alphabet $\Sigma = \Sigma^{nat+} \times \Sigma^{nat+}$. Il contient 231 lettres, le nombre de couples ordonnés sur un alphabet à 21 lettres.

Chaque paire de cystéines oxydées d'une protéine p est donc modélisée par un vecteur de taille 231 où chaque coordonnée indique le nombre de fois où la paire de résidus correspondante apparaît dans l'environnement local de ces cystéines.

Description d'une expérience

Soit un nombre n de ponts disulfures par protéine et soit $\mathcal{P}_n$ l'ensemble des protéines contenant n ponts. Les ensembles $\mathcal{P}_n$ dont nous disposons étant de tailles réduites (indiquées dans la table A.2), la méthode de validation choisie est la validation croisée 10-folds. Une étude de la pertinence de cette méthode de validation peut être trouvée, par exemple, dans [Kohavi, 1995] ou encore dans [Dietterich, 1998].

Une expérience consiste donc à diviser aléatoirement l'ensemble des protéines de $\mathcal{P}_n$ en dix ensembles de tailles similaires et à répéter dix fois le procédé qui consiste à apprendre un classifieur à partir des données extraites des protéines de 9 de ces 10 ensembles et à tester le classifieur sur l'ensemble de protéines restant, en changeant l'ensemble de protéines test à chaque itération. Chaque expérience permet donc d'obtenir des performances calculées sur la totalité des protéines de l'ensemble $\mathcal{P}_n$.

Protocoles de test des modèles calculés

Lorsqu'un classifieur a été calculé sur les données d'apprentissage extraites de l'ensemble de protéines correspondant, l'évaluation des performances de ce classifieur nécessite un protocole de test adapté au cas particulier des ponts disulfures. En effet, le classifieur permet de classer toutes les paires individuellement mais ne permet pas de prendre en compte la contrainte d'unicité d'appariement d'une cystéine.

Les algorithmes `NB` et `NBSSA-EM` fournissent tous les deux un modèle probabiliste qui peut être utilisé pour les calculs du classifieur naïf de Bayes mais qui permet également de calculer pour chaque paire d'environnements locaux la probabilité que cette paire forme un pont. Cette caractéristique permet de construire des méthodes de test adaptées à la situation, c'est-à-dire décidant pour une protéine d'une configuration complète des ponts de cette protéine.

Pour une protéine test, une première méthode naïve pour prédire la configuration des ponts de cette protéine pourrait consister à répéter jusqu'au classement de toutes les paires de cystéines : prédire un pont entre la paire de cystéines qui maximise la probabilité que cette paire forme un pont dans le modèle appris et classer négativement toutes les paires ayant une cystéine en commun avec cette paire prédite appariée.

Toutefois, cette méthode n'est pas optimale puisqu'elle ne permet pas d'obtenir la configuration la plus vraisemblable. Cette configuration optimale peut être obtenue par un simple algorithme de couplage parfait de poids maximal dans un graphe complet, où les sommets du graphe sont les cystéines oxydées d'une protéine et le poids des arêtes du graphe la probabilité que deux cystéines forment un pont dans le modèle appris. Ce protocole de test, utilisé dans nos expériences, est décrit dans la section 1.2.1, car il est également utilisé par les autres auteurs de travaux sur la prédiction des ponts disulfures.

3.3.2 Choix aléatoire des ponts

Le protocole décrit dans la section précédente permet d'évaluer les modèles appris par les algorithmes sur leur capacité à prédire correctement les ponts d'une protéine en assurant qu'exactement n ponts sont prédits avec ces modèles pour une protéine qui en contient n. Le critère d'évaluation est donc la proportion de ponts correctement retrouvés, correspondant au rappel positif.

Dans ce cas, il est nécessaire de comparer les résultats obtenus avec une méthode de sélection des ponts aléatoire pour s'assurer qu'il y a bien eu apprentissage, c'est-à-dire que les modèles qui ont été calculés par les algorithmes contiennent bien une information qui permette de distinguer les paires de cystéines appariées des paires non appariées.

La proportion de ponts correctement prédits par une méthode de sélection quelconque correspond à la probabilité pour tout pont d'être correctement prédit par cette méthode. Or, si on tire aléatoirement une configuration de ponts pour une protéine qui en contient n, il est possible de calculer la probabilité pour un pont donné d'être dans cette configuration tirée au hasard. Par définition, elle vaut le nombre de configurations dans lesquelles ce pont apparaît, divisé par le nombre total de configurations envisageables.

Le nombre total t_n de configurations envisageables pour n ponts est donné par la formule $t_n = \prod_{i=1}^{n} 2i - 1$. Cette formule se démontre très simplement par récurrence sur le nombre de ponts. De plus, le nombre de configurations dans lesquelles se trouve un pont donné est simplement égal au nombre de configurations des $n - 1$ ponts restants, soit t_{n-1}. La probabilité de prédire correctement un pont par un tirage aléatoire d'une configuration de n ponts est donc t_{n-1}/t_n, soit $1/(2n - 1)$.

Par exemple, pour des protéines contenant 2 ponts, la probabilité pour tout pont d'être correctement prédit vaut 1/3. La table 3.3 indique pour chaque nombre n de ponts par protéine, l'espérance du pourcentage de ponts correctement retrouvés par un choix aléatoire des ponts d'une protéine avec n ponts. Ces performances d'un choix aléatoire des ponts sont mises en évidence dans les travaux existants sur la prédiction des ponts disulfures. Elles donnent en quelque sorte, pour chaque nombre de ponts, le seuil minimal de performance pour le problème de la prédiction des ponts disulfures dans les protéines sachant l'état d'oxydation des cystéines.

Nombre de ponts par protéine	Performances attendues par un choix aléatoire des ponts
2	33.33%
3	20%
4	14.29%
5	11.11%

Table 3.3. Espérance du pourcentage de ponts correctement retrouvés lors d'un choix aléatoire d'une configuration des ponts disulfures.

3.3.3 Résultats expérimentaux

La table 3.4 présente les performances obtenues par les algorithmes `NB` et `NBSSA-EM` sur les protéines contenant de 2 à 5 ponts. Chaque résultat dans cette table est une moyenne calculée sur une série de 100 expériences menées suivant le protocole décrit en section 3.3.1 (100 cross-validations 10-folds complètes).

Les codages simple et double n'ont pas permis d'obtenir des performances meilleures que celles obtenues par un choix aléatoire des ponts (Table 3.3), c'est la raison pour laquelle ces résultats ne sont pas présentés ici. Seuls les résultats obtenus avec le codage croisé ont permis d'obtenir des résultats différents d'un choix aléatoire des ponts, nous les donnons dans cette section. De façon générale, toutes les expériences menées durant cette thèse sur les données biologiques ont montré que les codages simple et double ne permettaient pas d'obtenir des résultats satisfaisants. Il semble que ces codages soient trop peu informatifs pour la tâche considérée. De plus, plusieurs rayons r de fenêtres ont été expérimentés et, comme différents travaux portant sur ce problème [Fariselli and Casadio, 2001, Vullo and Frasconi, 2004, Ferrè and Clote, 2005b], les rayons r=5 et r=6 ont permis d'obtenir les meilleurs résultats. La table 3.4 présente les résultats obtenus pour r=6, légèrement meilleurs que ceux obtenus avec r=5.

Les résultats montrent que lorsque les paires de cystéines non appariées sont considérées comme des exemples n'apportant pas d'information sur la classe, l'algorithme naïf de Bayes obtient de meilleures performances en moyenne sur la prédiction des ponts que lorsque ces exemples sont considérés comme négatifs (performances de l'ordre de l'aléa). Ces performances sont tout de même bien inférieures à celles obtenues par les différents travaux sur ce problème (Tables 1.1 et 1.2 pages 37 et 38) qui considèrent plus d'informations pour prédire les ponts, des méthodes plus sophistiquées et des ensembles de protéines plus conséquents (non rendus publics lors de la réalisation de ces travaux).

L'hypothèse discutée dans ce chapitre semble donc, à première vue, vérifiée. Toutefois, il faut noter que la différence entre les performances obtenues en contexte semi-supervisé asymétrique (Algorithme `NBSSA-EM`) et en contexte supervisé (Algorithme `NB`) n'est pas validée par un test t apparié de Student à 95% de confiance, même pour les résultats sur les protéines à 2 et 3 ponts. Il n'est donc pas possible de certifier que cette différence est réellement significative même s'il semble bien y avoir une information détectée dans le cas asymétrique qui ne l'est pas dans le cas supervisé classique.

Nombre de ponts par protéine	Choix aléatoire des ponts	Performances avec `NB` (supervisé)	Performances avec `NBSSA-EM` (semi-sup. asymétrique)
2	33.3%	40.2%	**58.8%**
3	20%	17.5%	**33.4%**
4	14.3%	12.7%	**16.3%**
5	11.1%	5.8%	**13.2%**

Table 3.4. Moyennes des pourcentages de ponts correctement prédits par les algorithmes `NB` et `NBSSA-EM` pour les protéines contenant de 2 à 5 ponts et un codage croisé des attributs.

Une étude plus approfondie du déroulement de ces expériences a montré qu'une partie des protéines était toujours correctement ou incorrectement prédite lors des différentes validations croisées tandis que l'autre partie des protéines obtenait des résultats très variables. La même série d'expériences sans les protéines fréquemment mal classées permet d'obtenir de meilleures performances que celles indiquées dans la table 3.4, de l'ordre de celles obtenues dans les travaux de [Fariselli et al., 2002] (Table 1.1). Il semble donc qu'un lot de protéines suive un modèle similaire de formation des ponts, mais que les autres protéines aient une vraie différence avec ce premier lot. Les modèles appris semblent donc se spécifier sur ce lot, et ne pas avoir de réelles capacités de généralisation.

Les résultats issus de ces expériences sont tout de même d'une part encourageants, car l'hypothèse posée au début de cette étude semble être valide, mais d'autre part ils ne permettent pas de prouver que la différence observée est significative au sens d'un test statistique fiable. Or, les performances du classifieur appris en contexte supervisé sont très proches de celles obtenues par un choix aléatoire des ponts. Il est donc impossible d'affirmer avec certitude à partir de ces résultats que la seule information utilisée dans ces expériences pour prédire les ponts, les environnements locaux des cystéines oxydées, est une information pertinente pour cette tâche de prédiction.

La question de savoir si les résidus voisins d'une cystéine sont impliqués dans la formation des ponts, pour laquelle les biologistes du domaine eux-mêmes sont sceptiques [Gibrat, 2005, Geourjon, 2005], reste donc ouverte même si les résultats obtenus dans l'étude menée dans ce chapitre sont encourageants et semblent aller dans ce sens. Ces expériences montrent également la nécessité d'étudier plus formellement cette question de l'existence d'une information locale impliquée dans la formation des ponts pour espérer pouvoir y répondre.

Ces résultats ont néanmoins fait ressortir une information importante : ne pas avoir imposé que les paires de cystéines non appariées soient des paires ne pouvant pas former de pont disulfure semble malgré tout permettre d'extraire une information qui n'est pas extraite dans le cas supervisé. Ce constat, ainsi que l'étude menée sur l'apprentissage en contexte semi-supervisé asymétrique, ont reçu des retours positifs de la part de la communauté scientifique du domaine, qui ont autorisé leur publication dans les actes de la conférence CAp 2005 [Magnan, 2005], ainsi que dans la revue d'intelligence artificielle RIA [Magnan, 2006]. Cette étude a également motivé toute la suite des travaux de la thèse dans laquelle cette première hypothèse étudiée a été retravaillée pour la mise en place d'un protocole pour détecter la présence d'une information locale impliquée dans la formation de contacts locaux tels que les ponts disulfures.

Chapitre 4

Un protocole générique pour détecter une information locale

Sommaire

Ce chapitre présente une étude formelle de la notion d'affinité entre paires d'environnements locaux de résidus potentiellement appariés dans une protéine. Une modélisation de cette notion est proposée, basée en grande partie sur une évolution de l'hypothèse testée dans le chapitre précédent du mémoire. Les expériences menées dans ces précédents travaux n'ont pas permis de prouver que les environnements locaux des résidus appariés par un pont disulfure jouent un rôle dans la formation de ces ponts, et donc qu'il s'agit d'une information pertinente pour les prédire. Toutefois, ils semblent aller dans ce sens et encouragent donc à retravailler l'idée proposée sur la formalisation des données sachant l'information utilisée pour prédire les ponts : les résidus voisins des cystéines oxydées.

Cette question de l'existence d'une telle information impliquée dans la formation de ponts est également soulevée par certains biologistes [Gibrat, 2005, Geourjon, 2005] qui soutiennent qu'il n'est pas du tout évident que la formation de ponts tels que les ponts disulfures soit influencée par les résidus voisins des acides aminés concernés. Les travaux de la littérature ne permettent pas de répondre à cette question et les résultats obtenus dans le chapitre précédent ne permettent pas, même s'ils vont dans ce sens, de le prouver.

Cette question est pourtant essentielle pour la prédiction de contacts. En effet, c'est une information qui a montré toute son importance pour le problème de la prédiction de brins β, dont l'implication dans la formation de ponts disulfures n'a pas été prouvée mais qui est systématiquement utilisée dans les travaux sur la prédiction de ces ponts, et il y a de bonnes raisons de penser que c'est une information qui sera encore utilisée pour

la prédiction d'autres contacts ponctuels (Section 1.2). Il est donc important de tenter de modéliser de façon formelle cette notion d'information locale, c'est-à-dire la notion d'affinité entre paires d'environnements locaux, pour tenter de l'extraire et de pouvoir en prouver l'existence.

De plus, comme indiqué à la section 1.2, plusieurs contacts ponctuels entre deux acides aminés distants sur la séquence d'une protéine en contraignent ou stabilisent la structure. Il est donc intéressant de modéliser cette notion d'affinité locale indépendamment du type de contact considéré. C'est la raison pour laquelle nous ne considérons plus uniquement le cas des ponts disulfures mais tout contact ponctuel entre deux résidus distants d'une protéine.

Que ce soit d'un point de vue biologique ou informatique, il est intéressant de savoir mesurer l'impact des environnements locaux de résidus dans la formation de contacts entre ces résidus. Si cette information entre en jeu dans la formation de ponts, il faut trouver comment l'extraire et l'intégrer dans les travaux de prédiction des contacts, dans le cas contraire, il ne faut pas l'intégrer car elle introduit une information erronée. Notre approche se limite donc à étudier si l'information portée par les environnements locaux de résidus appariés est pertinente pour la prédiction de ces contacts. Pour lever toute ambiguïté, il est très clair que cette information, si elle existe, ne détermine certainement pas à elle seule les contacts, quel que soit le type de contact.

Nous proposons donc dans ce chapitre une modélisation de ce que pourrait être une fonction d'affinité entre paires d'environnements locaux de résidus potentiellement en interaction, et des conséquences d'une telle fonction sur les données observées : les paires appariées et non appariées. La modélisation proposée considère une fonction d'affinité à deux niveaux : un haut niveau d'affinité, signifiant que deux environnements locaux ont une forte propension à se retrouver appariés par leurs résidus centraux et un faible niveau d'affinité signifiant qu'une telle paire a une faible propension à se retrouver appariée. Le modèle proposé n'interdit pas la formation d'un contact entre paires à faible niveau d'affinité et, tout comme nous le proposions dans les travaux présentés dans le chapitre précédent, une paire avec un haut niveau d'affinité peut très bien être observée comme ne formant pas un pont pour un ensemble de raisons locales et globales.

On montre alors que si une telle information existe entre deux paires d'environnements locaux, alors les paires observées appariées et non appariées, au travers des protéines dont la structure est connue, n'en fournissent qu'une observation indirecte. En effet, les paires observées appariées et non appariées seraient donc des paires classées par cette fonction d'affinité mais corrompues par du bruit de classification avant d'être observées comme étant ou non en contact.

Le modèle de bruit induit par une telle formalisation est une généralisation du bruit de classification uniforme (Section 2.1.3), noté CN, puisqu'il suppose que les exemples de chaque classe (haut et bas niveau d'affinité) sont corrompus par un bruit uniforme constant mais dont le taux d'inversion de la classe de chaque exemple dépend de la classe d'appartenance de l'exemple. Nous appelons ce bruit *bruit de classification conditionnel à chaque classe*, et le notons CCCN.

Si notre hypothèse est correcte et si la fonction d'affinité peut être représentée dans

une classe de fonctions calculables à partir de données corrompues par un tel modèle de bruit, alors un algorithme qui permet ce calcul doit être capable d'apprendre la fonction d'affinité à partir des exemples dont on dispose. On doit alors être capable de vérifier que les environnements locaux des résidus appariés sont impliqués dans la formation de ponts en vérifiant que les paires à haut niveau d'affinité ont plus de chances d'être appariées que les paires à faible niveau d'affinité.

Il découle immédiatement de cette étude un protocole pour détecter, extraire et évaluer une telle information si elle existe puisque, si on est capable d'apprendre une fonction d'affinité qui possède ces caractéristiques, alors on aura bien prouvé qu'elle existe et influence la formation de ponts. Ce protocole est utilisable en pratique malgré le faible nombre de données disponibles dans les bases de données et il est indépendant du contact, il permet donc d'être utilisé pour tout type de contact ponctuel entre deux résidus.

4.1 Modélisation des données

Cette section présente une modélisation du problème de la prédiction de contacts ponctuels entre deux résidus distants d'une protéine (ponts disulfures, salins, etc) et des environnements locaux de tels résidus potentiellement en interaction.

La structure primaire d'une protéine peut être considérée comme un mot de Σ^* où Σ est l'ensemble des 20 acides aminés ou un autre alphabet similaire. Soit $\mathcal{P} \subset \Sigma^*$ l'ensemble des protéines avec un nombre pair de résidus impliqués dans un contact ponctuel (les cystéines oxydées pour les ponts disulfures par exemple) et soit $\mathcal{P}_n \subset \mathcal{P}$ l'ensemble des protéines avec $2n$ résidus appariés. On a donc $\mathcal{P} = \bigcup_n \mathcal{P}_n$.

Soit $\mathcal{G}$ l'ensemble des graphes non-orientés pour lesquels chaque sommet est de degré 1 ($\mathcal{G}$ est l'ensemble des couplages parfaits) et soit $\phi : \mathcal{P} \to \mathcal{G}$ une fonction qui associe un graphe de $\mathcal{G}$ à une protéine de $\mathcal{P}$ de telle façon que chaque sommet du graphe soit un des résidus impliqués dans un contact et un arc entre deux sommets du graphe un pont entre les deux résidus concernés. L'objectif d'un problème de prédiction de contacts entre deux résidus distants d'une protéine est donc d'approximer avec la meilleure précision possible la fonction ϕ à partir d'exemples issus d'expériences.

L'information située autour des résidus appariés est peut-être impliquée dans ces appariements. En général, des segments centrés sur les résidus appariés de taille $2r+1$ sont considérés, où r est appelé le rayon des segments. Soit P une distribution de probabilité sur $\mathcal{P}$ et soit $\Omega_r = \Sigma^{2r+1}$ l'ensemble des segments de protéines de taille $2r+1$. Les éléments de Ω_r sont les environnements locaux de résidus appariés.

Sur l'exemple donné à la figure 4.1, les environnements locaux des cystéines sont les mots $w_4 = XXCCL$, $w_5 = XCCLY$, $w_{10} = GKCRR$, $w_{16} = XGCSS$, $w_{21} = ASCCQ$ et $w_{22} = SCCQR$ avec $r = 2$.

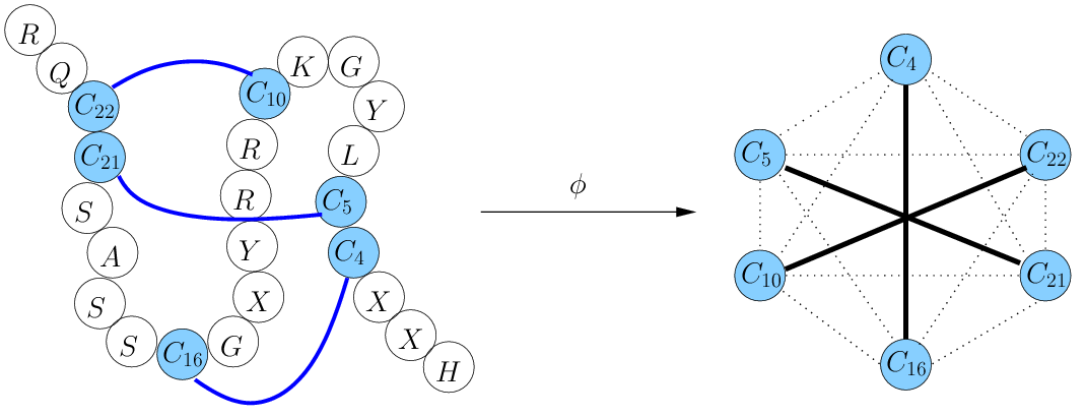


Figure 4.1. Une séquence d'acides aminés (identifiant PDB `1AS5`) avec trois ponts disulfures. La prédiction de ces ponts est déterminée par la connaissance de la fonction ϕ qui calcule la configuration correcte des ponts d'une protéine.

On dira que deux environnements locaux w et w' d'une même protéine p sont distincts si les positions des résidus centraux de ces segments sur la séquence primaire de p sont distinctes. Pour $w, w' \in \Omega_r$, on note :

- $P(w)$ la probabilité que w soit l'environnement local d'un résidu impliqué dans un pont pour une protéine $p \in \mathcal{P}$,
- $P(w, w')$ la probabilité que w et w' soient des environnements locaux distincts de résidus potentiellement appariés dans une protéine $p \in \mathcal{P}$,
- $P(w, w'|n)$ la probabilité que w et w' soient des environnements locaux distincts de résidus potentiellement appariés dans une protéine $p \in \mathcal{P}_n$,
- $P(B(w, w')|w, w', n)$ la probabilité que w et w' soient appariés par leurs résidus centraux sachant qu'ils en sont les environnements locaux dans une protéine $p \in \mathcal{P}_n$.

4.2 Un modèle de détection de l'information locale

Nous considérons la question suivante : certains couples (w, w') d'environnements locaux ont-ils plus de chances que d'autres d'être liés par un pont ? Autrement dit, existe-t-il une information portée par les environnements locaux de résidus potentiellement appariés qui joue un rôle dans le fait que les résidus soient ou non en contact ?

4.2.1 Une première approche de l'information locale

Une première approche de l'existence d'une information locale impliquée dans le fait qu'il y ait ou non un pont entre deux résidus potentiellement appariés peut se décrire ainsi :

- considérons des protéines contenant n ponts ; s'il n'y a pas d'information locale qui contribue à la formation des ponts, alors $P(B(w, w')|w, w', n) = \frac{1}{2n-1}\ \forall (w, w')$ (car il y a $n(2n-1)$ paires de résidus potentiellement en interaction et n ponts formés),
- réciproquement, si $\forall (w, w')$, $P(B(w, w')|w, w', n) = \frac{1}{2n-1}$, alors il n'y a pas d'information locale car le résultat ne dépend pas de w et w'.

Cette approche fournit un méthode statistique simple pour décider de l'existence d'une information locale impliquée dans la formation de ponts. Le problème de cette méthode est qu'il est impossible d'estimer $P(B(w, w')|w, w', n)$ sans hypothèse supplémentaire. Considérons des environnements de petite taille : 7 ($r = 3$). Alors $|\{(w, w'), w, w' \in \Omega_r\}| = 20^{12} \simeq 4.10^{14}$, et seules quelques centaines d'exemples sont disponibles dans les bases de données.

Il est donc nécessaire d'émettre une ou plusieurs hypothèses qui permettent de simplifier cette approche pour que le faible nombre de données dont on dispose par les bases de données de structure de protéines déterminées expérimentalement ne rende pas impossible son application. Cette simplification doit donc rendre non nécessaire la connaissance des probabilités $P(B(w, w')|w, w', n)$ tout en conservant la plus grande partie de l'information portée par ces probabilités : la propension de deux environnements locaux à être appariés.

4.2.2 Une approche simplificatrice et raisonnable

La solution que nous proposons pour appliquer une approche similaire à celle présentée à la section précédente est de supposer l'existence d'une fonction d'affinité discrète g censée représenter la probabilité que deux environnements locaux soient liés par un pont : $g : \Omega_r \text{ x } \Omega_r \rightarrow Y$, avec Y ordonné et $|Y|$ petit, telle que :

- $g(w_1, w_2) = g(w'_1, w'_2) \Rightarrow P(B(w_1, w_2)|w_1, w_2, n) \simeq P(B(w'_1, w'_2)|w'_1, w'_2, n)$
- $y < y' \Rightarrow P(B(w_1, w_2)|g(w_1, w_2) = y) < P(B(w'_1, w'_2)|g(w'_1, w'_2) = y')$ $(y, y' \in Y)$

On a alors $P(B(w, w')|w, w', n) \simeq P(B(w, w')|g(w, w'), n)$. En considérant le cas le plus simple envisageable : $Y = \{-1, 1\}$ (Figure 4.2) : les paires de fenêtres sont alors partitionnées en deux classes correspondant à deux niveaux d'affinité (-1 signifiant un bas niveau d'affinité et 1 un haut niveau d'affinité). Dans ce cas :

$$P(B(w, w')|w, w', n) \simeq P(B(w, w')|g(w, w'), n) = \begin{cases} \alpha_1^n \ si \ g(w, w') = 1 \\ \alpha_0^n \ si \ g(w, w') = -1 \end{cases}$$

En supposant qu'une telle fonction d'affinité g existe et joue un rôle dans l'appariement des résidus, on doit avoir α_1^n significativement plus grand que α_0^n. En d'autres termes, il doit y avoir plus de paires de résidus appariés quand les environnements de ces résidus ont une forte affinité l'un pour l'autre ($g(w, w') = 1$) que quand ils ont un faible niveau d'affinité ($g(w, w') = -1$).

Dans ce modèle, on considère que chaque paire d'environnements locaux de résidus potentiellement en contact n'appartient qu'à une seule classe attribuée par cette fonction g. Les observations de paires dont on dispose au travers des protéines dont la structure est connue n'en fourniraient alors qu'une observation indirecte : certaines paires classées -1 par g peuvent être observées appariées et à l'inverse, une paire avec un haut niveau d'affinité peut ne pas être observée appariée dans une protéine dont on connaît la structure. Cette situation est schématisée en figure 4.3.

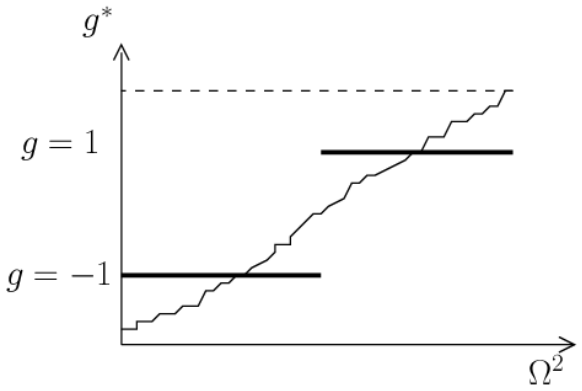


Figure 4.2. Représentation d'une fonction d'affinité g à deux niveaux en fonction de la valeur de $g^* = P(B(w, w')|w, w', n)$. Les paires (w, w') d'environnements locaux sont ordonnées en abscisse selon la valeur de g^*.

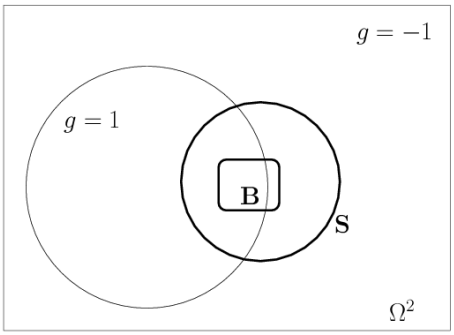


Figure 4.3. Schéma de la répartition des paires d'environnements locaux de Ω^2 pour une fonction g à deux valeurs. S représente les paires potentiellement appariées d'un ensemble de protéines $P \in \mathcal{P}_n$ et B forme l'ensemble des paires effectivement appariées.

Cette modélisation reprend l'idée exposée dans le chapitre précédent tout en la retravaillant. En effet, nous indiquions que certaines considérations biologiques, liées au choix de l'environnement local des résidus comme information pour prédire si ces résidus sont appariés ou non, rendaient ambiguë la classe des paires de résidus observés comme non appariés. Les expériences menées dans ce chapitre montraient qu'il semblait que ne pas imposer de classe négative aux paires de résidus non appariés permettait d'apprendre des classifieurs plus performants sur la prédiction des ponts. Dans le modèle présenté ici, cette idée est également présente et généralisée à toutes les paires de résidus observées.

L'expression donnée précédemment des probabilités $P(B(w, w')|g(w, w'), n)$ permet alors d'exprimer les paires d'environnements locaux observées comme des exemples corrompus par du bruit de classification dont on peut exprimer les taux de bruit en fonction de α_1^n et α_0^n : l'observation d'un pont B entre deux environnements locaux correspond donc à l'observation de $g = 1$ avec un bruit de classification $\eta^+ = 1 - \alpha_1^n$ et l'observation d'une paire non appariée correspond donc à l'observation de $g = -1$ avec un bruit de classification $\eta^- = \alpha_0^n$. La figure 4.4 donne un schéma de cette situation pour l'exemple de la figure 4.3.

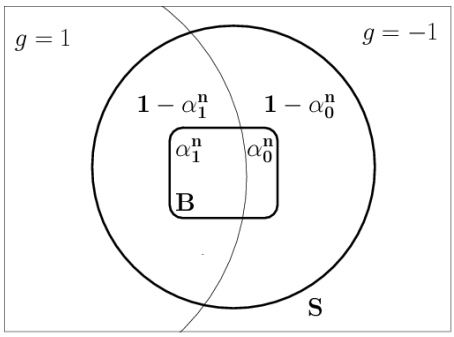


Figure 4.4. Vue détaillée de l'ensemble S de la figure 4.3. Les protéines dont sont extraites les paires de segments de S appartiennent à $\mathcal{P}_n$ et $\alpha_1^n = P(B(w, w') \mid g(w; w')=1, n)$ et $\alpha_0^n = P(B(w, w') \mid g(w, w') = -1, n)$

C'est une situation qui n'est pas inhabituelle en apprentissage automatique à la différence du domaine de la biologie pour lequel le bruit généralement considéré est un bruit sur les descriptions, appelé *malicious noise* [Kearns and Li, 1988]. La présence de bruit de classification est régulièrement observée dans des problèmes réels de classification. Ce qui est plus inhabituel, c'est le modèle de bruit considéré. Ce modèle de bruit est une généralisation du bruit de classification uniforme (Section 2.1.3), noté CN, puisqu'il considère le même modèle de bruit sur chacune des classes sans imposer que les taux de bruit qui affectent les données soient identiques. Nous l'appelons *bruit de classification conditionnel à chacune des classes*, noté CCCN.

Le problème de savoir s'il existe une information portée par l'environnement local de résidus favorisant leur appariement peut donc se ramener à trouver une telle fonction d'affinité g à partir de données corrompues par du bruit de classification de type CCCN.

La présence d'information locale doit théoriquement pouvoir être détectée dans ce modèle sous réserve que la fonction d'affinité locale puisse être représentée par une fonction apprenable à partir de données sujettes à du bruit de classification CCCN.

Dans ce cas, la fonction d'affinité doit pouvoir être approchée de manière arbitrairement proche, en supposant que l'on dispose de suffisamment d'exemples, et ce, même en l'absence d'information sur la fonction ϕ calculant le graphe de connexion d'une protéine.

L'étude présentée ici est novatrice sur la question de l'existence d'affinités locales impliquées dans la formation de contacts ponctuels entre résidus distants. Les biologistes du groupe de travail de l'ACI Genoto3D, avec qui nous travaillons, ont approuvé cette approche qui prend en considération les caractéristiques d'un tel problème biologique et la formalisation de celui-ci sous forme d'un problème d'apprentissage avec bruit de classification. De même, cette étude a reçu l'assentiment de spécialistes du domaine qui ont autorisé sa publication dans [Magnan et al., 2007a, Magnan et al., 2007b].

4.3 Un protocole basé sur l'apprentissage avec bruit CCCN

Il découle immédiatement de l'étude présentée dans la section précédente un protocole pour détecter et extraire une information locale impliquée dans la formation de contacts ponctuels. Si cette information existe et si elle peut être représentée par une fonction appartenant à une classe de fonctions apprenable en contexte CCCN, alors on doit pouvoir l'apprendre par une méthode d'apprentissage appropriée et vérifier qu'elle est significative en vérifiant que les ponts se forment plus souvent entre deux résidus centraux d'environnements locaux à haut niveau d'affinité qu'entre des résidus dont les environnements locaux ont un faible niveau d'affinité l'un pour l'autre.

4.3.1 Le modèle de bruit CCCN

Le modèle de bruit de classification CCCN induit par la formalisation proposée d'une fonction d'affinité locale est proche de deux modèles de bruit de classification bien connus en apprentissage automatique (Sections 2.1.3 et 2.1.4) :

– il est plus général que le bruit de classification uniforme CN puisqu'il considère ce modèle de bruit sur chacune des classes sans imposer que les taux de bruit qui corrompent les exemples soient les mêmes pour chacune des classes. On peut également raisonnablement envisager qu'au moins un des deux taux de bruit η^+ ou η^- soit supérieur à 0.5. Cette éventualité est importante pour le problème biologique étudié ici puisque, quelle que soit la proportion de paires qui ont un fort niveau d'affinité, lorsque le nombre de ponts par protéine considéré augmente, la probabilité que ces paires forment un pont dans une protéine décroît rapidement, entraînant également une augmentation du taux de bruit η^+.

– ce modèle est proche d'un modèle de bruit CPCN pour lequel le nombre de partitions est fixe, connu et égal à 2 : la classe positive formant la première partition et la classe négative la deuxième. Le taux de bruit sur chacune des partitions est alors respectivement η^+ et η^-. Par contre, les taux de bruit η^+ et η^- ne sont pas contraints à être inférieurs à 0.5, ce qui constitue une différence avec le modèle de bruit de classification CPCN.

Ce modèle de bruit CCCN a déjà été considéré par Avrim Blum et Tom Mitchell lors des travaux publiés dans [Blum and Mitchell, 1998], dans lesquels les auteurs ont introduit le *co-training*. Aucun dénomination n'a été attribuée à ce modèle de bruit lors de ces travaux principalement voués à définir le *co-training*. Un résultat est donné dans cet article sur le modèle de bruit CCCN dans le cadre PAC, il annonce l'égalité entre les classes de concepts CN-apprenables et les classes de concepts CCCN-apprenables. Toutefois, ce résultat n'a pas été démontré, nous en fournissons la démonstration en annexe C.

Plusieurs études ont été menées sur l'apprentissage en contexte CN, et des méthodes performantes ont été proposées (voir Sections 2.1.3, 2.1.4 et 2.3.3). Toutefois, ces méthodes ne sont pas directement utilisables dans un cadre CCCN pour deux raisons essentielles : les hypothèses sur lesquelles sont basées ces méthodes ne sont plus vérifiées et parce que les calculs menés pour la construction des classifieurs dans ce contexte utilisent les propriétés induites par la présence de ce bruit. Ces propriétés ne sont plus valides lorsque le bruit qui corrompt la classe observée des données n'est plus de type CN.

Un exemple d'algorithme reconnu pour son efficacité, son expressivité et sa rapidité est la méthode des *SVM soft-margin* [Vapnik, 1995, Shawe-Taylor and Cristianini, 2000, Smola et al., 2000]. Cet algorithme est très performant même lorsque quelques *outliers* (exemples très distants de la distribution du reste des exemples) sont présents dans les données dont on dispose, ou lorsqu'un bruit de classification corrompt les exemples assez proches du séparateur cible (pour un taux de bruit raisonnable toutefois). A l'inverse, c'est un algorithme qui n'est pas tolérant à de très forts taux de bruit sur l'ensemble des exemples d'apprentissage, en particulier sur les exemples éloignés du séparateur recherché. Or, il y a de fortes raisons de penser que les taux de bruit qui affectent les données considérées ici soient assez importants, en particulier lorsque le nombre de ponts augmente. Cet algorithme ne peut donc pas être utilisé dans ce contexte d'apprentissage.

Ces constats entraînent la nécessité d'étudier l'apprentissage à partir de données corrompues par du bruit de classification conditionnel à chaque classe, et la nécessité de construire de nouvelles méthodes adaptées pour ce contexte d'apprentissage.

4.3.2 Le double intérêt d'une étude de l'apprentissage en contexte CCCN

Les deux chapitres qui suivent présentent une étude de l'apprentissage à partir de données sujettes à du bruit CCCN. Deux intérêts majeurs motivent cette étude :

- elle est nécessaire pour l'application du protocole de détection d'information locale proposé dans ce chapitre. En effet, il est nécessaire de connaître les conditions dans lesquelles on peut appliquer ce protocole et de mettre en place des méthodes appropriées à ce contexte d'apprentissage.

- le domaine de l'apprentissage automatique serait enrichi d'une telle étude car le modèle de bruit CCCN est plus général et plus réaliste que le modèle de bruit CN. Il est donc intéressant d'en étudier les contraintes et de proposer des méthodes adaptées à ce type de données.

Troisième partie

Etude de l'apprentissage avec bruit CCCN et Expérimentation du protocole de détection d'affinités locales

Chapitre 5

Apprentissage statistique supervisé avec bruit CCCN

Sommaire

Ce chapitre présente une première étude de l'apprentissage supervisé à partir de données corrompues par un bruit de classification de type CCCN (Section 4.3.1). Comme indiqué dans le chapitre précédent, il est indispensable, pour pouvoir appliquer le protocole proposé pour la détection d'une information locale impliquée dans l'appariement de résidus distants d'une protéine, de mener des études de ce contexte d'apprentissage et de proposer de nouvelles méthodes adaptées à ce modèle de bruit.

L'étude présentée ici se place dans le cadre général de l'apprentissage statistique et on considère donc uniquement des problèmes de classification supervisée binaires ($\mathcal{Y} = \{+, -\}$). On fait l'hypothèse dans ce contexte que la distribution P sur $\mathcal{X}$ sous-jacente au problème de classification est un mélange de deux distributions : la distribution $P(.|+)$ des exemples positifs et la distribution $P(.|-)$ des exemples négatifs. Lorsqu'un bruit de classification de type CCCN est ajouté aux données, les distributions $P^{\overrightarrow{\eta}}$ associées à chacune des classes deviennent elles-même des mélanges des distributions originales.

Dans certains cas particuliers, les classifieurs de Bayes associés aux distributions P et $P^{\overrightarrow{\eta}}$ sont identiques et la stratégie qui consiste à minimiser le risque empirique est consistante aussi bien pour la distribution originale que pour la distribution corrompue $P^{\overrightarrow{\eta}}$. Mais dans la plupart des cas, les classifieurs de Bayes associés aux deux distributions ne coïncident pas et il semble alors que le problème soit mal posé et qu'il soit nécessaire d'adopter d'autres stratégies.

Sous certaines hypothèses sur la distribution P sous-jacente au problème, on peut néanmoins montrer que le problème peut être résolu même lorsque les classifieurs de Bayes associés aux distributions P et $P^{\overrightarrow{\eta}}$ ne coïncident pas. C'est notamment le cas lorsque les distributions conditionnelles à chaque classe $P(.|y)$ appartiennent à un ensemble de distributions dont les 2-mélanges de distributions de cet ensemble sont identifiables. Dans ce cas, la distribution $P^{\overrightarrow{\eta}}$ détermine la distribution P et le problème devient bien posé.

Ce résultat a pour corollaire immédiat que lorsque les distributions conditionnelles à chaque classe sont des distributions produits (P respecte l'hypothèse naïve de Bayes), alors $P^{\overrightarrow{\eta}}$ détermine P et donc aussi les paramètres des classifieurs naïfs de Bayes. L'identifiabilité des mélanges finis de distributions produits a déjà été discutée au chapitre 3, elle a été montrée dans [Geiger et al., 2001, Whiley and Titterington, 2002]. Notons que dans l'étude présentée dans [Yang et al., 2003], les auteurs proposent une méthode pour éliminer du bruit de classification qui affecte les données observées, dédiée aux classifieurs naïfs de Bayes. Toutefois, le modèle de bruit considéré dans ces travaux ne peut être comparé au modèle CCCN : ils ne peuvent donc pas être utilisés dans ce cas.

L'identifiabilité de mélanges finis de distributions produits se fait sous des conditions assez faibles mais nous avons montré que cette identification avait une vitesse de convergence très lente dans les expériences menées en section 3.2.4. Identifier le mélange associé à chacune des classes séparément n'est donc pas une stratégie efficace pour identifier les paramètres de la distribution originale P. Nous établissons alors des formules analytiques pour identifier plus efficacement les paramètres des mélanges en fonction de la distribution corrompue $P^{\overrightarrow{\eta}}$. Ces formules permettent d'établir des estimateurs des paramètres du classifieur naïf de Bayes à partir de données de type CCCN. Nous montrons également qu'elles permettent de retrouver l'expression analytique du paramètre $P(+)$ (formule 3.5) établi dans la section 3.2.2 dans le cas semi-supervisé asymétrique où seuls des exemples positifs et non étiquetés sont disponibles.

Les estimateurs obtenus dans le contexte CCCN permettent de construire une adaptation de l'algorithme naïf de Bayes dans ce contexte, appelée `NB-CCCN`. De la même façon qu'en contexte semi-supervisé asymétrique, cet algorithme peut être combiné avec la méthode E.M. pour maximiser la vraisemblance des données d'apprentissage. L'algorithme proposé est appelé `NB-CCCN-EM`.

Afin de tester la qualité des estimations calculées par ces algorithmes et les performances sur la tâche de classification des classifieurs ainsi obtenus, nous présentons des expériences sur des données artificielles et sur des données issues de l'UCI [Asuncion and Newman, 2007]. Ces expériences montrent que le bruit CCCN peut être efficacement détecté et éliminé des données, montrant ainsi que les classifieurs naïfs de Bayes peuvent être efficacement calculés dans ce contexte.

5.1 Cas général

Cette première section présente une étude de l'apprentissage supervisé avec la présence de bruit de classification conditionnel à chaque classe. Après la présentation formelle de ce modèle de bruit sur les classes (Section 5.1.1), nous montrons que ce problème connaît une simplification naturelle dans certains cas (Section 5.1.2) mais que, sans hypothèse supplémentaire, ce problème est généralement mal posé (Section 5.1.3). Nous montrons également que certaines hypothèses permettent toutefois de rendre ce problème bien posé en donnant un exemple d'une telle hypothèse (Section 5.1.4).

5.1.1 Le bruit de classification conditionnel à chaque classe (CCCN)

Soit $\mathcal{X}$ un espace discret de description et $\mathcal{Y} = \{+, -\}$ l'ensemble des classes. On suppose en apprentissage supervisé que l'ensemble de données $S_{lab} = \{(x_1, y_1), ..., (x_l, y_l)\}$ dont on dispose est indépendamment et identiquement distribué selon une distribution P sur $\mathcal{X} \times \mathcal{Y}$ fixe mais inconnue (Section 2.1.2). L'objectif est alors de construire un classifieur $f : \mathcal{X} \to \mathcal{Y}$ qui minimise l'erreur réelle de f : $R(f) = P(y \neq f(x))$, c'est-à-dire qui approche au mieux le classifieur de Bayes f^* défini par $f^*(x) = \underset{y}{argmax}\ P(y|x)$.

On suppose ici que les exemples de S_{lab} sont soumis à un bruit de classification conditionnel à chaque classe (CCCN). On considère alors que les exemples de S_{lab} sont distribués indépendamment selon la distribution de probabilité $P^{\overrightarrow{\eta}}$ définie par :

$$\begin{cases} P^{\overrightarrow{\eta}}(x,+) = (1-\eta^+) \cdot P(x,+) + \eta^- \cdot P(x,-) \\ P^{\overrightarrow{\eta}}(x,-) = \eta^+ \cdot P(x,+) + (1-\eta^-) \cdot P(x,-) \end{cases} \tag{5.1}$$

avec $\overrightarrow{\eta} = (\eta^+, \eta^-) \in [0,1]^2$ le vecteur des taux de bruit qui affectent les données. La figure 5.1 illustre la définition de cette distribution $P^{\overrightarrow{\eta}}$. L'objectif d'un tel problème est identique au problème original (sans bruit de classification) : minimiser le risque d'un classifieur *relativement à la distribution* P. Remarquons toutefois que si l'on pose :

- $P'(x,+) = P(x,-)$ et $P'(x,-) = P(x,+)\ \forall x \in \mathcal{X}$
- $\eta'^- = 1 - \eta^+$ et $\eta'^+ = 1 - \eta^-$,

les distributions $P^{\overrightarrow{\eta}}$ et $P'^{\overrightarrow{\eta}}$ sont identiques alors que les classifieurs de Bayes associés à P et à P' sont complémentaires. Pour lever cette ambiguïté, on supposera que $\eta^+ + \eta^- \leq 1$.

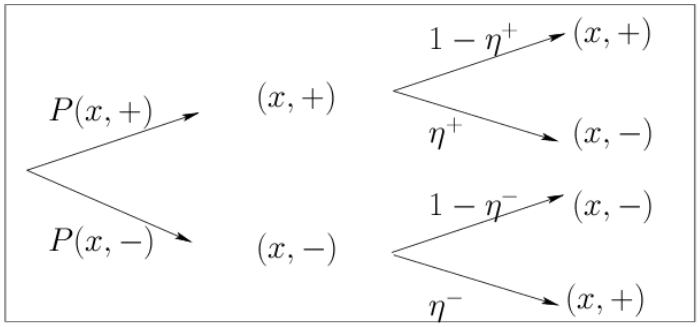


Figure 5.1. Définition d'un bruit additionnel de classification conditionnel à chaque classe.

De plus, si $\eta^+ + \eta^- = 1$, alors $P^{\overrightarrow{\eta}}(x|+) =$

$$\frac{P^{\overrightarrow{\eta}}(x,+)}{P^{\overrightarrow{\eta}}(+)} = \frac{(1-\eta^+)\cdot P(x,+) + \eta^- \cdot P(x,-)}{(1-\eta^+)\cdot P(+) + \eta^- \cdot (1-P(+))} = \frac{(1-\eta^+)\cdot P(x)}{1-\eta^+} = P(x)$$

et de même $P^{\overrightarrow{\eta}}(x|-) =$

$$\frac{P^{\overrightarrow{\eta}}(x,-)}{P^{\overrightarrow{\eta}}(-)} = \frac{\eta^+ \cdot P(x,+) + (1-\eta^-)\cdot P(x,-)}{(1-\eta^-)\cdot P(-) + \eta^+ \cdot (1-P(-))} = \frac{(1-\eta^-)\cdot P(x)}{(1-\eta^-)} = P(x)$$

On a donc $P^{\overrightarrow{\eta}}(x|+) = P^{\overrightarrow{\eta}}(x|-) = P(x)\ \forall x \in \mathcal{X}$, c'est-à-dire que les classes positives et négatives ne présentent plus aucune différence. On ne peut alors plus espérer faire mieux qu'une classification aléatoire des données. Nous supposerons donc que $\eta^+ + \eta^- < 1$.

5.1.2 Une simplification naturelle du problème dans certains cas

Nous montrons dans cette section que dans certains cas particuliers, ce problème d'apprentissage connaît une simplification naturelle. C'est notamment ce qu'il se passe lorsque les classifieurs de Bayes associés à chacune des deux distributions P et $P^{\overrightarrow{\eta}}$ sont identiques, c'est-à-dire : $P^{\overrightarrow{\eta}}(+|x) \geq P^{\overrightarrow{\eta}}(-|x) \Leftrightarrow P(+|x) \geq P(-|x)$. En effet :

$$\begin{aligned} & P^{\overrightarrow{\eta}}(+|x) \geq P^{\overrightarrow{\eta}}(-|x) \\ \Leftrightarrow\ & (1-\eta^+)\cdot P(+|x) + \eta^- \cdot P(-|x) \geq (1-\eta^-)\cdot P(-|x) + \eta^+ \cdot P(+|x) \\ \Leftrightarrow\ & (1-2\eta^+)\cdot P(+|x) \geq (1-2\eta^-)\cdot P(-|x) \end{aligned}$$

On observe que c'est par exemple le cas lorsque le bruit de classification est uniforme (CN), c'est-à-dire lorsque $\eta^-{=}\eta^+$ et pour un taux de bruit inférieur à 0.5, ces équivalences montrent en effet que les distributions P et $P^{\overrightarrow{\eta}}$ définissent exactement le même classifieur de Bayes. C'est également le cas lorsque le problème est déterministe, *i.e.* $P(+|x) = 0$ ou $P(-|x) = 0$ pour tout $x \in \mathcal{X}$, et que les taux de bruit η^- et η^+ sont inférieurs à 0.5.

Dans ces situations, la stratégie qui consiste à minimiser le risque empirique est consistante aussi bien pour la distribution originale P que pour la distribution $P^{\overrightarrow{\eta}}$: il est tout aussi pertinent de chercher un classifieur qui minimise le risque empirique sur des données tirées selon la distribution $P^{\overrightarrow{\eta}}$ que sur des données tirées selon la distribution P. Cela signifie notamment que la comparaison de deux classifieurs sur le taux d'erreur commis par ces classifieurs sur des données corrompues par du bruit CCCN est consistante et que l'erreur empirique est un bon indicateur de la qualité du classifieur appris. Mais lorsque les classifieurs de Bayes associés aux deux distributions ne coïncident pas, il devient nécessaire de trouver une autre stratégie.

5.1.3 Cas général : un problème mal posé

Nous montrons ici que sans hypothèse supplémentaire, le problème de l'apprentissage supervisé avec bruit de classification CCCN est un problème mal posé. Pour tout classifieur $f : \mathcal{X} \to \mathcal{Y}$, et en notant $p = P(f(x) = +)$ et $P_j(x,i) = P((x,i)|f(x) = j)$ $(i,j \in \{+,-\})$, le risque $R^{\overrightarrow{\eta}}(f) = P^{\overrightarrow{\eta}}(f(x) \neq y)$ s'écrit :

$$\begin{aligned} R^{\overrightarrow{\eta}}(f) &= P_-^{\overrightarrow{\eta}}(x,+)\cdot(1-p) + P_+^{\overrightarrow{\eta}}(x,-)\cdot p \\ &= [(1-\eta^+)P_-(x,+) + \eta^- P_-(x,-)](1-p) + [(1-\eta^-)P_+(x,-) + \eta^+ P_+(x,+)]p \\ &= (1-p)\cdot[(1-\eta^- - \eta^+)P_-(x,+) + \eta^-] + p\cdot[(1-\eta^- - \eta^+)P_+(x,-) + \eta^+] \\ &= (1-\eta^- - \eta^+)\cdot[P_-(x,+)\cdot(1-p) + P_+(x,-)\cdot p] + \eta^+\cdot p + \eta^-\cdot(1-p) \\ &= (1-\eta^- - \eta^+)\cdot R(f) + \eta^+\cdot p + \eta^-\cdot(1-p) \end{aligned}$$

On cherche donc à minimiser :

$$R(f) = \frac{R^{\overrightarrow{\eta}}(f) - \eta^+\cdot p - \eta^-\cdot(1-p)}{1-\eta^- - \eta^+} \tag{5.2}$$

ce qui ne correspond pas forcément à minimiser $R^{\overrightarrow{\eta}}(f)$ et ce qui peut être une tâche impossible puisqu'en général, on ne suppose pas que les taux de bruit sont connus.

Considérons un exemple simple. Soient $\mathcal{X} = \{a\}$, P_1 telle que $P_1(-|a) = 1/3$, $\overrightarrow{\eta_1} = (0,0)$, P_2 telle que $P_2(-|a) = 2/3$ et $\overrightarrow{\eta_2} = (1/2, 0)$. Les classifieurs de Bayes f_1 et f_2 associés aux distributions P_1 et P_2 sont complémentaires ($f_1(a) = +$ et $f_2(a) = -$) et de risque identique ($R(f_1) = R(f_2) = 1/3$). Par contre, les distributions $P_1^{\overrightarrow{\eta_1}}$ et $P_2^{\overrightarrow{\eta_2}}$ sont identiques ($P_1^{\overrightarrow{\eta_1}}(-|a) = P_2^{\overrightarrow{\eta_2}}(-|a) = 1/3$), les classifieurs de Bayes $f_1^{\overrightarrow{\eta_1}}$ et $f_2^{\overrightarrow{\eta_2}}$ associés le sont donc également ($f_1^{\overrightarrow{\eta_1}}(a) = f_2^{\overrightarrow{\eta_2}}(a) = +$) avec un risque $R^{\overrightarrow{\eta_1}}(f_1^{\overrightarrow{\eta_1}}) = R^{\overrightarrow{\eta_2}}(f_2^{\overrightarrow{\eta_2}}) = 1/3$. Minimiser le risque empirique pour $P_2^{\overrightarrow{\eta_2}}$ n'est donc pas une stratégie consistante, et comme $P_1^{\overrightarrow{\eta_1}}$ et $P_2^{\overrightarrow{\eta_2}}$ sont indistinguables et que les taux de bruit sont inconnus, le problème est insolvable.

5.1.4 Une première condition d'identifiabilité de la distribution P

L'apprentissage supervisé avec bruit CCCN est un problème mal posé dans le cas général. Toutefois, lorsqu'on fait l'hypothèse que la distribution originale P du problème appartient à une classe particulière de distributions, alors il se peut que le problème devienne solvable et que la distribution $P^{\overrightarrow{\eta}}$ d'où proviennent les exemples détermine à elle seule la distribution P. La notion d'identifiabilité d'une distribution en contexte CCCN peut se définir ainsi :

Définition 5.1. *Soit $\mathcal{P}$ un ensemble de distributions sur $\mathcal{X}\times\mathcal{Y}$. On dit que $\mathcal{P}$ est identifiable avec bruit de classification conditionnel à chaque classe si pour toute distribution $P\in\mathcal{P}$ et pour tout taux de bruit η^+ et η^- tels que $\eta^+ + \eta^- < 1$, $P^{\overrightarrow{\eta}}$ détermine P, c'est-à-dire que $\forall P_1, P_2 \in \mathcal{P}$, $\forall \overrightarrow{\eta}_1 = (\eta_1^-, \eta_1^+)$, $\overrightarrow{\eta}_2 = (\eta_2^-, \eta_2^+) \in [0,1]^2$ tels que $\eta_1^- + \eta_1^+ < 1$ et $\eta_2^- + \eta_2^+ < 1$:*

$$P_1^{\overrightarrow{\eta}_1} = P_2^{\overrightarrow{\eta}_2} \Rightarrow P_1 = P_2 \text{ et } \overrightarrow{\eta}_1 = \overrightarrow{\eta}_2.$$

Nous étudions en premier lieu la possibilité que $P^{\overrightarrow{\eta}}$ détermine P sans hypothèse supplémentaire sur la distribution P. Notons :

$$p = P(y=+) = \sum_{x\in X} P(x,+). \tag{5.3}$$

En posant $P_y(x) = P(x|y)$ pour toute distribution P, le système (5.1) peut se réécrire :

$$\begin{cases} P_+^{\overrightarrow{\eta}}(x) = \alpha P_+(x) + (1-\alpha)P_-(x) \\ P_-^{\overrightarrow{\eta}}(x) = \beta P_+(x) + (1-\beta)P_-(x) \end{cases} \tag{5.4}$$

avec

$$\alpha = \frac{p \cdot (1 - \eta^+)}{p \cdot (1 - \eta^+) + (1 - p) \cdot \eta^-} \tag{5.5}$$

$$\beta = \frac{p \cdot \eta^+}{p \cdot \eta^+ + (1 - p) \cdot (1 - \eta^-)} \tag{5.6}$$

Les distributions $P_+^{\overrightarrow{\eta}}(x)$ et $P_-^{\overrightarrow{\eta}}(x)$ peuvent donc s'exprimer comme des mélanges des deux distributions originales $P_+(x)$ et $P_-(x)$. A partir de ce système et des équations (5.5) et (5.6), on peut montrer facilement le lemme suivant :

Lemme 5.1. *Soient P une distribution de probabilité sur $\mathcal{X} \times \mathcal{Y}$, $\overrightarrow{\eta} = (\eta^-, \eta^+) \in [0,1]^2$ tels que $\eta^- + \eta^+ < 1$ et soient p, α et β définis respectivement en (5.3), (5.5) et (5.6). On a alors :*

$$(\alpha = 0 \Leftrightarrow p = 0) \Rightarrow \beta = 0$$
$$(\beta = 1 \Leftrightarrow p = 1) \Rightarrow \alpha = 1$$
$$(\alpha = \beta) \Leftrightarrow (p = 0 \vee p = 1)$$

Démonstration. Les relations se dérivent directement des formules (5.5) et (5.6). □

Ces relations permettent d'exprimer les taux de bruit η^+ et η^- qui corrompent les données de chaque classe en fonction de α, β et p :

$$\eta^- = \frac{(p - \beta)(1 - \alpha)}{(1 - p)(\alpha - \beta)} \text{ et } \eta^+ = \frac{\beta(\alpha - p)}{p(\alpha - \beta)}. \tag{5.7}$$

Même si α et β sont connus, les valeurs de p, η^- et η^+ ne sont pas encore déterminées car pour tout $p \in [min(\alpha, \beta), max(\alpha, \beta)]$, il existe des valeurs de η^+ et η^- qui sont consistantes avec les données. Toutefois, on montre la proposition suivante :

Proposition 5.1. *Soit $\mathcal{P}$ un ensemble de distributions de probabilités sur $\mathcal{X} \times \mathcal{Y}$ et soit $\mathcal{Q} = \{\ P(\cdot|y) \mid y{=}-$ ou $y{=}+, P \in \mathcal{P}\}$. Si les 2-mélanges de $\mathcal{Q}$ sont identifiables, alors $\mathcal{P}$ est identifiable avec bruit de classification conditionnel à chaque classe.*

Démonstration. Soient $P \in \mathcal{P}$ et η^-, η^+ des taux de bruit tels que $\eta^- + \eta^+ < 1$. Il existe des coefficients de mélange uniques α et β tels que :

$$P_+^{\overrightarrow{\eta}}(x) = \alpha P_+(x) + (1 - \alpha) P_-(x)$$
$$P_-^{\overrightarrow{\eta}}(x) = \beta P_+(x) + (1 - \beta) P_-(x)$$

Or, on a $P^{\overrightarrow{\eta}}(+) = (1 - \eta^+)p + \eta^-(1 - p) = \frac{\alpha(p-\beta)}{\alpha-\beta} + \frac{(1-\alpha)(p-\beta)}{\alpha-\beta} = \frac{p-\beta}{\alpha-\beta}$ et donc

$$p = \beta + (\alpha - \beta) \cdot P^{\overrightarrow{\eta}}(+). \tag{5.8}$$

Et les équations (5.7) déterminent η^- et η^+. □

On obtient donc une première condition pour laquelle la distribution $P^{\overrightarrow{\eta}}$ détermine la distribution P. Dans ce cas, le problème de l'apprentissage à partir de données corrompues par du bruit de classification conditionnel à chaque classe est faisable.

5.2 Apprendre les mélanges de distributions produits avec du bruit CCCN

Les résultats obtenus à la section précédente impliquent que si la distribution P sous-jacente au problème suit l'hypothèse naïve de Bayes, alors la distribution $P^{\overrightarrow{\eta}}$ détermine la distribution P et le problème devient bien posé. En effet, les 2-mélanges de distributions produits sont identifiables [Geiger et al., 2001] sous des conditions faibles.

Cela implique également que les paramètres des classifieurs naïfs de Bayes sont déterminés par la distribution $P^{\overrightarrow{\eta}}$ et donc estimables à partir de données corrompues par ce modèle de bruit. Néanmoins, les estimations que l'on obtiendrait à partir de données issues des distributions $P_+^{\overrightarrow{\eta}}$ et $P_-^{\overrightarrow{\eta}}$ de façon indépendante seraient des estimations dont la convergence est très lente. En effet, les expériences menées à la section 3.2.4 montrent que si on estime les deux mélanges issus des distributions $P_+^{\overrightarrow{\eta}}$ et $P_-^{\overrightarrow{\eta}}$ de façon indépendante, cette estimation souffrira de la lenteur de convergence des estimateurs. Nous montrons dans cette section qu'à partir de données issues de la distribution $P^{\overrightarrow{\eta}}$, on peut obtenir des estimations simples et efficaces des coefficients des deux mélanges ainsi que des paramètres qui dépendent de ces coefficients en considérant les deux mélanges ensemble.

5.2.1 Expressions analytiques des coefficients des mélanges

Nous fournissons ici les expressions analytiques des coefficients des mélanges définis par le système (5.4). Ces formules permettent d'établir par la suite des estimateurs des paramètres des classifieurs naïfs de Bayes en contexte CCCN. Toutefois, ces formules permettent également d'être utilisées pour retrouver la formule (3.5) donnée en section 3.2.2 pour le calcul des paramètres du classifieur naïf de Bayes en contexte semi-supervisé asymétrique. Pour cette raison, nous ne considérons pas directement le système (5.4) mais tout système qui a la même structure que celui-ci.

Soient (i, j) une paire d'indices d'attributs distincts ($i, j \in \{1, ..., m\}$, $i \neq j$), P_1 et P_2 deux distributions produits sur $\mathcal{X}^i \times \mathcal{X}^j$, et x^i et x^j les attributs correspondants à $\mathcal{X}^i$ et $\mathcal{X}^j$ ($x^i \in \mathcal{X}^i$ et $x^j \in \mathcal{X}^j$). Pour toute distribution P, pour tout $i \in \{1, ..., m\}$, et pour tout $k \in \mathcal{X}^i$, on note $P(x^i = k)$ par $P^i(k)$. Soient :

$$\begin{cases} Q_\alpha = \alpha \cdot P_1 + (1 - \alpha) \cdot P_2 \\ Q_\beta = \beta \cdot P_1 + (1 - \beta) \cdot P_2 \end{cases} \tag{5.9}$$

deux mélanges des distributions produits P_1 et P_2. On suppose que $\alpha \neq \beta$. On peut exprimer P_1 et P_2 comme des combinaisons linéaires de Q_α et Q_β :

$$\begin{cases} (\alpha - \beta) \cdot P_1 = (1 - \beta) \cdot Q_\alpha - (1 - \alpha) \cdot Q_\beta \\ (\alpha - \beta) \cdot P_2 = \alpha \cdot Q_\beta - \beta \cdot Q_\alpha \end{cases} \tag{5.10}$$

Soit $(k, l) \in \mathcal{X}^i \times \mathcal{X}^j$. P_1 et P_2 sont des distributions produits, donc on a :

$$Q_\alpha(k, l) = \alpha \cdot P_1(k, l) + (1 - \alpha) \cdot P_2(k, l) = \alpha P_1^i(k) P_1^j(l) + (1 - \alpha) P_2^i(k) P_2^j(l) \tag{5.11}$$

En remplaçant $P_1^i(k)$, $P_1^j(l)$, $P_2^i(k)$ et $P_2^j(l)$ par les expressions fournies par les équations (5.10), on obtient alors :

$$\begin{aligned}(\alpha-\beta)^2 Q_\alpha(k,l) = \quad & \alpha[(1-\beta)Q_\alpha^i(k)-(1-\alpha)Q_\beta^i(k)][(1-\beta)Q_\alpha^j(l)-(1-\alpha)Q_\beta^j(l)] \\ & +(1-\alpha)[\alpha Q_\beta^i(k)-\beta Q_\alpha^i(k)][\alpha Q_\beta^j(l)-\beta Q_\alpha^j(l)] \end{aligned} \tag{5.12}$$

Soit, après simplification :

$$(\alpha-\beta)^2 D = \alpha(1-\alpha)C, \tag{5.13}$$

avec

$$C = (Q_\alpha^i(k)-Q_\beta^i(k))(Q_\alpha^j(l)-Q_\beta^j(l)) \tag{5.14}$$

$$D = Q_\alpha(k,l) - Q_\alpha^i(k)Q_\alpha^j(l) \tag{5.15}$$

De la même façon que celle qui a permis de passer de l'expression de $Q_\alpha(k,l)$ (5.11) à l'équation (5.13), l'expression de $Q_\beta(k,l)$ permet d'obtenir :

$$(\alpha-\beta)^2 E = \beta(1-\beta)C, \tag{5.16}$$

avec

$$E = Q_\beta(k,l) - Q_\beta^i(k)Q_\beta^j(l) \tag{5.17}$$

On montre maintenant que l'on peut obtenir, à partir des équations (5.13) et (5.16), une expression analytique des coefficients de mélange α et β uniquement en fonction de Q_α et Q_β. Supposons tout d'abord que $\beta = 1$ ou $\beta = 0$, alors l'équation (5.13) peut être utilisée directement pour calculer α :

$$\alpha = \begin{cases} \frac{D}{D+C} \text{ si } \beta = 1 \\ \frac{C}{D+C} \text{ si } \beta = 0 \end{cases} \tag{5.18}$$

Supposons maintenant que $\beta(1-\beta) \neq 0$. A partir de l'équation (5.13), on obtient :

$$\alpha^2 = \frac{\alpha C - \beta D(\beta - 2\alpha)}{C+D}$$

En remplaçant l'expression de α^2 obtenue dans l'équation (5.16), on obtient une expression de α comme une fonction de β :

$$\alpha = \beta \cdot \frac{(1-\beta)(C+D) - \beta E}{E(1-2\beta)} \tag{5.19}$$

On peut maintenant introduire cette expression de α (5.19) dans l'équation (5.16) et on obtient alors l'équation du quatrième degré suivante :

$$\beta \cdot (1-\beta) \cdot (\beta^2 - \beta + \lambda_\beta) = 0 \tag{5.20}$$

avec

$$\lambda_\beta = \frac{CE}{(C+D+E)^2 - 4DE} \tag{5.21}$$

On a supposé que $\beta(1-\beta) \neq 0$, donc

$$\beta \in \left\{ \frac{1+\sqrt{1-4\lambda_\beta}}{2}, \frac{1-\sqrt{1-4\lambda_\beta}}{2} \right\} \tag{5.22}$$

ce qui signifie que deux solutions (α_1, β_1) et (α_2, β_2) sont admissibles pour le problème puisque β détermine α par (5.19). Notons que $\alpha_2 = 1 - \alpha_1$ et que $\beta_2 = 1 - \beta_1$.

On a donc prouvé la proposition suivante :

Proposition 5.2. *Soient $Q_\alpha = \alpha P_1 + (1-\alpha)P_2$ et $Q_\beta = \beta P_1 + (1-\beta)P_2$ deux mélanges de distributions produits P_1 et P_2. Si on suppose que $\alpha \neq \beta$, alors (5.18), (5.19) et (5.22) fournissent des expressions analytiques des coefficients de mélange α et β.*

5.2.2 Classifieur naïf de Bayes en contexte semi-supervisé asymétrique

En section 3.2, nous avions montré que les paramètres des classifieurs naïfs de Bayes étaient déterminés et efficacement estimables en contexte semi-supervisé asymétrique dans lequel on suppose que les seuls exemples observés sont positifs et non étiquetés distribués respectivement selon $P(.|+)$ et $P(.)$ sur $\mathcal{X}$. Nous avions alors montré que tous les paramètres du modèle θ qui définit le classifieur naïf de Bayes étaient déterminés dès lors que le paramètre $P(+)$ l'était. Nous avons fourni une formule analytique qui permet de calculer ce paramètre à partir des distributions connues (formule 3.5).

La proposition 5.2 montrée à la section précédente peut également être appliquée pour retrouver cette expression de $P(+)$ en contexte semi-supervisé asymétrique. C'est la raison pour laquelle nous n'avons pas directement exprimé α et β dans le contexte d'apprentissage CCCN. Ce résultat constitue un corollaire de la proposition 5.2 :

Corollaire 5.1. *Soit P une distribution sur $\mathcal{X}\times\mathcal{Y}$ telle que P_+ et P_- sont des distributions produits sur $\mathcal{X}$. Soient x^i et x^j deux attributs distincts, (k,l) deux valeurs de ces attributs ($k \in \mathcal{X}^i$, $l \in \mathcal{X}^j$) et notons, comme dans la section 3.2.2, $\alpha_{ik,jl} = P(x^i = k, x^j = l)$, $\alpha_{ik} = P(x^i = k)$ et $p_{ik} = P(x^i = k|+)$. Alors :*

$$P(+) = \frac{\alpha_{ik,jl} - \alpha_{ik} \cdot \alpha_{jl}}{\alpha_{ik,jl} + p_{ik} \cdot p_{jl} - \alpha_{ik} \cdot p_{jl} - p_{ik} \cdot \alpha_{jl}} \tag{5.23}$$

Démonstration. Posons $Q_\alpha = \alpha_{ik,jl} = P(+)P(x^i{=}k, x^j{=}l|+) + (1-P(+))P(x^i{=}k, x_j{=}l|-)$ et $Q_\beta = P(x^i{=}k, x^j{=}l|+)$. Avec les notations $p_{ik} = P(x^i{=}k|+)$, $q_{ik} = P(x^i{=}k|-)$ et en rappelant que l'on suppose que les distributions $P(.|+)$ et $P(.|-)$ sont des distributions produits alors Q_α et Q_β s'écrivent :

$$Q_\alpha = \alpha_{ik,jl} = P(+) \cdot p_{ik}p_{jl} + (1 - P(+)) \cdot q_{ik}q_{jl} \text{ et } Q_\beta = p_{ik}p_{jl}$$

Q_α est un mélange des deux distributions produits $P(.|+)$ et $P(.|-)$ avec $\alpha = P(+)$ comme coefficient de mélange. De même pour Q_β mais avec un coefficient de mélange $\beta = 1$. La formule 5.18 permet alors d'obtenir la formule du corollaire. □

5.2.3 Classifieur naïf de Bayes en contexte CCCN

Nous montrons maintenant que les formules (5.22) et (5.19) peuvent être utilisées pour calculer tous les paramètres des classifieurs naïfs de Bayes en contexte CCCN.

Soient (i, j) une paire d'indices d'attributs distincts, $\mathcal{X}^i$ et $\mathcal{X}^j$ les domaines de ces attributs. Soient P_1 et P_2 les distributions définies sur $\mathcal{X}^i \times \mathcal{X}^j$ par $P_1(k, l) = P_+(x^i{=}k, x^j{=}l)$ et $P_2(k, l) = P_-(x^i{=}k, x^j{=}l)$ et soient $Q_\alpha(k, l) = P_+^{\overrightarrow{\eta}}(x^i{=}k, x^j{=}l)$ et $Q_\beta(k, l) = P_-^{\overrightarrow{\eta}}(x^i{=}k, x^j{=}l)$ tel qu'indiqué par le système (5.4).

Deux paires de coefficients des mélanges (α_1, β_1) et (α_2, β_2) sont admissibles et peuvent être calculées en utilisant les équations (5.22) et (5.19). Chaque paire (α, β) de solutions admissibles permet d'obtenir une valeur pour les paramètres p, η^+ et η^- par les équations (5.8) et (5.7). Une seule de ces solutions est telle que $\eta^+ + \eta^- < 1$. On peut alors calculer les paramètres manquants des classifieurs naïfs de Bayes avec les équations (5.10).

5.2.4 Algorithme naïf de Bayes en contexte CCCN - `NB-CCCN`

Nous proposons à partir des résultats obtenus dans les sections précédentes, une adaptation de l'algorithme naïf de Bayes pour l'apprentissage des classifieurs naïfs de Bayes à partir de données corrompues par du bruit CCCN.

Soit $S_{lab}^{\overrightarrow{\eta}}$ un ensemble de données distribuées selon la distribution $P^{\overrightarrow{\eta}}$. Pour chaque paire d'attributs x^i et x^j et pour toute paire de valeurs d'attributs $(k, l) \in \mathcal{X}^i \times \mathcal{X}^j$, soient :

$$\begin{aligned}
\hat{C}_{ik,jl} &= (\widehat{P_+^{\overrightarrow{\eta}}}(x^i = k) - \widehat{P_-^{\overrightarrow{\eta}}}(x^i = k)) \cdot (\widehat{P_+^{\overrightarrow{\eta}}}(x^j = l) - \widehat{P_-^{\overrightarrow{\eta}}}(x^j = l)),\\
\hat{D}_{ik,jl} &= \widehat{P_+^{\overrightarrow{\eta}}}(x^i = k, x^j = l) - \widehat{P_+^{\overrightarrow{\eta}}}(x^i = k)\widehat{P_+^{\overrightarrow{\eta}}}(x^j = l),\\
\hat{E}_{ik,jl} &= \widehat{P_-^{\overrightarrow{\eta}}}(x^i = k, x^j = l) - \widehat{P_-^{\overrightarrow{\eta}}}(x^i = k)\widehat{P_-^{\overrightarrow{\eta}}}(x^j = l)
\end{aligned}$$

où les $\widehat{P_+^{\overrightarrow{\eta}}}$ et les $\widehat{P_-^{\overrightarrow{\eta}}}$ sont des estimations empiriques des $P_+^{\overrightarrow{\eta}}$ et des $P_-^{\overrightarrow{\eta}}$ correspondants calculées sur $S_{lab}^{\overrightarrow{\eta}}$. Une estimation $\hat{\lambda}_\beta$ de $\lambda_\beta = \beta - \beta^2$ est alors calculée par la formule :

$$\hat{\lambda}_\beta = \frac{\sum \hat{C}_{ik,jl}\hat{E}_{ik,jl}}{\sum(\hat{C}_{ik,jl} + \hat{D}_{ik,jl} + \hat{E}_{ik,jl})^2 - 4\hat{D}_{ik,jl}\hat{E}_{ik,jl}} \tag{5.24}$$

où les sommes sont considérées sur toutes les paires (i, j) d'attributs et toutes les paires (k, l) de valeurs de ces attributs (voir formule (5.21)). De façon similaire à l'obtention de la formule (5.21), on peut établir une formule qui permet de calculer directement les solutions α_1 et α_2 admissibles. La solution de l'équation correspondante $\lambda_\alpha = \alpha - \alpha^2$ peut alors être estimée par la formule :

$$\hat{\lambda}_\alpha = \frac{\sum \hat{C}_{ik,jl}\hat{D}_{ik,jl}}{\sum(\hat{C}_{ik,jl} + \hat{D}_{ik,jl} + \hat{E}_{ik,jl})^2 - 4\hat{D}_{ik,jl}\hat{E}_{ik,jl}} \tag{5.25}$$

Soient β_1 et β_2 (resp. α_1 et α_2) les deux solutions de l'équation $\hat{\lambda}_\beta = \beta - \beta^2$ (resp. $\hat{\lambda}_\alpha = \alpha - \alpha^2$). Une seule de ces paires est compatible avec les hypothèses. Un modèle θ peut alors être calculé grâce aux équations (5.10) et (5.8). Cet algorithme est décrit par l'algorithme 8, il est noté `NB-CCCN`.

Algorithme 8 `NB-CCCN` - Algorithme naïf de Bayes CCCN

Entrée: $S_{lab}^{\overrightarrow{\eta}}$, un ensemble de données corrompues par du bruit CCCN

1 - Calculer $\hat{\lambda}_\alpha$ et $\hat{\lambda}_\beta$ avec les formules (5.25) et (5.24) sur $S_{lab}^{\overrightarrow{\eta}}$
2 - Calculer les valeurs de α et β admissibles en résolvant $\hat{\lambda}_\beta = \beta - \beta^2$ et $\hat{\lambda}_\alpha = \alpha - \alpha^2$
3 - Sélectionner l'unique paire de solution admissible $(\hat{\alpha}, \hat{\beta})$
4 - Calculer les paramètres du modèle $\hat{\theta}$ avec (5.10) et (5.8)

Sortie: un modèle $\hat{\theta}$

5.2.5 Algorithme naïf de Bayes `NB-CCCN` avec E.M. - `NB-CCCN-EM`

Etant donné un échantillon $S_{lab}^{\overrightarrow{\eta}}$ de données corrompues par du bruit de classification conditionnel à chaque classe, on peut construire un classifieur naïf de Bayes par l'estimateur du maximum de vraisemblance si on sait quels exemples sont corrompus, ce qui n'est pas le cas dans cette situation.

Toutefois, comme indiqué dans la section 2.2, la méthode E.M. peut alors être utilisée dans ce type de situation. A la section 3.2.3, nous avons montré comment, en contexte semi-supervisé asymétrique, on peut appliquer la méthode E.M. à partir d'un premier modèle calculé sur les données. Les données manquantes de la méthode E.M. étaient alors les classes des exemples non étiquetés de l'ensemble S_{unl}.

Dans le contexte CCCN considéré ici, les données manquantes de la méthode E.M. sont les indicateurs des données de $S_{lab}^{\overrightarrow{\eta}}$ qui sont corrompues par du bruit de classification. Or, le modèle $\theta = \{p = P(+), p_{ik} = P(x^i = k|+), q_{ik} = P(x^i = k|-)\}$ et le vecteur de bruit $\overrightarrow{\eta} = \{\eta^+, \eta^-\}$ que l'on peut obtenir respectivement par l'algorithme `NB-CCCN` et les formules (5.7) pendant le déroulement de cet algorithme permettent de calculer pour tout exemple $(x, y) \in S_{lab}^{\overrightarrow{\eta}}$, la probabilité $P(C(x, y)|\theta, \overrightarrow{\eta})$ (notée par la suite $P(C(x, y))$) que l'exemple (x, y) soit corrompu par du bruit dans le modèle $(\theta, \overrightarrow{\eta})$.

En utilisant la notation $\bar{y}$ pour la classe opposée de y quel que soit $y \in \{+, -\}$, cette probabilité vaut :

$$P(C(x, y)) = \frac{P(\bar{y}|x, \theta) \cdot \eta^{\bar{y}}}{P(\bar{y}|x, \theta) \cdot \eta^{\bar{y}} + P(y|x, \theta) \cdot (1 - \eta^y)} \tag{5.26}$$

Cette formule permet alors de calculer pour tout exemple de $S_{lab}^{\overrightarrow{\eta}}$ et pour tout $y \in \{+, -\}$ la probabilité que la classe de cet exemple était y avant l'application du bruit.

Elle permet donc d'appliquer la méthode E.M. en posant comme modèle initial $(\theta_0, \overrightarrow{\eta_0})$ le modèle calculé par l'algorithme `NB-CCCN` et en utilisant les formules qui suivent pour le calcul d'un modèle $(\theta_{n+1}, \overrightarrow{\eta_{n+1}})$ étant donné le modèle courant $(\theta_n, \overrightarrow{\eta_n})$ de l'algorithme E.M. de façon à maximiser la vraisemblance de $S_{lab}^{\overrightarrow{\eta}}$.

En notant $l = |S_{lab}^{\overrightarrow{\eta}}|$, $S_y^{\overrightarrow{\eta}} = \{(x,y) \in S_{lab}^{\overrightarrow{\eta}}\}$, les paramètres p^{n+1}, p_{ik}^{n+1} et q_{ik}^{n+1} du modèle θ_{n+1} et le vecteur de bruit $\overrightarrow{\eta_{n+1}} = \{\eta_{n+1}^+, \eta_{n+1}^-\}$ se calculent alors de la façon suivante étant donné le modèle θ_n et le vecteur $\overrightarrow{\eta_n}$ courants :

$$p^{n+1} = \frac{\sum\limits_{S_+^{\overrightarrow{\eta}}} (1 - P(C(x,+)|\theta_n, \overrightarrow{\eta_n})) + \sum\limits_{S_-^{\overrightarrow{\eta}}} P(C(x,-)|\theta_n, \overrightarrow{\eta_n})}{l} \tag{5.27}$$

$$p_{ik}^{n+1} = \frac{\sum\limits_{\substack{(x,+)\in S_+^{\overrightarrow{\eta}} \\ /x^i=k}} (1 - P(C(x,+)|\theta_n, \overrightarrow{\eta_n})) + \sum\limits_{\substack{(x,-)\in S_-^{\overrightarrow{\eta}} \\ /x^i=k}} P(C(x,-)|\theta_n, \overrightarrow{\eta_n})}{l} \tag{5.28}$$

$$q_{ik}^{n+1} = \frac{\sum\limits_{\substack{(x,+)\in S_+^{\overrightarrow{\eta}} \\ /x^i=k}} P(C(x,+)|\theta_n, \overrightarrow{\eta_n}) + \sum\limits_{\substack{(x,-)\in S_-^{\overrightarrow{\eta}} \\ /x^i=k}} (1 - P(C(x,-)|\theta_n, \overrightarrow{\eta_n}))}{l} \tag{5.29}$$

$$\eta_{n+1}^+ = \frac{\sum\limits_{S_-^{\overrightarrow{\eta}}} P(C(x,-)|\theta_n, \overrightarrow{\eta_n})}{\sum\limits_{S_+^{\overrightarrow{\eta}}} (1 - P(C(x,+)|\theta_n, \overrightarrow{\eta_n})) + \sum\limits_{S_-^{\overrightarrow{\eta}}} P(C(x,-)|\theta_n, \overrightarrow{\eta_n})} \tag{5.30}$$

$$\eta_{n+1}^- = \frac{\sum\limits_{S_+^{\overrightarrow{\eta}}} P(C(x,+)|\theta_n, \overrightarrow{\eta_n})}{\sum\limits_{S_-^{\overrightarrow{\eta}}} (1 - P(C(x,-)|\theta_n, \overrightarrow{\eta_n})) + \sum\limits_{S_+^{\overrightarrow{\eta}}} P(C(x,+)|\theta_n, \overrightarrow{\eta_n})} \tag{5.31}$$

Nous notons `NB-CCCN-EM` l'algorithme correspondant (Algorithme 9). Cet algorithme fournit un modèle θ_c qui maximise localement la vraisemblance de $S_{lab}^{\overrightarrow{\eta}}$.

Algorithme 9 `NB-CCCN-EM` - Algorithme naïf de Bayes CCCN avec E.M.

Entrée: $S_{lab}^{\overrightarrow{\eta}}$, un ensemble de données corrompues par du bruit CCCN

1 - $(\theta_0, \overrightarrow{\eta_0}) =$ `NB-CCCN`$(S_{lab}^{\overrightarrow{\eta}})$
2 - $\forall (x,y) \in S_{lab}^{\overrightarrow{\eta}}$, calculer $Pr(C(x,y)|\theta_n, \overrightarrow{\eta_n})$ avec la formule (5.26) pour le n courant
3 - Calculer le modèle θ_{n+1} avec les formules (5.27), (5.28) et (5.29)
4 - Calculer le vecteur $\overrightarrow{\eta_{n+1}}$ avec les formules (5.30) et (5.31)
5 - Itérer à l'étape 2 jusqu'à convergence

Sortie: le modèle θ_c obtenu après convergence

5.3 Evaluation des algorithmes NB-CCCN et NB-CCCN-EM

Cette section présente deux séries d'expériences menées respectivement sur des données artificielles et sur des données issues de l'UCI [Asuncion and Newman, 2007] qui ont pour objectif d'évaluer empiriquement les performances des algorithmes NB-CCCN et NB-CCCN-EM sur deux catégories de données différentes : les premières, artificielles, sont générées selon le modèle CCCN pour lequel les algorithmes ont été proposés, et les secondes, issues de l'UCI, n'ayant aucune garantie de correspondre au modèle désiré.

Nous comparons dans ces expériences les performances de ces algorithmes à celles de deux autres versions de l'algorithme naïf de Bayes qui peuvent être utilisées dans ce contexte :

- l'algorithme NB supervisé (Algorithme 1) directement lancé sur les ensembles $S_{lab}^{\overrightarrow{\eta}}$ de données corrompues par du bruit CCCN. Cet algorithme apprend alors les paramètres de la distribution $P^{\overrightarrow{\eta}}$ sans prendre en compte le bruit présent,

- l'algorithme NB-UNL non supervisé (Algorithme 7), qui peut également être utilisé lorsque les données sont corrompues par du bruit de classification puisqu'il ne prend pas en compte les étiquettes des exemples.

Notons dès maintenant que si ces expériences montrent que le bruit CCCN peut être efficacement éliminé des données et que l'on peut apprendre efficacement les classifieurs de Bayes dans ce contexte, des données test qui seraient également bruitées selon le même modèle de bruit ne pourraient pas en attester.

En effet, la formule (5.2) établie dans la section 5.1.3 de ce chapitre montre que la minimisation du risque empirique n'est pas une bonne stratégie lorsque du bruit CCCN affecte les données. Dans une situation classique, les données de test ou de validation qui servent à sélectionner ou évaluer un classifieur sont généralement corrompues par du bruit de classification si les données d'apprentissage le sont. L'inconsistance de cette stratégie implique qu'il est alors difficile de comparer ou évaluer des classifieurs appris sur des données corrompues par du bruit CCCN puisqu'il n'est alors pas consistant de comparer l'erreur empirique commise par ces classifieurs.

Concernant les classifieurs naïfs de Bayes, nous n'avons actuellement pas de stratégie pour pallier ce problème d'évaluation en contexte CCCN, à la différence des travaux présentés au chapitre suivant sur l'apprentissage de séparateurs linéaires. De façon plus générale, nous n'avons pas encore trouvé de stratégie consistante et indépendante de la classe de fonctions dans laquelle on travaille pour résoudre ce problème. Y a-t-il une stratégie consistante dans tous les cas ? La formule 5.2 permet-elle de sélectionner un modèle ? La question reste actuellement ouverte.

De ce fait, les séries d'expériences présentées dans les sections suivantes respectent des protocoles qui assurent que les données servant à tester les classifieurs appris sont non corrompues par du bruit de classification. Cela permet alors d'estimer directement le risque de ces classifieurs pour la distribution originale des exemples et non pour la distribution corrompue.

5.3.1 Expériences sur des données artificielles

Cette section présente une série d'expériences menées sur des données générées artificiellement à partir de modèles θ_c tirés aléatoirement. La connaissance des modèles cibles qui génèrent les données permet ainsi d'évaluer les algorithmes proposés en contexte CCCN aussi bien sur leurs capacités à estimer correctement les paramètres du modèle cible que sur les performances de classification des classifieurs définis par les modèles appris.

Protocole expérimental

Du fait des conditions d'application de l'algorithme `NB-UNL` (Algorithme 7), nous considérons ici des attributs binaires ($\mathcal{X} = \{0,1\}^m$). De plus, les différentes séries d'expériences menées pour différentes valeurs du nombre m d'attributs binaires menant à des courbes et des conclusions similaires, nous ne présentons ici que les résultats obtenus pour un nombre d'attributs $m = 10$ ($\mathcal{X} = \{0,1\}^{10}$).

Le modèle cible $\theta_c = \{P(+), P_+, P_-\} = \{p, p_{ik}, q_{ik}, k \in \{0,1\}, i \in \{1,...,10\}\}$ à partir duquel les données sont générées est tiré aléatoirement. Les distributions P_+ et P_- de θ_c sont des distributions produits sur $\mathcal{X}$ tirées selon une distribution uniforme. Les ensembles d'apprentissage et de test sont tirés selon le modèle θ_c.

Pour chaque nombre l de données appartenant à l'ensemble $\{100, 200, ..., 3000\}$, nous générons 200 ensembles $S_{lab} = \{(x_1, y_1), ..., (x_l, y_l)\}$ de données d'apprentissage non corrompues par du bruit de classification à partir du modèle cible θ_c. On simule pour chacun de ces ensembles S_{lab} un processus de bruit CCCN en échangeant l'étiquette de chaque donnée positive avec une probabilité η^+ et l'étiquette de chaque donnée négative avec une probabilité η^-. Les résultats présentés considèrent un bruit $\overrightarrow{\eta} = (\eta^+, \eta^-) = (0.5, 0.2)$. On obtient ainsi des ensembles $S_{lab}^{\overrightarrow{\eta}}$ qui respectent les conditions d'un bruit de classification CCCN. Un ensemble de données test S_{test} de taille 1000 est également généré à partir de θ_c ; ces données ne sont pas bruitées pour les raisons évoquées en introduction de la section 5.3. Tous les résultats présentés dans cette section sont des moyennes calculées sur les 200 expériences correspondant aux 200 ensembles de données d'apprentissage de taille l.

Précision des estimations des paramètres du classifieur

Dans un premier temps, nous comparons la qualité des estimations des paramètres des classifieurs naïfs de Bayes fournies par les quatre algorithmes `NB`, `NB-CCCN`, `NB-CCCN-EM` et `NB-UNL` pour la série d'expériences décrite précédemment.

Le critère choisi pour la comparaison de ces estimations est la divergence de Kullback-Leibler d_{kl} entre la distribution cible P sur $\mathcal{X} \times \mathcal{Y}$ et les distributions estimées $\hat{P}$ sur $\mathcal{X} \times \mathcal{Y}$ par chaque algorithme. La divergence de Kullback-Leibler entre deux distributions quelconques P et P' définies sur un ensemble discret $\mathcal{X}$ vaut :

$$d_{kl}(P, P') = \sum_{x \in \mathcal{X}} P(x) log \frac{P(x)}{P'(x)}$$

Cette mesure ne définit pas une mesure de distance puisqu'elle n'est pas symétrique. Toutefois, elle est connue pour être un très bon indicateur de la proximité entre deux distributions de probabilités.

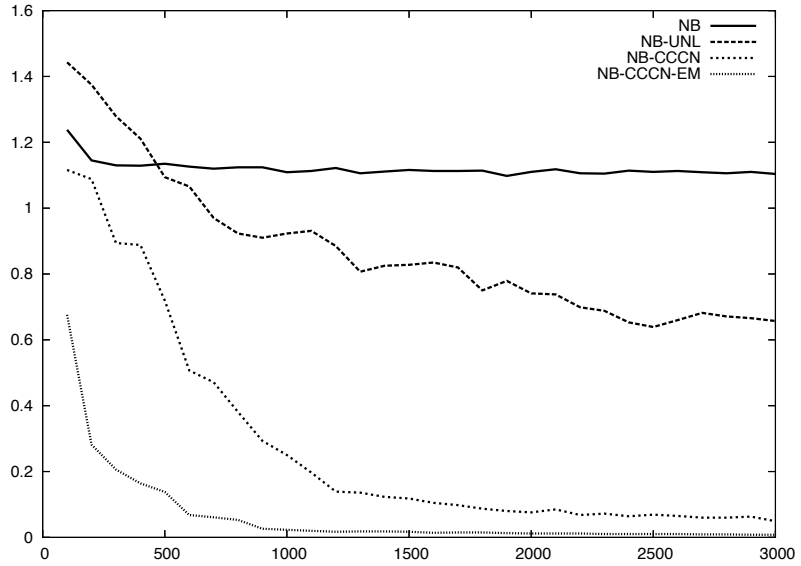


Figure 5.2. Moyennes des divergences de Kullback-Leibler entre la distribution cible et celles calculées par les algorithmes NB, NB-CCCN, NB-CCCN-EM et NB-UNL en fonction de la taille de l'ensemble d'apprentissage.

La figure 5.2 montre l'évolution de la divergence de Kullback-Leibler entre la distribution cible et les distributions estimées $\hat{P}$ par les quatre algorithmes en fonction de la taille de l'ensemble de données d'apprentissage $S_{lab}^{\overrightarrow{\eta}}$.

Ces résultats montrent que les algorithmes NB-CCCN et NB-CCCN-EM fournissent de bien meilleures estimations des probabilités de la distribution cible et convergent beaucoup plus rapidement que les autres algorithmes. On constate également que les estimations fournies par l'algorithme NB-UNL convergent très lentement et sont beaucoup moins précises que celles obtenues par les deux algorithmes naïfs de Bayes en contexte CCCN. Ce constat est également fait dans les expériences menées à la section 3.2.4 sur l'estimation des paramètres du classifieur naïf de Bayes en contexte semi-supervisé asymétrique.

Des expériences similaires ont été menées dans lesquelles le modèle initial choisi de l'algorithme NB-CCCN-EM n'est pas le modèle issu de l'algorithme NB-CCCN mais un modèle tiré aléatoirement avec plusieurs répétitions de l'expérience puis sélection parmi les modèles obtenus de celui qui maximise la vraisemblance de $S_{lab}^{\overrightarrow{\eta}}$. Ces expériences montrent que les résultats sont similaires à ceux obtenus avec le modèle initial issu de l'algorithme NB-CCCN, mais nécessitent un temps d'exécution très important car de nombreux modèles initiaux doivent alors être tirés pour obtenir un bon maximum local de la vraisemblance alors qu'une seule exécution de l'algorithme suffit dans le premier cas.

C'est un résultat intéressant qui montre que les modèles calculés par l'algorithme NB-CCCN (sans utiliser la méthode EM), ne sont pas très éloignés de ceux qui maximisent la vraisemblance de $S_{lab}^{\overrightarrow{\eta}}$. Ils montrent également que les deux algorithmes NB-CCCN et NB-CCCN-EM peuvent être associés pour apprendre efficacement les paramètres des classifieurs naïfs de Bayes.

Comparaison des performances sur la tâche de classification

Nous nous intéressons maintenant aux performances sur la tâche de classification des classifieurs appris par les quatre algorithmes considérés dans cette section. Les résultats présentés ici ont été obtenus par la même série d'expériences que celle qui a permis d'obtenir les résultats présentés précédemment sur la qualité des estimations des paramètres du classifieur calculées par les mêmes algorithmes.

Deux critères sont étudiés ici :

– le risque empirique $\hat{R}(f) = P(f(x) \neq y)$ des classifieurs obtenus par les quatre algorithmes calculé sur l'ensemble test S_{test} ou le taux empirique de bonne prédiction calculé sur S_{test} qui est égal à $1 - \hat{R}(f)$,

– le *F-score* ou *F-measure* défini par $F = \frac{2 \cdot TP}{FP + 2 \cdot TP + FN}$ où TP est le nombre d'exemples positifs de S_{test} correctement classés, FP le nombre d'exemples positifs de S_{test} incorrectement classés, et FN le nombre d'exemples négatifs de S_{test} incorrectement classés.

Les résultats obtenus pour ces deux critères sont indiqués dans la table 5.1 et les figures 5.3 et 5.4 proposent une vue graphique de l'évolution des taux d'erreurs empiriques et des F-scores, calculés sur l'ensemble S_{test}, des classifieurs appris en fonction de la taille de l'ensemble d'apprentissage. Ces résultats montrent que les trois algorithmes `NB-UNL`, `NB-CCCN` et `NB-CCCN-EM` permettent d'apprendre les classifieurs naïfs de Bayes malgré la présence de bruit de classification CCCN. Toutefois l'algorithme `NB-UNL` converge beaucoup plus lentement vers un classifieur performant. De façon tout à fait cohérente à l'étude menée dans les sections précédentes, l'algorithme `NB` n'identifie pas du tout les paramètres de la distribution originale et ne permet pas de converger vers le modèle cible θ_c. Ces résultats illustrent l'étude théorique menée sur l'apprentissage en contexte CCCN et montrent que les classifieurs naïfs de Bayes peuvent être efficacement appris en présence de bruit CCCN.

Algorithme	$\lvert S_{lab}\rvert =$	100	500	2000	3000
θ_c	$1-\hat{R}(f)$	0.884	0.884	0.884	0.884
	F	0.925	0.925	0.925	0.925
`NB`	$1-\hat{R}(f)$	0.534 (*0.007*)	0.571 (*0.002*)	0.584 (*0.001*)	0.590 (*0.000*)
	F	0.562 (*0.012*)	0.600 (*0.005*)	0.619 (*0.001*)	0.627 (*0.001*)
`NB-CCCN`	$1-\hat{R}(f)$	**0.742** (*0.002*)	0.793 (*0.001*)	0.857 (*0.000*)	0.868 (*0.000*)
	F	**0.842** (*0.004*)	0.876 (*0.000*)	0.909 (*0.000*)	0.915 (*0.000*)
`NB-CCCN-EM`	$1-\hat{R}(f)$	0.723 (*0.014*)	**0.845** (*0.005*)	**0.878** (*0.000*)	**0.879** (*0.000*)
	F	0.810 (*0.013*)	**0.897** (*0.003*)	**0.921** (*0.000*)	**0.921** (*0.000*)
`NB-UNL`	$1-\hat{R}(f)$	0.645 (*0.006*)	0.705 (*0.006*)	0.795 (*0.004*)	0.834 (*0.002*)
	F	0.722 (*0.009*)	0.777 (*0.007*)	0.851 (*0.005*)	0.881(*0.002*)

Table 5.1. Résultats pour la tâche de classification obtenus par les classifieurs calculés avec les quatre algorithmes `NB`, `NB-CCCN`, `NB-CCCN-EM`, `NB-UNL` et par le modèle cible θ_c en fonction de la taille de l'ensemble d'apprentissage (Ecarts-types en italique).

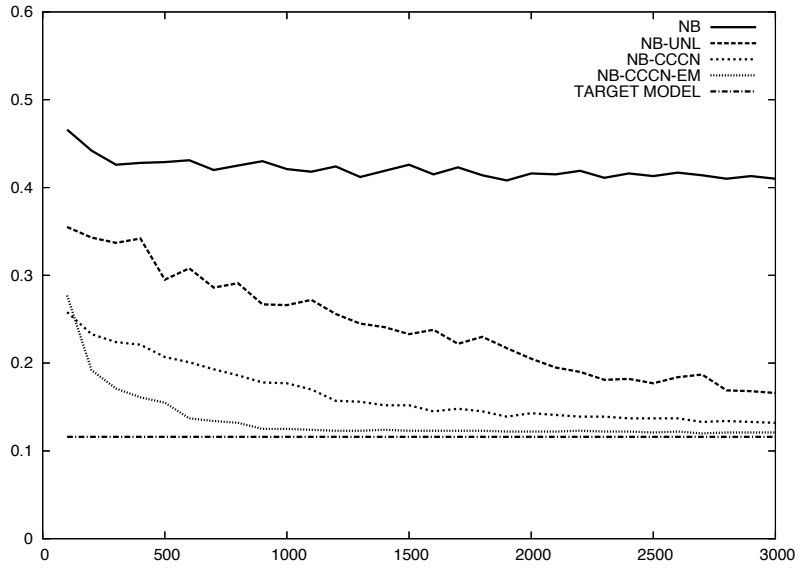


Figure 5.3. Moyennes des erreurs empiriques calculées sur l'ensemble S_{test} des classifieurs appris par les algorithmes NB, NB-CCCN, NB-CCCN-EM, NB-UNL et du modèle cible θ_c en fonction de la taille de l'ensemble d'apprentissage (vue graphique de la Table 5.1) .

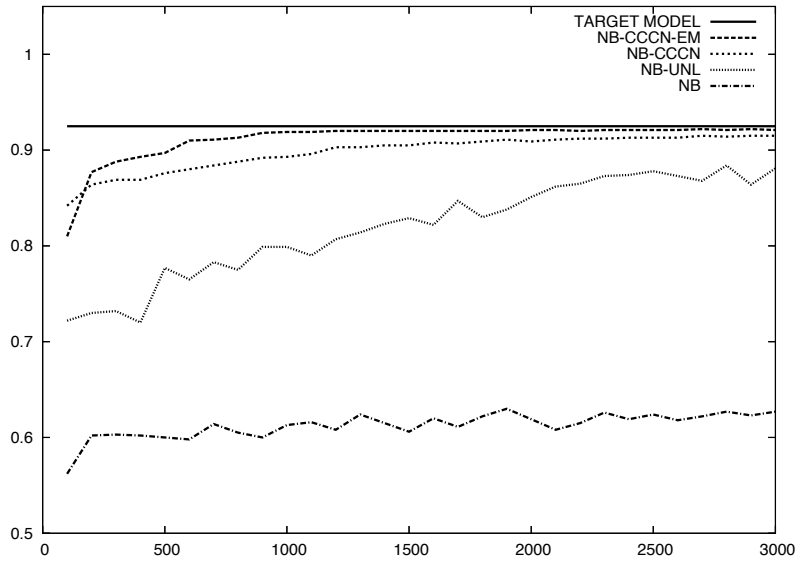


Figure 5.4. Moyennes des F-scores, calculés sur l'ensemble S_{test}, des classifieurs appris par les algorithmes NB, NB-CCCN, NB-CCCN-EM, NB-UNL et du modèle cible θ_c en fonction de la taille de l'ensemble d'apprentissage (vue graphique de la Table 5.1).

5.3.2 Expériences sur des données issues de l'UCI

Cette section présente une seconde série d'expériences sur des données issues de la base de données de l'UCI [Asuncion and Newman, 2007]. Ces expériences ont pour intérêt principal d'évaluer les performances des algorithmes proposés (`NB-CCCN` et `NB-CCCN-EM`) sur des données réelles ne répondant pas exactement aux conditions pour lesquelles ils ont été proposés afin d'évaluer si ces algorithmes restent performants.

Six jeux de données ont été expérimentés : House Votes, Tic Tac Toe, Hepatitis, Breast Cancer, Breast Cancer Wisconsin et Balance Scale. Une description sommaire de ces ensembles de données est fournie dans la table 5.2. Pour l'ensemble Balance Scale, seules les données classées '`right`' et '`left`' ont été utilisées ; les données classées '`balanced`' ont été supprimées de l'ensemble original pour réduire le nombre de classes à 2.

S	$\|S\|$	Nb Attributs x^i	$\|\mathcal{X}^i\|$
House Votes	433	16	2
Tic Tac Toe	958	9	3
Hepatitis	155	19	2-10
Breast Cancer	286	9	2-11
B. C. Wisc.	699	9	10
Bal. Scale	576	4	5

Table 5.2. Description sommaire des six ensembles de données issus de l'UCI sur lesquels les expériences ont été menées. La colonne $|S|$ indique la taille de ces ensembles, la colonne suivante donne le nombre d'attributs descriptifs des données et $|\mathcal{X}^i|$ est le nombre de valeurs que peuvent prendre les attributs.

Le protocole expérimental suivi est identique pour chaque ensemble de données :

- Les algorithmes `NB`, `NB-CCCN` et `NB-CCCN-EM` sont tout d'abord testés sur chacun des jeux de données sans les modifier. Les résultats indiqués dans la table 5.3 sont des moyennes calculées sur dix validations croisées 10-folds.

- La même série d'expériences est ensuite menée à la différence qu'à chaque expérience, les ensembles de données d'apprentissage sont corrompus avec un bruit de classification CCCN pour des taux de bruit $\eta^- = 0.2$ et $\eta^+ = 0.5$ (sans modifier la classe des données test pour les raisons expliquées précédemment). Les algorithmes sont alors relancés sur les ensembles modifiés.

Les performances de la règle majoritaire (la classe la plus souvent observée dans les données d'apprentissage est attribuée à toutes les données test) sont également indiquées dans la table 5.3. Elles permettent d'avoir des performances seuil pour une règle de classification de base. De plus, les algorithmes `NB-CCCN` et `NB-CCCN-EM` permettant d'estimer les taux de bruit qui affectent les données, ces estimations sont également indiquées dans la table 5.3. Elles permettent de s'assurer que les algorithmes ont convergé vers un modèle pour lequel les bruits calculés sur les données d'apprentissage avec ce modèle sont proches de ceux qui affectent ces données.

Les résultats obtenus sur les jeux de données House Votes, Hepatitis et Breast Cancer Wisconsin montrent clairement que lorsque du bruit CCCN est ajouté aux données, les algorithmes NB-CCCN et NB-CCCN-EM détectent et éliminent ce bruit efficacement conservant des performances en classification similaires au cas non bruité.

Les résultats obtenus sur l'ensemble Tic Tac Toe et Breast Cancer sont assez proches de ceux de la règle majoritaire, les classifieurs naïfs de Bayes ne semblent pas adaptés à ces jeux de données, avec ou sans bruit ajouté aux données. Néanmoins, avec du bruit CCCN, on pourrait faire pire que la règle majoritaire. Enfin, les résultats obtenus sur l'ensemble Balance Scale montrent que les deux algorithmes NB-CCCN et NB-CCCN-EM obtiennent de moins bonnes performances lorsque du bruit de classification est ajouté aux données d'apprentissage. Toutefois, les performances restent nettement meilleures que celles de la règle majoritaire ou de celles obtenues par l'algorithme NB.

S	Bruit		Classe Maj.	NB	NB-CCCN	NB-CCCN-EM
House Votes	$\eta^- = 0$	acc	0.617	0.904	**0.916**	0.882
	$\eta^+ = 0$	lk	-	-3134	-3035	**-2915**
	$\eta^- = 0.2$	acc	0.383	0.866	**0.900**	0.873
	$\eta^+ = 0.5$	lk	-	-4130	**-3037**	-3041
		$\overrightarrow{\hat{\eta}}$	-	-	$(0.33, 0.58)$	$(0.20, 0.56)$
TicTacToe	$\eta^- = 0$	acc	0.653	**0.697**	0.682	**0.697**
	$\eta^+ = 0$	lk	-	**-8726**	-8854	**-8726**
	$\eta^- = 0.2$	acc	0.347	0.562	**0.664**	0.587
	$\eta^+ = 0.5$	lk	-	-8828	-8818	**-8815**
		$\overrightarrow{\hat{\eta}}$	-	-	$(0.24, 0.62)$	$(0.21, 0.56)$
Hepatitis	$\eta^- = 0$	acc	0.790	0.827	**0.850**	0.770
	$\eta^+ = 0$	lk	-	-1982	-2416	**-1902**
	$\eta^- = 0.2$	acc	0.240	0.590	**0.811**	0.758
	$\eta^+ = 0.5$	lk	-	-2095	-2273	**-1946**
		$\overrightarrow{\hat{\eta}}$	-	-	$(0.25, 0.55)$	$(0.29, 0.45)$
Br. Cancer	$\eta^- = 0$	acc	0.703	0.730	**0.760**	0.718
	$\eta^+ = 0$	lk	-	-2520	-2682	**-2448**
	$\eta^- = 0.2$	acc	0.327	0.581	**0.732**	0.722
	$\eta^+ = 0.5$	lk	-	-2573	-2623	**-2479**
		$\overrightarrow{\hat{\eta}}$	-	-	$(0.19, 0.59)$	$(0.33, 0.56)$
Br. C. Wisc.	$\eta^- = 0$	acc	0.655	0.973	0.972	**0.975**
	$\eta^+ = 0$	lk	-	-7244	-7790	**-7096**
	$\eta^- = 0.2$	acc	0.345	0.964	0.967	**0.974**
	$\eta^+ = 0.5$	lk	-	-9015	-7818	**-7395**
		$\overrightarrow{\hat{\eta}}$	-	-	$(0.02, 0.05)$	$(0.22, 0.50)$
BaL. Scale	$\eta^- = 0$	acc	0.500	**0.994**	0.980	0.993
	$\eta^+ = 0$	lk	-	-3485	**-3445**	-3484
	$\eta^- = 0.2$	acc	0.500	0.743	**0.847**	0.794
	$\eta^+ = 0.5$	lk	-	**-3611**	-3710	**-3611**
		$\overrightarrow{\hat{\eta}}$	-	-	$(0.10, 0.52)$	$(0.06, 0.36)$

Table 5.3. Taux empirique de bonne prédiction (acc), log-vraisemblance (lk) et bruits estimés calculés pour les quatre algorithmes sur les ensembles de données issus de l'UCI sans bruit et avec bruit CCCN ($\eta^- = 0.2$ et $\eta^+ = 0.5$).

5.3.3 Discussion

Les expériences menées dans ce chapitre montrent que lorsque du bruit CCCN est ajouté aux données, les algorithmes `NB-UNL`, `NB-CCCN` et `NB-CCCN-EM` permettent tous les trois de détecter et supprimer le bruit de classification des données et obtiennent des performances proches des performances que l'on obtiendrait sur les mêmes données non corrompues par du bruit de classification. Toutefois, les algorithmes `NB-CCCN` et `NB-CCCN-EM` montrent des performances sur la tâche d'estimation et sur la tâche de classification plus élevées que celles obtenues par l'algorithme `NB-UNL`. Cette conclusion rejoint celle donnée dans les expériences menées dans la section 3.2.4 qui indiquait déjà que ce dernier algorithme nécessitait un grand nombre d'exemples pour obtenir des estimations de bonne qualité.

Lorsque le critère de comparaison est la vraisemblance, l'algorithme `NB-CCCN-EM` obtient de façon évidente de meilleures performances que l'algorithme `NB-CCCN`. En particulier, pour les expériences menées sur des données artificielles, générées de façon à ce qu'elles suivent exactement le modèle CCCN tel que défini au début de ce chapitre, cet algorithme obtient donc de meilleures performances en classification que l'algorithme `NB-CCCN` puisque le principe de maximisation de la vraisemblance est une bonne heuristique dans cette situation. Toutefois, pour les expériences menées sur des données issues de l'UCI, non générées de façon à répondre exactement au modèle attendu, l'algorithme `NB-CCCN` obtient de meilleures performances en classification que l'algorithme `NB-CCCN-EM` pour 5 jeux de données sur 6. Le principe du maximum de vraisemblance semble, sur ces données, ne pas être une heuristique optimale pour chercher un modèle du problème et on voit ici tout l'intérêt et la complémentarité de ces deux algorithmes. Cette étude a fait l'objet de deux publications respectivement dans les conférences ICML 2006 [Denis et al., 2006b] et CAp 2006 [Denis et al., 2006a].

Ces travaux laissent une question ouverte importante : comment évaluer ou comparer les performances de classifieurs lorsqu'il n'est pas possible d'obtenir des données non corrompues par du bruit de classification CCCN ? Nous travaillons actuellement encore sur cette question, encouragés en particulier par un résultat obtenu dans le chapitre suivant, sur l'apprentissage de séparateurs linéaires en contexte CCCN, qui montre qu'un critère de sélection consistant peut être établi pour cette classe de fonctions particulière.

Chapitre 6

Apprentissage de séparateurs linéaires en présence de bruit CCCN

Sommaire

Suite aux travaux menés au chapitre précédent, la question des classes de fonctions qui sont apprenables en présence de bruit CCCN s'est posée. Les classifieurs naïfs de Bayes sont des classifieurs robustes et souvent efficaces, bien que peu expressifs, et leur apprentissage en contexte CCCN est un résultat intéressant. Toutefois, dans l'intérêt de mettre en place le protocole proposé au chapitre 4, nous avons cherché à apprendre d'autres fonctions prédictives.

L'étude présentée en annexe C de ce mémoire montre que, sous certaines hypothèses sur les taux de bruit, les classes de concepts PAC-apprenables avec bruit de classification conditionnel à chaque classe, CCCN, sont les mêmes que les classes de concepts PAC-apprenables avec bruit de classification uniforme CN. Ce résultat théorique connaît actuellement la limitation de l'hypothèse qui conditionne ce résultat sur les valeurs des taux de bruit des modèles CN et CCCN qui doivent être inférieurs à 0.5. Toutefois, il encourage fortement à étudier individuellement les classes de concepts CN-apprenables pour tenter de montrer qu'elles sont apprenables sans cette condition sur les taux de bruit.

La question s'est donc naturellement posée pour le cas des séparateurs linéaires, dont on sait par [Blum et al., 1996] qu'ils sont efficacement PAC-apprenables en présence de bruit de classification CN. Dans ces travaux, les auteurs proposent un algorithme basé sur un pré-traitement des données pour éliminer les *outliers* (points éloignés de la

distribution du reste des exemples) qui pénalisent l'algorithme standard du perceptron (Section 2.3), et qui permet d'apprendre en temps polynomial les séparateurs linéaires à partir de données corrompues par du bruit CN grâce à une version modifiée du perceptron.

Ce résultat est obtenu par une étude théorique complexe, qui s'accompagne d'un algorithme également complexe à mettre en œuvre. Nous montrons ici, qu'une version simplifiée de cet algorithme, qui ne traite pas le problème des *outliers* et non polynomiale, peut être généralisée au cas CCCN et procure un moyen simple et efficace d'apprendre les séparateurs linéaires dans ce contexte d'apprentissage. Les résultats présentés ici ne généralisent donc pas le résultat donné dans [Blum et al., 1996] – nous ne montrons pas que les séparateurs linéaires sont efficacement PAC-apprenables en contexte CCCN – mais généralisent les résultats donnés dans ces travaux concernant la possibilité d'obtenir un bon vecteur de mise à jour de l'algorithme du perceptron directement à partir des données, en calculant une estimation de la somme des exemples mal classés par le perceptron courant de l'algorithme (expliqué en section préliminaire 2.3.2).

Suite à ces travaux, une implémentation de cet algorithme dans le logiciel `NoTALAP` (Annexe B), développé dans le cadre de cette thèse, a été effectuée. Quelques expériences sur des données artificielles sont décrites dans les sections 6.1.2 et 6.2.2 de ce chapitre, principalement destinées à illustrer les résultats théoriques présentés dans les sections qui les précédent. Cet algorithme a été choisi pour expérimenter le protocole de détection d'affinités locales impliquées dans la formation de contacts entre acides aminés d'une protéine (Chapitre 7).

6.1 Premier cas : les bruits η^+ et η^- sont connus

Nous montrons dans cette première partie du chapitre que lorsque les taux de bruit η^+ et η^- du modèle CCCN sont connus, alors on peut obtenir un estimateur consistant de la somme des exemples mal classés, relativement à leur classe originale et non celle observée, par tout hyperplan. Cette somme constitue un bon vecteur de mise à jour de l'algorithme du perceptron (Section 2.3.2). Dans [Blum et al., 1996], les auteurs montrent comment estimer directement cette somme sur des données corrompues par du bruit de classification CN lorsque le taux de bruit η qui corrompt les données est connu. Nous montrons ici que l'on peut également estimer ce vecteur dans le cas d'un bruit CCCN.

6.1.1 Calcul du vecteur de mise à jour de l'algorithme du perceptron

Soit $S = \{(x, l(x)) \mid l(x) = sign(w^* \cdot x)\}$ un ensemble d'apprentissage séparable linéairement, w^* partitionne S en deux sous-ensembles S^+ et S^- tels que S^+ est l'ensemble de données positives ($w^* \cdot x > 0$) de S et S^- est l'ensemble de données négatives ($w^* \cdot x < 0$) de S. De même, l'hyperplan courant w de l'algorithme du perceptron partitionne S en deux sous-ensembles S_+ et S_- tels que $S_+ = \{x \mid w \cdot x > 0\}$ est l'ensemble des données classées + par w et $S_- = \{x \mid w \cdot x < 0\}$ est l'ensemble des données classées − par w. La seconde partition est la seule observation disponible.

Pour obtenir un estimateur du vecteur de mise à jour $\sum l(x_B)x_B$ du perceptron désiré, on définit une troisième partition de S par l'intersection des deux partitions précédentes (Figure 6.1). On obtient alors quatre ensembles : $S^+_+ = S^+ \cap S_+$, $S^+_- = S^+ \cap S_-$, $S^-_- = S^- \cap S_-$ et $S^-_+ = S^- \cap S_+$. Cette partition n'est évidemment pas observable, mais nous montrons que l'on peut obtenir un estimateur consistant de la somme $Sum[S^\alpha_\beta]$ (α, $\beta \in \{+, -\}$) des éléments des ensembles S^α_β ne dépendant que des observations disponibles, de η^+ et η^- et donc obtenir un estimateur du vecteur $\sum l(x_B)x_B = Sum[S^+_-] - Sum[S^-_+]$ avec ces nouvelles notations.

En effet, nous montrons que S_+, η^+ et η^- permettent d'obtenir un estimateur consistant des sommes $Sum[S^+_+]$ et $Sum[S^-_+]$. La même démonstration montre que S_-, η^+ et η^- permettent d'obtenir un estimateur des sommes $Sum[S^+_-]$ et $Sum[S^-_-]$.

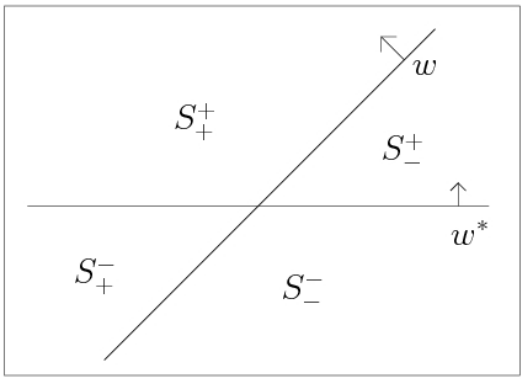


Figure 6.1. Représentation planaire de la partition de l'ensemble d'apprentissage induite par le séparateur optimal w^* et l'hyperplan courant w de l'algorithme du perceptron.

Notons $Pos[S]$ (resp. $Neg[S]$) les exemples d'un ensemble S observés positifs (resp. négatifs) et considérons les sommes $Sum[Pos[S_+]]$ et $Sum[Neg[S_+]]$ des données observées positives et négatives dans S_+, on peut écrire :

$$\begin{cases} Sum[Pos[S_+]] = Sum[Pos[S_+^+]] + Sum[Pos[S_+^-]] \\ Sum[Neg[S_+]] = Sum[Neg[S_+^+]] + Sum[Neg[S_+^-]] \end{cases}$$

Or, l'ensemble S_+^+ (resp. S_+^-) ne contient que des vrais exemples positifs (resp. vrais négatifs). L'hypothèse d'uniformité du bruit CCCN implique donc les égalités :

- $E[Sum[Pos[S_+^+]] = Sum[S_+^+] \cdot (1-\eta^+)$ et $E[Sum[Neg[S_+^+]] = Sum[S_+^+] \cdot \eta^+$
- $E[Sum[Pos[S_+^-]] = Sum[S_+^-] \cdot \eta^-$ et $E[Sum[Neg[S_+^-]] = Sum[S_+^-] \cdot (1-\eta^-)$

où les espérances (notées $E[.]$) considérées sont relatives aux différentes observations que l'on pourrait avoir de S avec le même modèle de bruit CCCN. La valeur de $Sum[S_+^+]$ (resp. $Sum[S_+^-]$) peut donc s'exprimer en fonction des espérances des valeurs de $Sum[Pos[S_+^+]]$, $Sum[Neg[S_+^+]]$ et η^+ (resp. $Sum[Pos[S_+^-]]$, $Sum[Neg[S_+^-]]$ et η^-). En considérant une observation particulière des exemples de S corrompus par du bruit CCCN, ces expressions fournissent des estimations de $Sum[S_+^+]$ et $Sum[S_+^-]$, notées $\widehat{Sum}[S_+^+]$ et $\widehat{Sum}[S_+^-]$, qui permettent de réécrire le système précédent :

$$\begin{cases} Sum[Pos[S_+]] = \widehat{Sum}[S_+^+] \cdot (1-\eta^+) + \widehat{Sum}[S_+^-] \cdot \eta^- \\ Sum[Neg[S_+]] = \widehat{Sum}[S_+^+] \cdot \eta^+ + \widehat{Sum}[S_+^-] \cdot (1-\eta^-) \end{cases}$$

c'est-à-dire $\begin{cases} \widehat{Sum}[S_+^+] = \frac{(1-\eta^-)\cdot Sum[Pos[S_+]] - \eta^- \cdot Sum[Neg[S_+]]}{1-\eta^+-\eta^-} \\ \widehat{Sum}[S_+^-] = \frac{(1-\eta^+)\cdot Sum[Neg[S_+]] - \eta^+ \cdot Sum[Pos[S_+]]}{1-\eta^+-\eta^-} \end{cases}$

après résolution et sous la condition $\eta^+ + \eta^- \neq 1$. De même, on obtient :

$$\begin{cases} \widehat{Sum}[S_-^-] = \frac{(1-\eta^+)\cdot Sum[Neg[S_-]] - \eta^+ \cdot Sum[Pos[S_-]]}{1-\eta^+-\eta^-} \\ \widehat{Sum}[S_-^+] = \frac{(1-\eta^-)\cdot Sum[Pos[S_-]] - \eta^- \cdot Sum[Neg[S_-]]}{1-\eta^+-\eta^-} \end{cases}$$

à partir de S_-, η^+ et η^-. En posant $x_{upd} = \widehat{Sum}[S_-^+] - \widehat{Sum}[S_+^-]$, on obtient un vecteur de mise à jour pour l'algorithme du perceptron qui a la propriété désirée : $E[x_{upd}] = \sum l(x_B)x_B$ et qui s'exprime en fonction des observations, de η^+ et η^- :

$$x_{upd} = \frac{(1-\eta^-)\cdot Sum[Pos[S_-]] - \eta^- \cdot Sum[Neg[S_-]] - (1-\eta^+)\cdot Sum[Neg[S_+]] + \eta^+ \cdot Sum[Pos[S_+]]}{1-\eta^+-\eta^-} \tag{6.1}$$

Nous n'avons pas de borne théorique sur le nombre d'exemples nécessaires pour assurer que cet estimateur garde les bonnes propriétés du vecteur $\sum l(x_B)x_B$ qui garantissent la convergence de l'algorithme. Elle est vraisemblablement fortement dépendante de la distribution sous-jacente des exemples. Néanmoins, les expériences menées laissent penser que c'est un excellent estimateur même pour un faible nombre de données. Nous proposons l'algorithme du Perceptron CCCN (Bruits connus) (Algorithme 10) comme adaptation directe de l'algorithme du perceptron au cas de données sujettes à du bruit CCCN (η^+ et η^- connus). Cet algorithme prend en entrée un ensemble S^η de données bruitées tel que l'ensemble S associé est linéairement séparable, les taux de bruit η^+ et η^-, le paramètre σ, et retourne avec forte probabilité une hypothèse qui sépare les exemples de S.

Algorithme 10 Algorithme du perceptron CCCN (η^+ et η^- connus)

Entrée: $S^\eta = \{(x_1, l^\eta(x_1)), ..., (x_n, l^\eta(x_n))\}$, η^+, η^-, σ

$w = \overrightarrow{0}$, $i = 0$
Tant que $i < \frac{1}{\sigma^2}$ **faire**
  Calculer x_{upd} pour l'hyperplan courant w (voir formule (6.1) Section 6.1.1)
  $w = w + \frac{x_{upd}}{||x_{upd}||}$, $i \leftarrow i + 1$
Fin Tant que

Sortie: w tel que $w \cdot l(x)x > 0 \ \forall x \in S^\eta$ avec forte probabilité

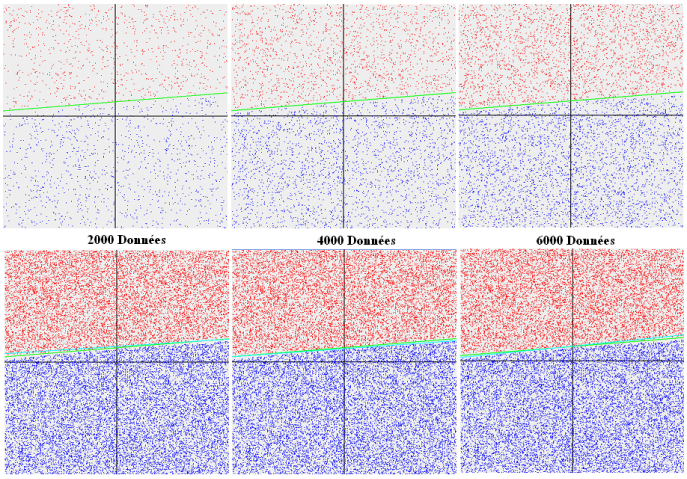
6.1.2 Expérimentation de l'algorithme sur des données artificielles

Cette section présente quelques expériences ponctuelles réalisées sur des données générées artificiellement destinées à illustrer les performances de l'algorithme du perceptron CCCN (Algorithme 10) lorsque les taux de bruit η^+ et η^- qui affectent les données sont connus. Ces expériences sont ponctuelles et réalisées pour des données appartenant à l'espace $\mathbb{R}^2$ de façon à pouvoir illustrer graphiquement les données d'apprentissage, de test et les hyperplans cibles et appris par l'algorithme. Ces graphiques ont été obtenus par l'intermédiaire du logiciel `NoTALAP` (Annexe B) développé durant la thèse.

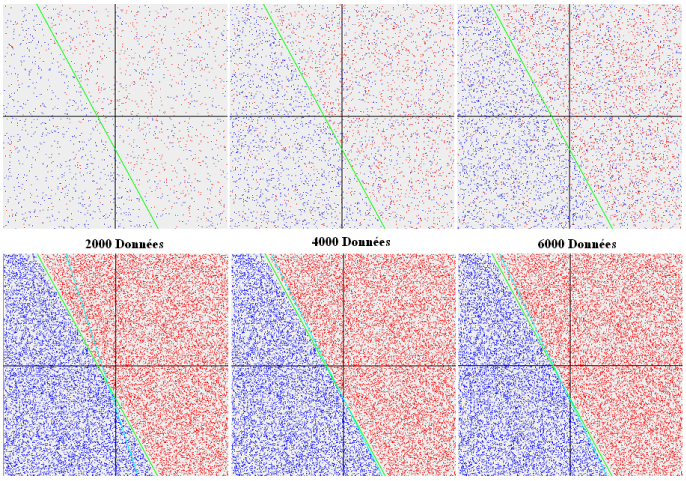
Les exemples de $\mathcal{X} = \mathbb{R}^2$ sont tirés de façon uniforme dans le plan. Les coefficients des hyperplans cibles w^*, qui classent les exemples, sont également tirés aléatoirement. Le paramètre de marge σ est fixé à 1 degré, tous les exemples trop proches des hyperplans cibles sont éliminés des données. Les figures 6.2(a), 6.2(b) et 6.2(c) montrent trois expériences similaires. Pour chacune de ces expériences, le taux de bruit qui affecte les données est fixe ($(\eta^+, \eta^-) = (0, 0)$ dans le premier cas – 6.2(a) –, $(\eta^+, \eta^-) = (0.1, 0.3)$ dans le second – 6.2(b) – et $(\eta^+, \eta^-) = (0.1, 0.6)$ dans le dernier – 6.2(c) –), mais le nombre de données d'apprentissage varie. Dans les trois cas, trois tailles l d'ensembles d'apprentissage sont expérimentées : 2000, 4000 et 6000. Le protocole d'une expérience est le suivant :

- fixer un taux de bruit η^+ et un taux de bruit η^- (indiqués dans la légende des figures)
- générer un hyperplan w^* qui sera utilisé pour classer les données (cible),
- générer l ($l \in \{2000, 4000, 6000\}$) données d'apprentissage de façon à ce qu'aucune ne soit trop proche de w^* (paramètre σ),
- corrompre les données d'apprentissage avec un taux de bruit η^+ pour les exemples positifs et avec un taux de bruit η^- pour les exemples négatifs,
- générer 10000 données test pour évaluer les performances de l'hyperplan appris,
- lancer l'algorithme du perceptron CCCN sur chacun des ensembles d'apprentissage.

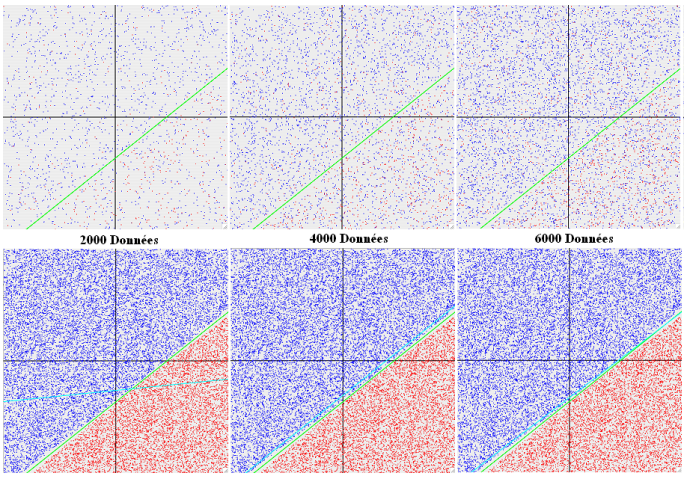
Les figures montrent, pour chaque taux de bruit et pour chaque taille de l'échantillon d'apprentissage, deux images : la première représentant les données d'apprentissage fournies à l'algorithme (les exemples positifs sont colorés en bleu, les négatifs en rouge) ainsi que l'hyperplan cible, et la seconde (en dessous de la première), les données test, l'hyperplan cible et l'hyperplan appris par l'algorithme du perceptron CCCN sur les données d'apprentissage. Aucune série volumineuse d'expériences n'est présentée ici, l'intérêt étant assez mineur : les données suivant le modèle CCCN, l'algorithme fournit de très bonnes performances quand les taux de bruit sont connus.



(a)



(b)



(c)

Figure 6.2. Trois expériences avec (η^+, η^-) (a) $= (0, 0)$ (b) $= (0.1, 0.3)$ (c) $= (0.1, 0.6)$. Pour chaque expérience, les figures situées en haut montrent les ensembles d'apprentissage de taille respective 2000, 4000 et 6000, ainsi que l'hyperplan cible. Celles du bas montrent les données test, l'hyperplan cible (en vert) et l'hyperplan appris (en bleu) à partir des données d'apprentissage décrites sur les figures du haut.

6.2 Cas général : les bruits η^+ et η^- sont inconnus

Nous montrons dans cette section que lorsque les taux de bruit qui affectent les données ne sont pas connus, il est possible d'apprendre des séparateurs linéaires par l'algorithme du perceptron CCCN proposé à la section précédente pour différentes valeurs de ces taux de bruit et de sélectionner de façon consistante une hypothèse parmi celles apprises.

6.2.1 Un critère de sélection consistant en contexte CCCN

Lorsque les bruits η^+ et η^- ne sont pas connus, il est nécessaire de scanner l'intervalle $[0, 1]$ pour les valeurs de η^+ et η^- (en excluant les cas $\eta^+ + \eta^- = 1$), puis de lancer l'algorithme 10 avec chacun de ces taux de bruit en entrée, et enfin de sélectionner un modèle parmi ceux obtenus. Or, l'étude menée en section 5.1.3 montre que la sélection d'un classifieur sur le critère de minimisation du risque empirique n'est pas une méthode consistante en règle générale lorsqu'un bruit CCCN affecte les données. Il est donc indispensable d'établir une autre stratégie pour sélectionner un classifieur.

Nous proposons un critère de sélection, que nous appelons `DPSS`, formé à partir de deux autres critères qui pourraient être utilisés pour sélectionner une hypothèse mais qui, séparément, sont tous les deux inconsistants. Lors du parcours de l'intervalle $[0, 1]$ pour les valeurs de η^+ et η^-, on notera η_i^+ et η_i^- les bruits fournis à l'algorithme 10 et η_{is}^+ et η_{is}^- les bruits calculés sur les données d'apprentissage en utilisant l'hypothèse w_i issue de l'algorithme. Les deux critères que nous utilisons sont alors :

- $\Delta_i = |\eta_i^+ - \eta_{is}^+| + |\eta_i^- - \eta_{is}^-|$,
- $||x_{upd}||$ où l'on considère que x_{upd} est le numérateur de l'expression de x_{upd} (6.1) calculé avec l'hypothèse w_i issue de l'algorithme du perceptron CCCN.

Séparément, ces deux critères ne permettent pas de sélectionner un hyperplan parmi ceux appris par l'algorithme du perceptron CCCN. Par contre, nous montrons que :

Théorème 6.1. *En faisant l'hypothèse qu'avec les bruits cibles η_c^+ et η_c^- en entrée, l'algorithme 10 produit une hypothèse w_c telle que $\Delta_c = 0$ et $||x_{upd}|| = 0$, ce qui correspond à avoir les conditions exactes d'un bruit CCCN sur les données d'apprentissage, nous montrons que le processus de sélection suivant est consistant :*

- *Soient $L^k_{\Delta_i}$ et $L^k_{||x_{upd}||}$ les ensembles des k hyperplans obtenus par l'algorithme 10 qui minimisent respectivement les valeurs de Δ_i et de $||x_{upd}||$.*
- *Initialiser k à 1,*
- *Tant que $L^k_{\Delta_i} \cap L^k_{||x_{upd}||} = \emptyset$, incrémenter k de 1,*
- *Choisir le premier modèle $w \in L^k_{\Delta_i} \cap L^k_{||x_{upd}||}$.*

Nous démontrons que ce processus de sélection est consistant en montrant que tout modèle non obtenu avec les bruits cibles et qui fait des erreurs ne peut avoir des valeurs nulles pour les deux critères en même temps (Δ_i et $||x_{upd}||$ sont complémentaires). Pour lever une ambiguïté entre des modèles indistinguables, nous posons $\eta_c^+ + \eta_c^- < 1$.

Soient η^+ et η^- tels que $\eta^+ \neq \eta_c^+$ et/ou $\eta^- \neq \eta_c^-$, w l'hyperplan produit par l'algorithme 10 avec η^+ et η^- en entrée, Δ^w et $||x_{upd}^w||$ les valeurs des deux critères pour w. Considérons que $\Delta^w = 0$, $||x_{upd}^w|| = 0$ et que w et w^* séparent S de façon différente (sinon w est une bonne solution). On montre que cette situation entraîne soit une contradiction soit que w est une bonne solution.

Considérons l'expression (6.1) de x_{upd} donnée à la section 6.1.1, la condition $||x_{upd}^w|| = 0$ s'écrit :

$$(1-\eta^-)\cdot Sum[Pos[S_-]] - \eta^- \cdot Sum[Neg[S_-]] - (1-\eta^+)\cdot Sum[Neg[S_+]] + \eta^+ \cdot Sum[Pos[S_+]] = 0$$

En introduisant les bruits cibles η_c^+ et η_c^- dans cette expression, on obtient :

$$\begin{aligned}&\big((1-\eta_c^-)\cdot Sum[Pos[S_-]] - \eta_c^- \cdot Sum[Neg[S_-]]\big) + (\eta_c^- - \eta^-)\cdot Sum[S_-] \\ &- \big((1-\eta_c^+)\cdot Sum[Neg[S_+]] - \eta_c^+ \cdot Sum[Pos[S_+]]\big) - (\eta_c^+ - \eta^+)\cdot Sum[S_+] = 0\end{aligned}$$

En utilisant les notations définies en section 6.1.1, cette condition devient :

$$(1-\eta_c^+-\eta_c^-)\cdot Sum[S_-^+] + (\eta_c^- - \eta^-)\cdot Sum[S_-] - (1-\eta_c^+-\eta_c^-)\cdot Sum[S_+^-] - (\eta_c^+ - \eta^+)\cdot Sum[S_+] = 0$$

$$\text{Soit } (\lambda+\alpha)\cdot Sum[S_-^+] - (\lambda+\beta)\cdot Sum[S_+^-] + \alpha \cdot Sum[S_-^-] - \beta \cdot Sum[S_+^+] = 0 \qquad (*)$$

avec $\alpha = (\eta_c^- - \eta^-)$, $\beta = (\eta_c^+ - \eta^+)$ et $\lambda = (1 - \eta_c^+ - \eta_c^-)$ (λ constant et > 0). Nous montrons maintenant que quelle que soit la valeur de α et β, on obtient toujours une contradiction avec les hypothèses faites sur la solution obtenue avec η^+ et η^-.

- **1er cas :** $\alpha > 0$ et $\beta > 0$
  On a supposé que $||x_{upd}^w|| = 0$, la condition (*) est donc vérifiée. Or, avec $\alpha > 0$ et $\beta > 0$ la partie gauche de (*) devient une somme de quatre vecteurs dans le même demi-espace défini par w : $\{x \in \mathbb{R}^m | w \cdot x < 0\}$. La seule solution possible est que $Sum[S_+^+] = Sum[S_-^-] = Sum[S_-^+] = Sum[S_+^-] = 0$, ce qui est impossible.
- **2ème cas :** $\alpha < 0$ et $\beta < 0$
  Pour les mêmes raisons géométriques, $Sum[S_+^+] = Sum[S_-^-] = 0$. (*) devient alors $(\lambda+\alpha)\cdot Sum[S_-^+] - (\lambda+\beta)\cdot Sum[S_+^-] = 0$. Dans ce cas, si $(\lambda+\alpha)$ et $(\lambda+\beta)$ sont de même signe alors on a également $Sum[S_-^+] = Sum[S_+^-] = 0$, ce qui est impossible, sinon, c'est que w classe tous les positifs négatifs et tous les négatifs positifs, w est donc une bonne solution.
- **3ème cas :** $\alpha \geq 0$ et $\beta < 0$ (démonstration analogue pour $\alpha > 0$ et $\beta \leq 0$, $\alpha \leq 0$ et $\beta > 0$ et $\alpha < 0$ et $\beta \geq 0$)
  Pour ce cas, c'est le critère Δ_i qui mène à la contradiction. Soient η_s^+ et η_s^- les bruits calculés sur S avec w : $\eta_s^+ = \frac{n_+^+ \cdot \eta_c^+ + n_+^- \cdot (1-\eta_c^-)}{n_+^+ + n_+^-}$ et $\eta_s^- = \frac{n_-^- \cdot \eta_c^- + n_-^+ \cdot (1-\eta_c^+)}{n_-^- + n_-^+}$ (avec $n_+^+ = |S_+^+|$, $n_-^+ = |S_-^+|$, $n_+^- = |S_+^-|$et $n_-^- = |S_-^-|$). Si $\Delta_w = 0$, alors $\eta_s^+ = \eta^+$ et $\eta_s^- = \eta^-$. On peut alors écrire les deux égalités précédentes : $(\eta^+ - \eta_c^+)\cdot n_+^+ = n_+^- \cdot (1 - \eta_c^- - \eta^+)$ et $(\eta^- - \eta_c^-)\cdot n_-^- = n_-^+ \cdot (1 - \eta_c^+ - \eta^-)$. Comme $\beta = (\eta_c^+ - \eta^+) < 0$, alors $\eta_c^- + \eta^+ < 1$ (soit $\eta_c^+ + \eta_c^- < 1$ en additionnant les deux égalités). Et comme $\alpha = (\eta_c^- - \eta^-) \geq 0$, alors $\eta_c^+ + \eta^- \geq 1$ (soit $\eta_c^+ + \eta_c^- \geq 1$ en additionnant les deux égalités). D'où la contradiction : $\eta_c^+ + \eta_c^- < 1$ et $\eta_c^+ + \eta_c^- \geq 1$.

Le critère proposé est donc consistant : le critère $||x_{upd}||$ assure la sélection d'une bonne hypothèse dans les deux premiers cas, Δ_i dans le troisième. Nous notons ce critère `DPSS`. Les hypothèses imposées peuvent être relâchées. En effet, la même démonstration prouve la consistance du critère lorsque $\eta_c^+ + \eta_c^- > 1$ ($\lambda < 0$). De plus, si les conditions exactes d'un bruit CCCN ne sont pas respectées sur un jeu de données ($\Delta_c > 0$ et $||x_{upd}|| > 0$), ce critère reste très robuste du fait de la complémentarité des deux critères utilisés. Ce résultat est très important pour le problème de la sélection de modèle lorsqu'un bruit CCCN affecte les données car c'est un critère utilisable en pratique sur des jeux de données réelles.

La question se pose naturellement de savoir si ce résultat est généralisable à d'autres classes de fonctions que les séparateurs linéaires. Nous n'avons actuellement pas de réponse à cette question. Toutefois, c'est un premier exemple de critère de sélection consistant en contexte CCCN qui ouvre une perspective très intéressante pour ce problème contraignant, engendré par la présence de bruit CCCN.

6.2.2 Expérimentation du critère de sélection

Cette section a pour objectif d'illustrer le processus de sélection d'un hyperplan en contexte CCCN proposé dans la section précédente. Nous avons fait le choix de présenter deux exemples d'expériences sous forme d'illustrations car elles montrent très clairement les faits exposés dans la section précédente : la complémentarité des deux critères Δ_i et $||x_{upd}||$, la qualité du critère proposé `DPSS`, robuste, même lorsque les hypothèses sur lesquelles repose la démonstration de sa consistance ne sont pas vérifiées.

La description donnée ici des images présentes dans les figures 6.3 et 6.4 est indispensable à leur compréhension. De même, une version couleur de ce document est requise pour la lecture de ces images.

Deux expériences indépendantes ont été menées sur des données artificielles. Ces deux expériences sont menées de façon similaire à la différence des taux de bruit qui affectent les données : $(\eta_c^+, \eta_c^-) = (0.1, 0.3)$ dans le premier cas (Figure 6.3) et $(\eta_c^+, \eta_c^-) = (0.2, 0.6)$ dans le second (Figure 6.4). Nous donnons maintenant la description de chaque expérience.

Une expérience consiste à tirer aléatoirement les coefficients d'un hyperplan w^* de $\mathbb{R}^2$ qui servira d'hyperplan cible durant toute l'expérience. Trois ensembles de données d'apprentissage sont alors générés aléatoirement (distribués uniformément dans le plan), ils contiennent respectivement 500, 2000 et 5000 données. Ces ensembles sont classés avec l'hyperplan w^* puis un bruit de classification CCCN est ajouté à ces données (les taux de bruit étant identiques pour les trois ensembles de données). De même, un ensemble de données test de taille 10000 est généré mais aucun bruit n'est ajouté pour la lisibilité des figures. Ces trois ensembles d'apprentissage (non représentés sur les figures 6.3 et 6.4) permettent d'apprendre w^*, mais avec un nombre de données qui diffère nettement entre chacun d'eux. Ces trois expériences d'apprentissage sont décrites respectivement par les sous-figures (a), (b) et (c) des figures 6.3 et 6.4.

Pour chacune des deux expériences et pour chacun des ensembles d'apprentissage généré artificiellement, l'algorithme du perceptron CCCN est lancé environ 2500 fois pour des valeurs de η^+ et η^- en entrées différentes (les taux de bruit parcourent

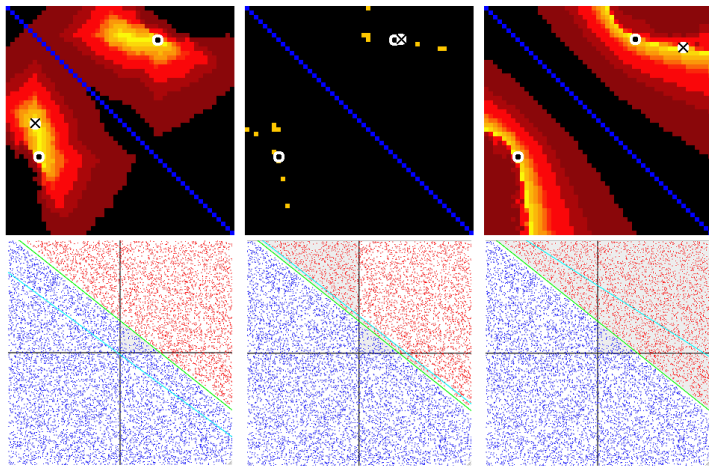
l'intervalle [0,1] par pas de 0.02 en excluant les cas $\eta^+ + \eta^- = 1$). La possibilité que l'algorithme converge vers le modèle complémentaire à la cible (vers un hyperplan w qui classe tous les positifs négativement et inversement) est laissée puisque c'est également une bonne solution (dans ce cas, les taux de bruit valent $\eta^+ = 1-\eta_c^-$ et $\eta^- = 1-\eta_c^+$).

Pour chacun des 2500 hyperplans appris par l'algorithme du perceptron CCCN sur un des ensembles d'apprentissage, les valeurs des trois critères de sélection Δ_i, `DPSS` et $||x_{upd}||$ sont calculées sur l'ensemble d'apprentissage. Les 3 matrices avec des diagonales bleues, situées en haut des groupes de 6 images, sont telles que les abscisses sont les valeurs du taux de bruit η^+ testées, les ordonnées sont les valeurs des taux de bruit η^- testées, et les couleurs représentent les valeurs calculées pour ces trois critères (de gauche à droite : Δ_i, `DPSS` et $||x_{upd}||$) groupées par tranches de valeurs. Le code de couleur utilisé est tel que plus la valeur du critère est petite, plus la couleur utilisée est claire. Seule exception, le critère `DPSS` (situé au milieu), pour lequel les seuls points colorés sont les points des ensembles $L^k_{\Delta_i}$ et $L^k_{||x_{upd}||}$ décrits dans la section précédente, pour k tel que $L^k_{\Delta_i} \cap L^k_{||x_{upd}||} \neq \emptyset$.

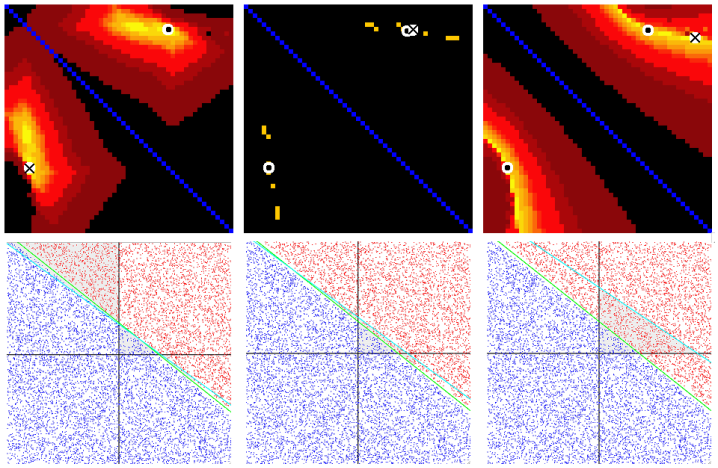
Sur ces mêmes dessins, les taux de bruit cibles η_c^+ et η_c^- (et leurs complémentaires) sont indiqués par des cercles blancs avec un point noir au milieu. Pour chaque critère, les taux de bruit qui ont permis d'apprendre l'hyperplan w qui minimise ce critère sont indiqués par des cercles blancs avec une croix noire au milieu. Les 3 figures en dessous de ces dessins montrent les données test, l'hyperplan cible (en vert) et l'hyperplan (en bleu) qui minimise le critère correspondant.

Par exemple, prenons la sous-figure 6.3(c) : les taux de bruit η^+ et η^- qui permettent d'apprendre l'hyperplan pour lequel le critère `DPSS` est minimal sont exactement les taux de bruit du modèle complémentaire qui a généré les données ($\eta^+ = 1 - \eta_c^-$ et $\eta^- = 1 - \eta_c^+$). L'hyperplan correspondant (image du bas au centre) confirme que c'est un choix consistant d'hyperplan. Toutes ces figures ont été obtenues par l'intermédiaire du logiciel `NoTALAP` développé dans le cadre de cette thèse (description en annexe B).

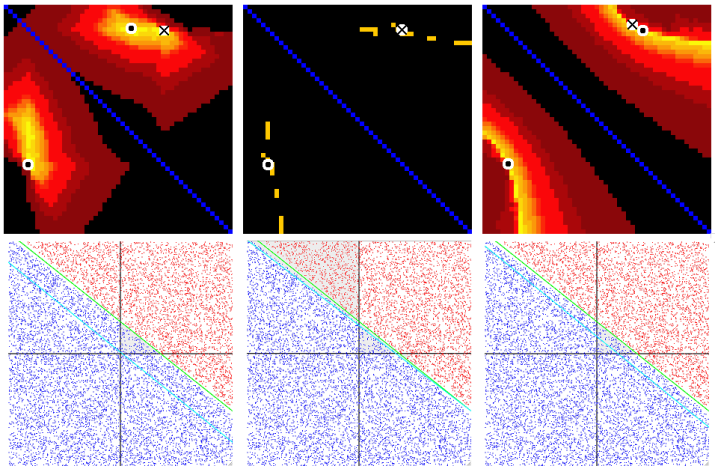
On remarque rapidement sur ces figures la complémentarité des deux critères Δ_i et $||x_{upd}||$. De même, on constate facilement leur inconsistance individuelle : les hyperplans sélectionnés avec ces critères sont rarement des hyperplans proches de la cible. Par contre, ces figures permettent de constater que les bruits qui minimisent le critère `DPSS` sont proches, voire identiques (le rond disparaît, remplacé par la croix) aux bruits cibles, et que les hyperplans sélectionnés par ce critère sont toujours meilleurs que les hyperplans sélectionnés par les autres critères. Tout ceci illustre donc bien l'étude menée précédemment sur ces critères de sélection.



(a)

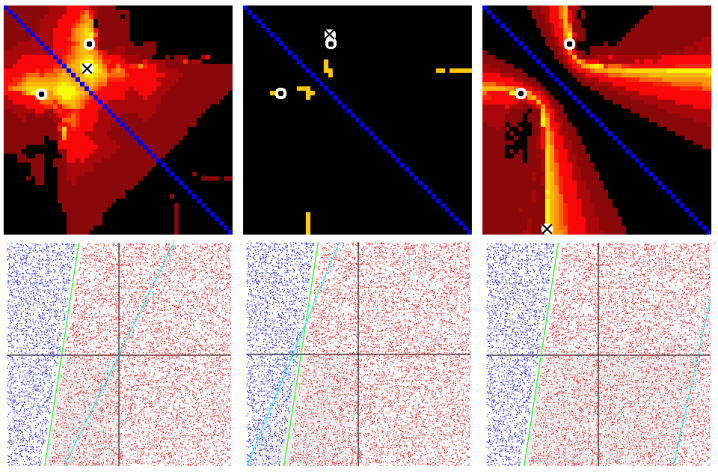


(b)

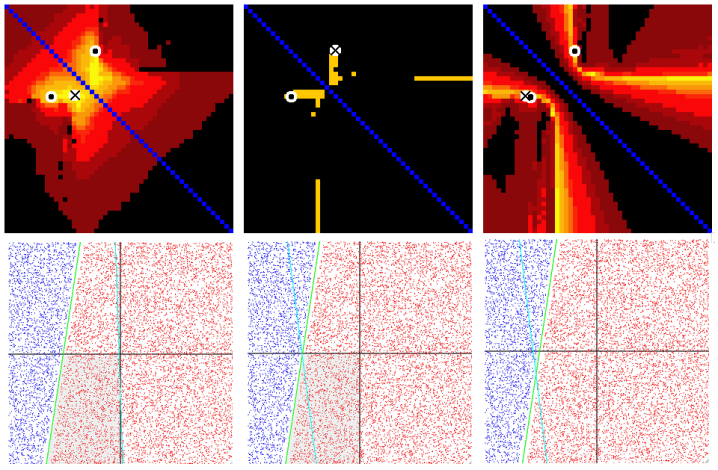


(c)

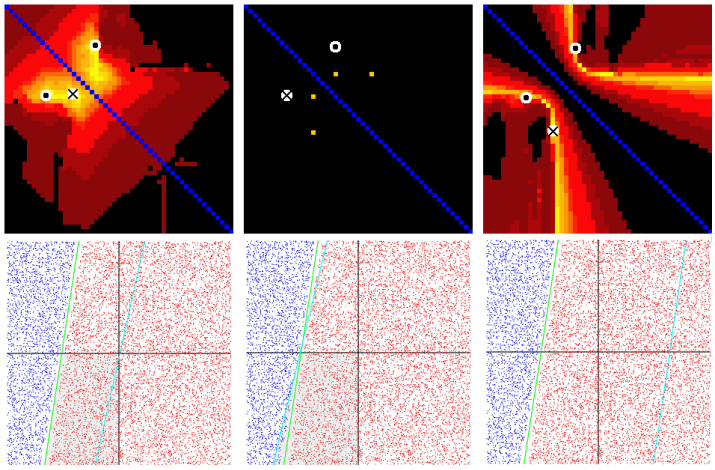
Figure 6.3. Trois exemples de sélection d'hyperplan. Les taux de bruit cibles sont $(\eta_c^+, \eta_c^-) = (0.1, 0.3)$. La taille des ensembles d'apprentissage est de (a) 500 données (b) 2000 données (c) 5000 données. Les trois critères de sélection illustrés sont (de gauche à droite) : Δ_i, `DPSS` et $||x_{upd}||$. La légende des figures est donnée en section 6.2.2.



(a)



(b)



(c)

Figure 6.4. Trois exemples de sélection d'hyperplan. Les taux de bruit cibles sont $(\eta_c^+, \eta_c^-) = (0.2, 0.6)$. La taille des ensembles d'apprentissage est de (a) 500 données (b) 2000 données (c) 5000 données. Les trois critères de sélection illustrés sont (de gauche à droite) : Δ_i, `DPSS` et $||x_{upd}||$. La légende des figures est donnée en section 6.2.2.

Chapitre 7

Expérimentation du protocole

Sommaire

Le dernier chapitre de ce mémoire présente la première expérimentation du protocole de détection d'une information locale impliquée dans la formation de contacts dans les protéines (Chapitre 4) à partir de l'implémentation Java de ce protocole, appelée NoTALAP (Annexe B), réalisée durant la thèse. Ces expériences considèrent deux contacts ponctuels distincts entre deux résidus distants sur la séquence d'une protéine : les ponts disulfures (Section 1.2.1) et les ponts salins (Section 1.2.2).

Les deux études menées précédemment (Chapitres 5 et 6) montrent que l'on peut apprendre efficacement les classifieurs naïfs de Bayes et les séparateurs linéaires à partir de données corrompues par du bruit de classification CCCN. Elles permettent donc de mener une première expérimentation du protocole sur des protéines dont la structure est connue. Toutefois, les expériences menées dans le chapitre 3.3 sur les protéines contenant des ponts disulfures ont montré que les classifieurs naïfs de Bayes utilisés lors de ces premières expériences n'ont obtenu des performances que très moyennes sur ce problème. C'est la raison pour laquelle nous menons les expériences présentées dans ce chapitre avec l'algorithme du perceptron proposé au chapitre précédent (Chapitre 6). Cet algorithme apprend des séparateurs linéaires en contexte CCCN. Si la fonction d'affinité recherchée est de cette forme, ou proche de cette forme, on a alors toute raison de penser que cet algorithme permet d'apprendre cette fonction.

7.1 Protocole expérimental

Les expériences présentées dans la section suivante ont été menées en suivant le protocole expérimental décrit dans cette section. Il est très proche de celui utilisé lors des expériences menées sur les ponts disulfures en section 3.3 et également très proche des protocoles suivis dans les travaux de la littérature sur la prédiction des ponts disulfures.

7.1.1 Jeux de données

Deux ensembles de données ont été expérimentés. Le premier contient des protéines dont on connaît les ponts disulfures et le deuxième des protéines dont on connaît les ponts salins. Pour les raisons données en section 3.3 (page 87) de ce mémoire, seules les protéines contenant de deux à cinq ponts sont considérées. Ces deux ensembles sont :

- `SPX`, un ensemble de ponts disulfures observés expérimentalement dans des protéines. Une description détaillée de cet ensemble se trouve en annexe A de ce mémoire. Il a été élaboré par les auteurs des travaux [Baldi et al., 2005, Cheng et al., 2006]. La table 7.1 indique le nombre de protéines à n ponts de cet ensemble ($2 \leq n \leq 5$). Le taux d'homologie des protéines de cet ensemble est inférieur ou égal à 30%. Cet ensemble est plus conséquent que celui utilisé dans nos expériences précédentes, `G3D-SS`. Il a été rendu public récemment et permet donc d'avoir un ensemble de données plus important, permettant une meilleure représentativité que le jeu initial dont nous disposions. Notons que les ensembles `G3D-SS` et `SPX` ne peuvent être fusionnés car ils contiennent une quantité non négligeable de protéines identiques.

- `G3D-SA`, un ensemble de ponts salins observés expérimentalement dans des protéines. Cet ensemble a été élaboré pour les besoins du groupe de travail de l'ACI Genoto3D (Section 1.3). Une description de cet ensemble est donnée en annexe A. La table 7.2 en rappelle les caractéristiques principales. Le taux d'homologie des protéines de cet ensemble est inférieur ou égal à 25%.

	Nombre de protéines	Nombre de ponts
2 ponts disulfures	211	422
3 ponts disulfures	219	657
4 ponts disulfures	88	352
5 ponts disulfures	49	245

Table 7.1. Statistiques de l'ensemble `SPX` (ponts disulfures)

	Nombre de protéines	Nombre de ponts
2 ponts salins	182	364
3 ponts salins	166	498
4 ponts salins	136	544
5 ponts salins	86	430

Table 7.2. Statistiques de l'ensemble `G3D-SA` (ponts salins)

7.1.2 Codage des paires d'environnements locaux

Le protocole pour extraire les paires d'environnements locaux des protéines et pour coder ces paires d'environnements locaux est décrit en section 3.3.1 (page 88) de ce mémoire. Le codage qui a permis d'obtenir les résultats présentés dans ce chapitre est le codage croisé des paires d'environnements locaux (page 88) et le rayon des fenêtres centrées sur les acides aminés impliqués dans un pont qui a permis d'obtenir les meilleurs résultats est $r = 6$. Ce rayon est identique à celui qui avait permis d'obtenir les meilleurs résultats dans les expériences menées en section 3.3 dans le cadre de la prédiction de l'appariement des cystéines oxydées d'une protéine. Il est également proche ou égal aux rayons ayant permis d'obtenir les résultats optimaux dans les travaux [Fariselli and Casadio, 2001, Vullo and Frasconi, 2004, Ferrè and Clote, 2005b, Cheng et al., 2006].

Toutefois, à la différence des expériences menées dans le chapitre 3, deux alphabets distincts sont considérés ici :

- La premier est l'alphabet naturel des acides aminés, noté Σ^{nat} (avec $|\Sigma^{nat}| = 20$), augmenté d'un caractère supplémentaire, noté X qui signale un acide aminé non déterminé expérimentalement ou un bout de protéine atteint. L'alphabet est $\Sigma = \Sigma^{nat} \cup \{X\}$ et contient 21 caractères. Dans ce cas, une paire d'environnements locaux (W, W') est modélisée par un vecteur de taille 231 (nombre de paires non ordonnées sur un alphabet à 21 lettres) pour lequel chaque coordonnée indique le nombre d'occurrences de la paire d'acides aminés correspondante dans (W, W').
- Le second alphabet permet de réduire la dimension de l'espace de représentation des paires d'environnements locaux en utilisant des clusters naturels d'acides aminés. On considère dans cet alphabet uniquement 7 lettres (plus le caractère X), représentant non plus l'acide aminé mais une propriété physico-chimique de l'acide aminé. Les 7 catégories considérées sont les *aliphatiques*, les *aromatiques*, les *alcools*, les *soufrés*, les *amides*, les *acides* et les *basiques*. Ainsi, l'alphabet contient 8 lettres et la dimension de l'espace de représentation des paires d'environnements locaux passe de 231 à 36.

7.1.3 Valeurs attendues lorsqu'aucune information locale n'est détectée

La table 7.3 indique la probabilité qu'une paire soit appariée sans considérer de fonction d'affinité g. Il est trivial que si les fonctions g apprises sont telles que $P(B|g = 1) = P(B|g = -1) = P(B)$ alors g ne possède pas les caractéristiques d'une fonction d'affinité et aucune information locale impliquée dans la formation des ponts n'a été détectée. En effet, on attend d'une fonction d'affinité d'être telle que la probabilité $P(B|g = 1)$ soit significativement supérieure à la probabilité $P(B|g = -1)$.

Nombre de ponts	P(B)
2	0.3333
3	0.2000
4	0.1429
5	0.1111

Table 7.3. Probabilités $P(B)$ d'observer une paire appariée dans une protéine contenant $2n$ résidus appariés deux à deux ($P(B) = 1/(2n - 1)$).

7.2 Résultats expérimentaux

Cette section présente les résultats obtenus par deux séries d'expériences menées respectivement sur les données ponts salins de l'ensemble `G3D-SA` et sur les données ponts disulfures de l'ensemble `SPX` (décrits en annexe A).

Trois critères sont reportés dans les tables données dans cette section :

1. $P(g{=}1)$, les probabilités que les paires d'environnements locaux aient un haut niveau d'affinité, calculées avec les fonctions linéaires obtenues par l'algorithme du perceptron CCCN présenté dans le chapitre 6,

2. $P(B|g{=}1)$, la probabilité d'observer un pont entre les résidus centraux de deux environnements locaux sachant que cette paire est prédite comme ayant un haut niveau d'affinité par la fonction g apprise,

3. $P(B|g{=}{-}1)$ la probabilité d'observer un pont entre les résidus centraux de deux environnements locaux sachant que cette paire est prédite comme ayant un bas niveau d'affinité par la fonction g apprise.

Remarquons que les taux de bruit CCCN induits par les fonctions d'affinités se dérivent directement de ces deux derniers critères. Le taux η^+ de bruit positif induit par g vaut $\eta^+ = 1 - P(B|g = 1)$ et le bruit négatif $\eta^- = P(B|g = -1)$.

Chaque résultat reporté dans les tables est une moyenne calculée sur cinq cross-validations 10-fold sur les protéines des ensembles `SPX` (et `G3D-SA`) qui contiennent n ponts disulfures (et salins) pour n allant de 2 à 5. Les écarts-types sont indiqués en italique dans les tables. L'expérimentation du protocole sur les données biologiques nécessite un temps d'exécution assez long : environ 3 à 4 jours pour toutes les expériences sur une catégorie de ponts et l'alphabet naturel des acides aminés, et environ une journée pour l'alphabet restreint présenté dans le protocole de ces expériences (Processeur Pentium 4 à 3 GHz). C'est la raison pour laquelle pas plus de cinq cross-validations ont été effectuées.

Enfin, il est important de s'assurer que l'information locale détectée, lorsque cela est le cas, n'est pas corrélée à la présence de brins β ou d'hélices α. A cette fin, nous avons lancé des expériences complémentaires pour vérifier que les prédictions des fonctions d'affinité g apprises (haut ou bas niveau d'affinité) n'étaient pas corrélées à une potentielle implication dans un brin β ou une hélice α des paires d'environnements locaux qui ont servi à les apprendre. Aussi, pour chaque fonction g, les fréquences de réponses de g pour les paires d'environnements locaux dont au moins un est impliqué dans un brin β ou une hélice α, ont été comparées aux mêmes fréquences calculées sur les paires d'environnements locaux non impliqués dans un élément de structure secondaire. Ces expériences (non reportées) montrent que les prédictions de toutes les fonctions d'affinité g calculées dans ces expériences ne sont aucunement corrélées à la présence d'éléments structuraux secondaires.

7.2.1 Résultats obtenus sur les données ponts salins

Nous présentons maintenant les résultats obtenus sur les protéines contenant des ponts salins (ensemble `G3D-SA`). Les tables 7.4 et 7.5 indiquent les caractéristiques des fonctions g calculées à partir des données respectivement en utilisant l'alphabet naturel des acides aminés et en utilisant l'alphabet de clusters décrit en section 7.1.2. La figure 7.1 propose une illustration des résultats présentés dans la table 7.4 qui permet de les visualiser. La section 7.2.3 propose une discussion sur ces résultats qui montrent qu'une information locale est clairement détectée : les paires d'environnements locaux prédites comme ayant un haut niveau d'affinité ont une probabilité plus grande d'être appariées par leurs résidus centraux que les paires prédites comme ayant un faible niveau d'affinité.

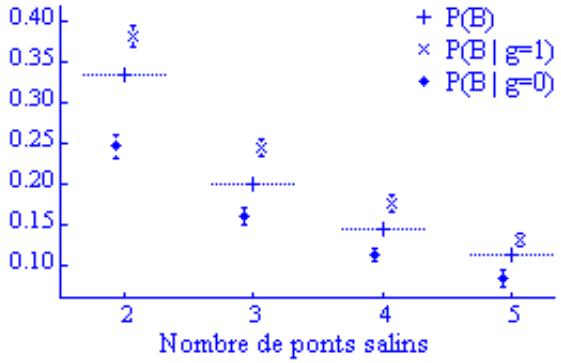


Figure 7.1. Vue graphique des résultats indiqués dans la table 7.4.

Nombre de ponts salins	P(g=1)	P(B|g=1)	P(B|g=0)
2	0.649 ± *0.037*	**0.381** ± *0.009*	**0.246** ± *0.012*
3	0.485 ± *0.068*	**0.243** ± *0.005*	**0.160** ± *0.007*
4	0.482 ± *0.061*	**0.174** ± *0.005*	**0.114** ± *0.003*
5	0.593 ± *0.047*	**0.129** ± *0.003*	**0.084** ± *0.005*

Table 7.4. Caractéristiques des fonctions d'affinité g calculées par l'algorithme du perceptron CCCN sur l'ensemble de protéines `G3D-SA` avec l'alphabet original des acides aminés.

Nombre de ponts salins	P(g=1)	P(B|g=1)	P(B|g=0)
2	0.527 ± *0.057*	**0.385** ± *0.012*	**0.275** ± *0.012*
3	0.417 ± *0.124*	**0.232** ± *0.009*	**0.177** ± *0.009*
4	0.678 ± *0.025*	**0.164** ± *0.002*	**0.099** ± *0.006*
5	0.131 ± *0.005*	**0.173** ± *0.008*	**0.102** ± *0.002*

Table 7.5. Caractéristiques des fonctions d'affinité g calculées par l'algorithme du perceptron CCCN sur l'ensemble de protéines `G3D-SA` avec l'alphabet des propriétés physico-chimiques des acides aminés.

7.2.2 Résultats obtenus sur les données ponts disulfures

Nous présentons maintenant les résultats obtenus sur les protéines contenant des ponts disulfures (SPX). Les tables 7.6 (illustrée en Figure 7.2) et 7.7 indiquent les caractéristiques des fonctions g calculées à partir des données respectivement en utilisant l'alphabet naturel des acides aminés et l'alphabet des propriétés physico-chimiques des acides aminés.

A la différence des résultats obtenus pour le contact des ponts salins, les fonctions calculées par l'algorithme du perceptron CCCN sur les protéines contenant des ponts disulfures ne montrent pas les caractéristiques d'une fonction d'affinité : les paires d'environnements locaux prédites comme ayant un haut niveau d'affinité ont des probabilités de former un pont similaires à celles des paires d'environnements locaux prédites comme ayant un bas niveau d'affinité. Ce constat implique qu'aucune information locale n'a été détectée dans ces expériences. Plusieurs explications sont discutées dans la section suivante.

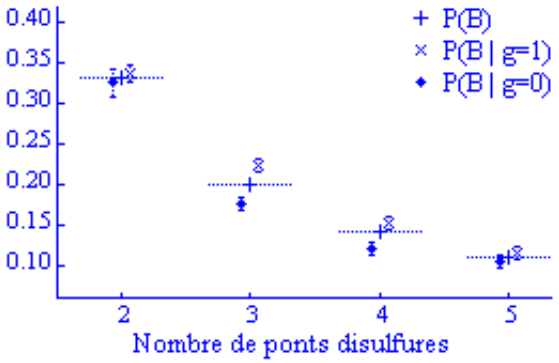


Figure 7.2. Vue graphique des résultats indiqués dans la table 7.6.

Nombre de ponts disulfures	P(g=1)	P(B|g=1)	P(B|g=0)
2	0.622 ± *0.088*	**0.338** ± *0.009*	**0.325** ± *0.018*
3	0.436 ± *0.031*	**0.228** ± *0.005*	**0.179** ± *0.002*
4	0.608 ± *0.087*	**0.154** ± *0.003*	**0.124** ± *0.008*
5	0.528 ± *0.051*	**0.116** ± *0.005*	**0.105** ± *0.005*

Table 7.6. Caractéristiques des fonctions d'affinité g calculées par l'algorithme du perceptron CCCN sur l'ensemble de protéines SPX avec l'alphabet original des acides aminés.

Nombre de ponts disulfures	P(g=1)	P(B|g=1)	P(B|g=0)
2	0.442 ± *0.049*	**0.333** ± *0.006*	**0.334** ± *0.005*
3	0.557 ± *0.043*	**0.218** ± *0.002*	**0.177** ± *0.004*
4	0.617 ± *0.040*	**0.148** ± *0.003*	**0.135** ± *0.005*
5	0.596 ± *0.061*	**0.115** ± *0.001*	**0.105** ± *0.003*

Table 7.7. Caractéristiques des fonctions d'affinité g calculées par l'algorithme du perceptron CCCN sur l'ensemble de protéines SPX avec l'alphabet des propriétés physico-chimiques des acides aminés.

7.2.3 Discussion

Les résultats obtenus sur les ponts salins révèlent qu'un signal clair est détecté : il existe une information locale impliquée dans la formation de ponts salins. Ce signal est constant au travers des différentes expériences (écarts-types petits en comparaison des différences entre les probabilités $P(B|g = 1)$ et $P(B|g = -1)$. La figure 7.1 montre que quel que soit le nombre de ponts considéré, l'algorithme apprend une fonction d'affinité g qui classe toujours plus de ponts observés comme ayant un haut niveau d'affinité que comme ayant un faible niveau d'affinité. Les résultats sont probants aussi bien lorsque l'alphabet considéré est celui des acides aminés (Table 7.4) que lorsqu'on utilise des clusters pour réduire la dimension des données (Table 7.5). Ce dernier point indique que l'information est conservée en ne considérant que les propriétés physico-chimiques des acides aminés, ce qui constitue une information intéressante aussi bien pour le problème biologique considéré que pour la réduction de dimension qu'il permet.

En d'autres termes, nous avons extrait une fonction d'affinité entre les environnements locaux des résidus impliqués dans des ponts salins. Les affinités détectées peuvent très certainement s'expliquer par la nature ionique des ponts salins, dont l'environnement local est certainement impliqué par la charge qu'ils apportent, mais également par les propriétés hydrophiliques des résidus qui entourent les ponts salins (généralement en surface de la protéine).

Toutefois, les résultats obtenus sur les données correspondant aux ponts disulfures (Figure 7.2, Tables 7.6 et 7.7) sont loin d'être aussi positifs. Aucun signal n'est détecté malgré les nombreux paramétrages des données testés (différents rayons r des fenêtres ont été testés, et de même, les trois codages simple, double et croisé des paires d'environnements locaux ont été testés). Aucune fonction apprise ne possède les caractéristiques désirées d'une fonction d'affinité. Les probabilités $P(B|g = 1)$ et $P(B|g = -1)$ sont très proches des probabilités de base $P(B)$ (Table 7.3).

Bien qu'il ne nous soit pas possible, suite à ces expériences, d'expliquer les raisons de ces résultats, plusieurs hypothèses peuvent toutefois être avancées :

1. **Réalité biologique**. La première raison qui peut expliquer ce résultat est qu'il est possible que les environnements locaux des cystéines oxydées ne jouent aucun rôle dans leur appariement. Cette explication est partagée par certains biologistes et biochimistes du domaine : les ponts disulfures sont des liaisons tellement fortes que les affinités entre leurs environnements locaux peuvent ne pas contraindre la formation de ces ponts. Notons toutefois que si cette explication est partagée par certains biologistes, elle remet en cause la validité des travaux de la littérature sur la prédiction des ponts disulfures (Section 1.2.1).
2. **Données sparses**. L'algorithme du perceptron peut souffrir de la dimension dans laquelle les données sont décrites si les données sont sparses. En effet, cet algorithme n'assure aucune optimalité (maximisation de la marge entre l'hyperplan appris et les données) dans la sélection d'une hypothèse, à la différence des SVM *soft-margin* par exemple (en contexte non bruité toutefois). Les vecteurs considérés ici (pour l'alphabet naturel des acides aminés) sont de taille 231 et beaucoup de coordonnées sont nulles. Pour s'assurer que les résultats obtenus ne sont pas dus à cette propriété

indésirable du perceptron, nous avons optimisé l'hyperplan w appris par cet algorithme en appliquant l'algorithme des SVM *soft-margin* (logiciel libre SVM^{Light}) sur les données d'apprentissage qui ont permis d'apprendre w mais classées par w. L'optimisation de la marge n'apporte aucune amélioration notable dans les résultats, la sparsité des données ne semble donc pas en être responsable.

3. **Une classe de séparateurs insuffisamment expressive**. Il est naturel de penser que la classe de fonctions des séparateurs linéaires est trop peu expressive pour trouver ne serait-ce qu'une approximation de la fonction g recherchée. Dans ce cas, il faut chercher dans d'autres classes de fonctions plus expressives que celles-ci, et développer de nouvelles méthodes qui permettent d'apprendre ces fonctions en contexte CCCN. C'est évidemment une des perspectives de cette thèse.

Les expériences menées et décrites dans ce chapitre ne permettent néanmoins pas de savoir quelle hypothèse est la plus probable. Le cas des ponts disulfures reste donc un problème ouvert.

Les résultats obtenus lors des expériences sur les protéines avec des ponts salins apportent une validation expérimentale du protocole proposé lors de nos travaux : il permet de détecter une information locale impliquée dans la formation de contacts entre deux résidus. Le protocole, ainsi que les résultats pour les ponts salins et les ponts disulfures, ont été validés par la communauté scientifique et publiés dans les actes de la conférence internationale BIBM [Magnan et al., 2007a].

Indépendamment du type de contact, nous pensons qu'il nous faut maintenant tenter d'appliquer des méthodes de sélection de variables pour diminuer la taille de l'espace de représentation et peut-être enlever une forme de bruit dans les données. Il n'est pas impensable que pour le cas des ponts salins, ne considérer que les paires d'acides aminés dont au moins un est chargé (dans les contextes locaux) puisse améliorer les résultats obtenus sans sélection de variables dans nos expériences.

De même, la possibilité de réunir toutes les protéines, sans les séparer par le nombre de ponts qu'elles contiennent, comme c'est actuellement le cas dans tous les travaux sur la prédiction des ponts disulfures ou dans les travaux de cette thèse, est étudiée et, si cette possibilité existe, cela pourrait également entraîner de nouveaux travaux. La raison principale qui motive cette séparation est que, pour chaque nombre n de ponts, les taux de bruit qui affectent les données sont différents : c'est une conséquence immédiate de la formalisation d'une fonction d'affinité que nous avons proposée, identique quel que soit le nombre de ponts dans la protéine, mais dont l'observation indirecte, grâce aux paires appariées et non appariées dont on dispose, varie selon le nombre de ponts. Or, les algorithmes CCCN proposés durant cette thèse considèrent que les taux de bruit qui affectent les données d'apprentissage sont inconnus mais fixes.

Enfin, l'extension de la méthode du perceptron CCCN aux machines à noyaux serait également un résultat important pour le protocole que nous proposons. Il y a fort à penser que si on peut apprendre des séparateurs non linéaires en présence de bruit CCCN et s'il existe une fonction noyau adaptée à nos données, alors une fonction d'affinité g, quand elle existe, pourra être apprise efficacement.

Conclusions et perspectives

Nos premiers travaux sur la prédiction de ponts disulfures nous ont rapidement entraînés à rendre centrale la question de l'existence d'une information locale impliquée dans la formation de contacts entre deux acides aminés distants sur la séquence d'une protéine. L'étude formelle menée sur cette notion d'affinité entre segments d'une protéine est novatrice et a permis d'établir un protocole basé sur des méthodes d'apprentissage pour répondre à cette question. Le contexte d'apprentissage CCCN induit par cette étude est peu fréquent dans la littérature, bien que ce soit une généralisation directe du contexte CN bien connu des chercheurs du domaine.

Les études menées dans ce cadre d'apprentissage ont permis de montrer que la présence de bruit de classification conditionnel à chaque classe était une vraie contrainte puisqu'en règle générale, elle rend le problème mal posé. Toutefois, elles ont également permis de montrer qu'en se restreignant à certaines classes de fonctions particulières, telles que les distributions produits ou dans un cadre déterministe avec les séparateurs linéaires, le problème peut alors se résoudre de façon efficace. Les différentes méthodes proposées dans cette thèse ont toutes été expérimentées sur des données artificielles comme réelles et ont montré une véritable efficacité à apprendre des classifieurs performants malgré la présence de forts taux de bruit. En particulier, les travaux sur les classifieurs naïfs de Bayes menés dans cette thèse confirment les bonnes propriétés qui leur sont attribuées dans la littérature du domaine : ce sont des classifieurs simples, facilement et rapidement calculables, qui s'adaptent sans trop de difficulté à différents contextes d'apprentissage.

Toutefois, les performances des algorithmes proposés durant ces travaux ne peuvent, actuellement, être attestées que par des données non corrompues par du bruit de classification, l'inconsistance du principe de minimisation du risque empirique rendant complexe l'évaluation et la comparaison des classifieurs. Malgré l'obtention d'un critère de sélection consistant dans le cas des séparateurs linéaires, la question de savoir si ce résultat peut se généraliser ou s'il existe un critère consistant quelle que soit la classe de fonctions étudiée reste ouverte et constitue une des premières perspectives de cette thèse.

Nous pensons également que le résultat de l'apprentissage de séparateurs linéaires en contexte CCCN doit pouvoir être étendu de façon à intégrer des fonctions noyaux, ce qui permettrait ainsi l'apprentissage de séparateurs non linéaires en contexte CCCN. Cette perspective a un double enjeu : ce serait évidemment un apport important pour le domaine de l'apprentissage automatique puisque la méthode des SVM est fortement pénalisée par la présence de tels taux de bruit, mais cela permettrait également de chercher des fonctions d'affinités impliquées dans la formation de contacts entre acides aminés d'une protéine dans des espaces de fonctions moins restrictifs que les séparateurs linéaires par exemple.

Néanmoins, l'expérimentation du protocole sur les données du jeu `G3D-SA`, qui décrit les ponts salins d'un ensemble de protéines, avec l'algorithme du perceptron CCCN proposé dans nos travaux, a d'ores et déjà donné des résultats probants, montrant qu'une information locale impliquée dans la formation de ces ponts est clairement détectée. Ces résultats fournissent une validation expérimentale du protocole et montrent que la formation de ponts salins est influencée par l'entourage des acides aminés concernés. Nous pensons qu'on doit pouvoir intégrer cette information pour le problème d'apprentissage de la prédiction des ponts salins comme attribut descriptif des paires d'acides aminés. Nous savons maintenant que cette information est pertinente.

Le cas particulier des ponts disulfures reste une question ouverte. N'y a-t-il pas d'information locale impliquée dans la formation de ces ponts comme le pensent certains biologistes ? Est-ce que cette information existe mais n'est pas approximable par un séparateur linéaire ? Peut-être que la réponse est proche de ces deux hypothèses : cette information pourrait exister mais ne pas s'exprimer suffisamment pour influencer la formation de ces liaisons qui, rappelons-le, sont beaucoup plus fortes que les ponts salins. Les expériences menées dans nos travaux ne permettent toutefois pas de savoir quelle hypothèse est la plus probable et il faut maintenant développer d'autres méthodes dans le contexte d'apprentissage CCCN pour espérer trouver une réponse à cette question.

Enfin, l'implémentation du protocole de détection d'affinités locales effectuée durant la thèse, appelée NoTALAP, pour *Noise Tolerant Algorithms to detect Local Affinities into Proteins*, permet d'ores et déjà d'appliquer le protocole décrit dans cette thèse. Il a été développé de façon à ce que l'implémentation d'algorithmes tolérants au bruit CCCN puisse se faire indépendamment du protocole biologique et que leur intégration à la plate-forme soit rapide. Toutefois, il reste encore du travail pour le rendre diffusable, notamment concernant la manipulation de ce logiciel, l'optimisation de certaines phases de travail, des protocoles de tests intensifs assurant sa stabilité et une documentation détaillée de son implémentation et du protocole d'ajout d'algorithmes.

Annexe A

Description des jeux de données G3D-SS, G3D-SA et SPX

Cette annexe du mémoire présente une description et des statistiques sur les trois jeux de données biologiques sur lesquels des expériences ont été menées dans cette thèse. Les jeux de données G3D-SS et SPX sont des ensembles de protéines pour lesquelles on connaît les ponts disulfures (Section 1.2.1 page 35) formés à l'intérieur de ces protéines et l'ensemble G3D-SA est un ensemble de protéines pour lequel les ponts salins (Section 1.2.2 page 39) contenus dans ces protéines sont donnés.

Origine des jeux de données

Les jeux de données G3D-SS et G3D-SA ont été constitués par Christophe Geourjon (IBCP, Lyon) pour les besoins du groupe de travail de l'A.C.I. GENOTO3D (Section 1.3 page 40). Ils sont extraits d'un même ensemble de 1688 séquences protéiques annotées de la base de données PDB, dont le taux d'homologie entre les séquences est inférieur à 25%. Le premier jeu de données – G3D-SS – donne les ponts disulfures observés expérimentalement dans ces protéines, 925 ponts au total, répartis sur 334 des 1688 protéines (1354 protéines n'ont aucun pont disulfure). Le deuxième ensemble de données – G3D-SS – indique les ponts salins dans ces protéines, 7594 ponts au total, répartis sur 1293 des 1688 protéines (395 protéines n'ont aucun pont salin).

Le jeu de données SPX est un ensemble de 1018 séquences protéiques dont le taux d'homologie entre les séquences est inférieur ou égal à 30%. Les ponts disulfures observés expérimentalement dans ces protéines sont indiqués, il y en a 2541 au total (toutes les protéines contiennent au moins un pont disulfure). Il a été constitué par Jianlin Cheng (UCI, Californie) pour les travaux présentés dans [Baldi et al., 2005, Cheng et al., 2006]. Ce jeu a été rendu public en 2006, il est actuellement l'ensemble de protéines annotées contenant des ponts disulfures le plus important disponible.

La table A.1 propose une synthèse des caractéristiques indiquées dans cette section sur les trois jeux de données. La section suivante propose quelques statistiques sur ces trois ensembles, comme par exemple la répartition des protéines par la longueur de leur séquence primaire et par le nombre de ponts qu'elles contiennent, les identifiants PDB des protéines sur lesquelles des travaux ont été menés dans cette thèse, etc.

Nom du jeu de données	Nombre de séquences protéiques	Taux d'homologie maximal	Type des ponts indiqués	Nombre de ponts au total	Nombre de séquences avec ponts	Nombre de séquences sans pont
G3D-SS	1688	25%	Disulfure	925	334	1354
G3D-SA	1688	25%	Salin	7594	1293	395
SPX	1018	30%	Disulfure	2541	1018	0

Table A.1. Caractéristiques générales des jeux de données G3D-SS, G3D-SA et SPX.

Statistiques sur les jeux de données

Répartition des protéines par nombre de ponts

Les figures A.1, A.2 et A.3 donnent, respectivement pour chacun des trois jeux de données G3D-SS, G3D-SA et SPX, la répartition des protéines par nombre de ponts qu'elles contiennent. Le nombre de protéines ne contenant pas de ponts n'est pas indiqué sur les diagrammes. Il est de 1354 pour G3D-SS, de 395 pour G3D-SA et de 0 pour SPX.

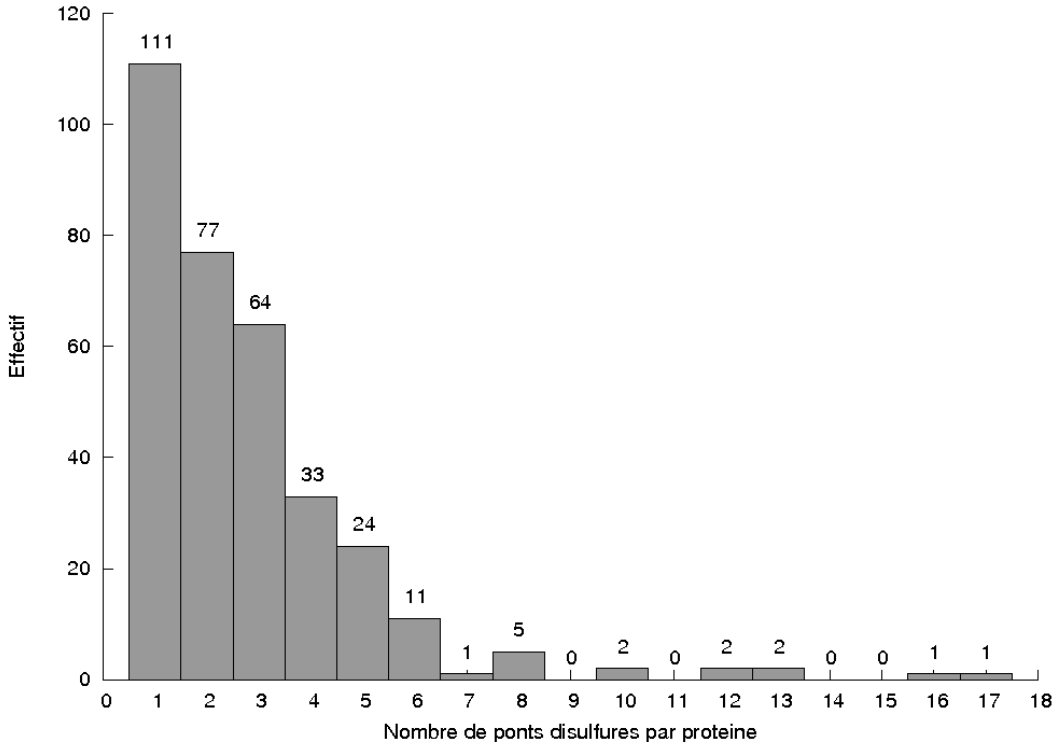


Figure A.1. Répartition des protéines de G3D-SS par nombre $n > 0$ de ponts disulfures qu'elles contiennent. 1354 protéines n'ont aucun pont disulfure ($n = 0$).

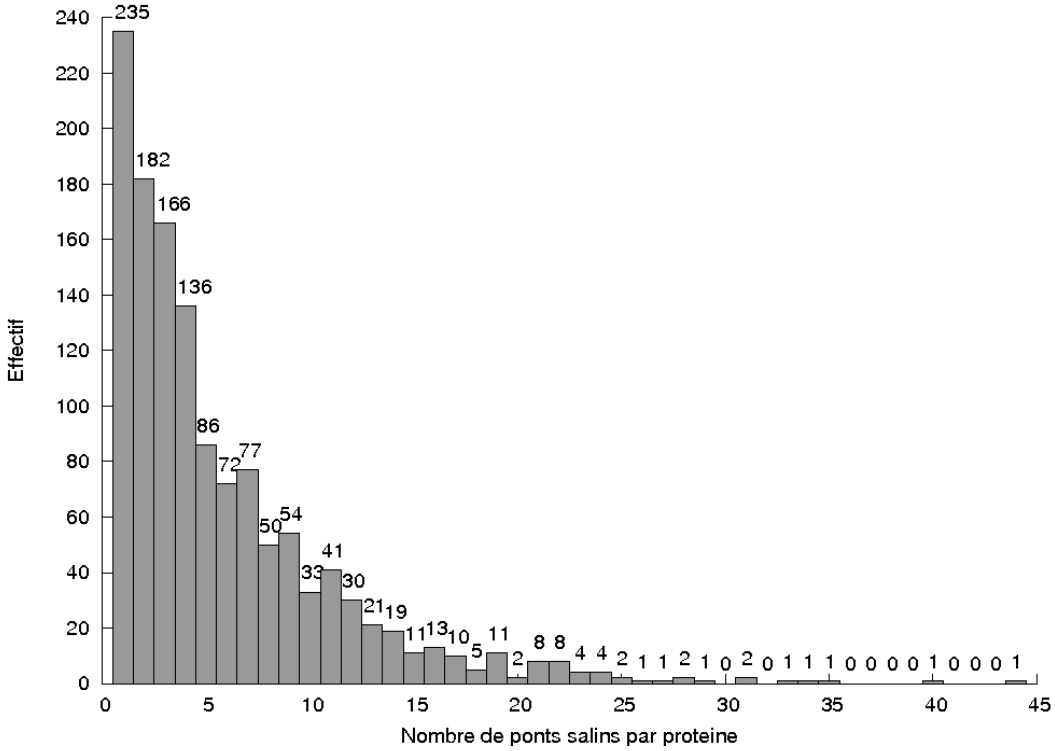


Figure A.2. Répartition des protéines de G3D-SA par nombre $n > 0$ de ponts salins qu'elles contiennent. 395 protéines n'ont aucun pont salin ($n = 0$).

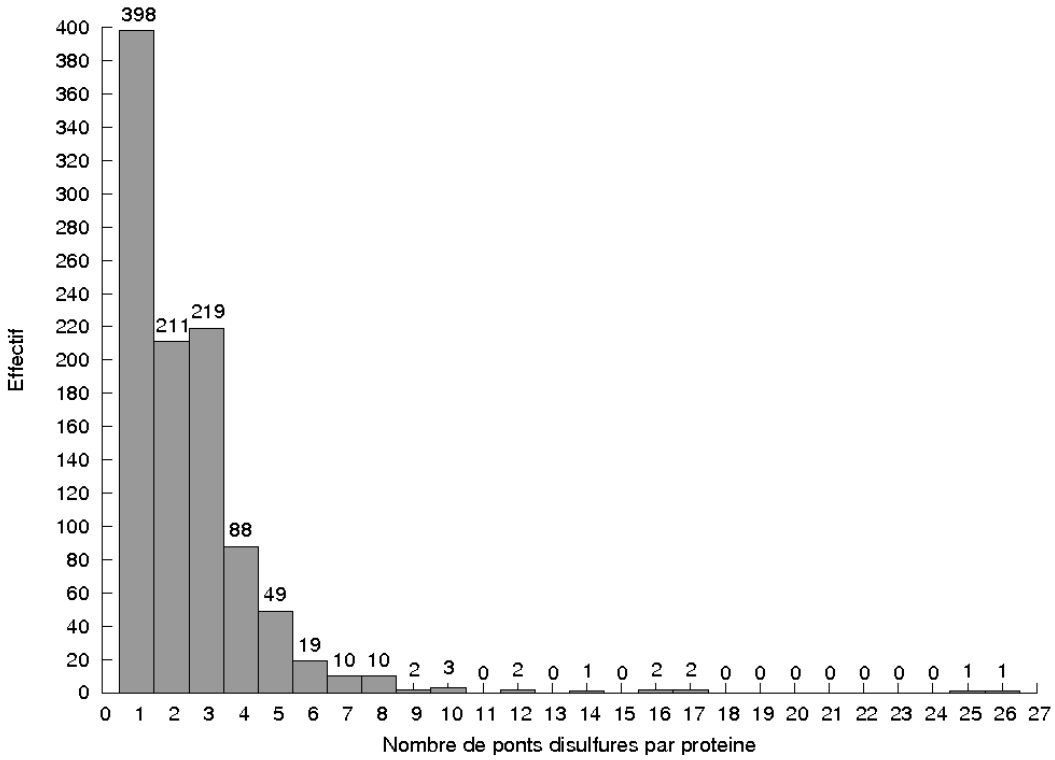


Figure A.3. Répartition des protéines de SPX par nombre $n > 0$ de ponts disulfures qu'elles contiennent.

Répartition des protéines par longueur de séquence primaire

La figure A.4 donne la répartition des 1688 protéines qui constituent les jeux de données G3D-SS et G3D-SA par longueur des séquences primaires de ces protéines regroupées par tranche de 50 acides aminés. De même, la figure A.5 donne la répartition des 1018 protéines contenues dans l'ensemble SPX par longueur des séquences primaires de ces protéines.

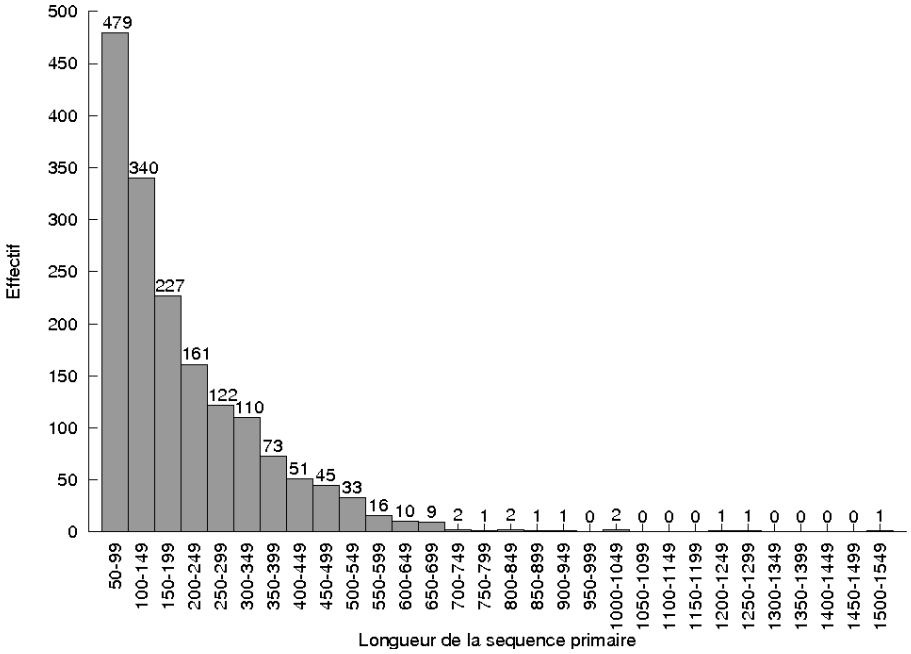


Figure A.4. Répartition des protéines de G3D-SS/SA par longueur des séquences primaires.

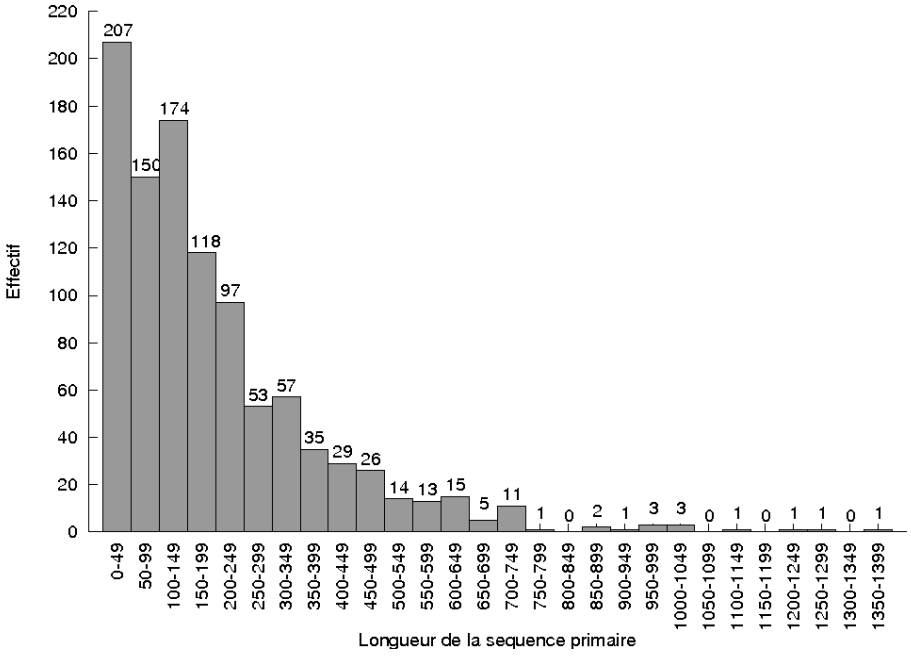


Figure A.5. Répartition des protéines de SPX par longueur des séquences primaires.

Fréquences des 20 acides aminés dans les deux ensembles de protéines

Les figures A.6 et A.7 indiquent les fréquences d'apparition des 20 acides aminés biologiques respectivement dans l'ensemble de protéines annotées dont les jeux de données `G3D-SS` et `G3D-SA` sont issus et dans l'ensemble de protéines annotées `SPX`.

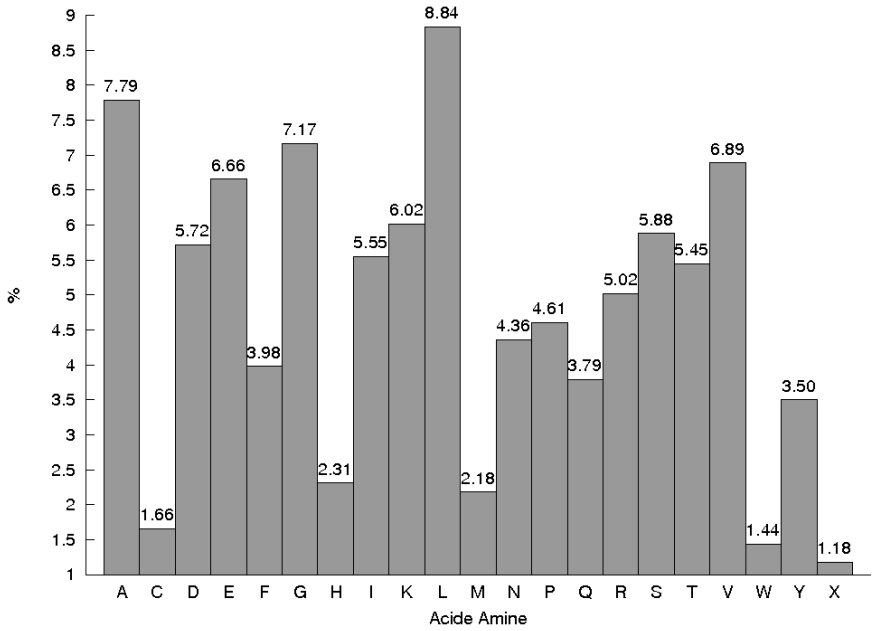


Figure A.6. Fréquences (×100) des 20 acides aminés dans l'ensemble de protéines `G3D-SS/SA`. Les acides aminés non résolus expérimentalement sont notés X

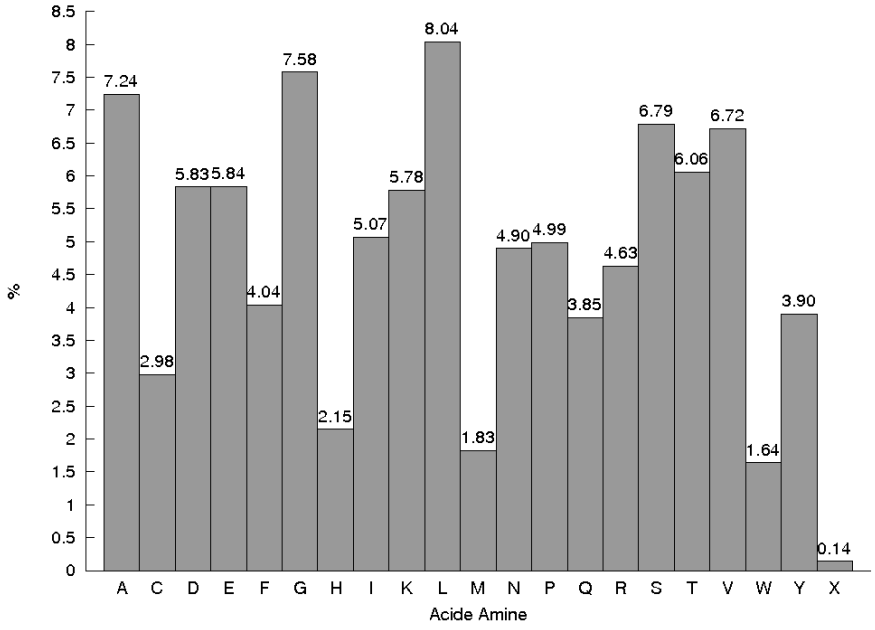


Figure A.7. Fréquences (×100) des 20 acides aminés dans l'ensemble de protéines `SPX`. Les acides aminés non résolus expérimentalement sont notés X

Identifiants PDB des protéines contenant de 2 à 5 ponts

Les expériences menées durant les travaux présentés dans ce mémoire ont été effectuées exclusivement sur les protéines des trois jeux de données qui contiennent de 2 à 5 ponts (disulfures et salins). Les tables A.2, A.3 et A.4 indiquent respectivement les identifiants PDB de ces chaînes protéiques pour les jeux de données G3D-SS, G3D-SA et SPX.

Chaînes protéiques contenant 2 ponts disulfures (77)
1tpn 1e8r 1hic 1bus 1cxw 1hfi 1h0z 1icw 1fgp 1g2t 1l9l 1f9p
1o8r 2hgf 1lyb 1o7c 1jbi 1n5p 1i17 1c3z 1g96 1ji8 1oqc 1spp
1akp 1r5r 1i85 1k3b 1ow4 2gmf 1kp4 1jpy 1exs 1apz 1bgc 1jtg
1i1r 1fnl 1n0s 1hng 1jfu 1qab 1kgy 1dy1 1wbc 1jbj 1seb 1im9
1n9d 1qjj 1bp3 1i9b 1i1c 1tfh 1nbq 1cn4 1jc9 1gy0 1j08 1deo
1dlp 1jp5 1lln 1npe 1kj3 1klf 1dl5 1eag 1nd6 2vsg 1itu 1js8
1ew2 1f8r 1ffr 1fi1 3bta
Chaînes protéiques contenant 3 ponts disulfures (64)
1apq 1p9j 1rfn 1e0f 1mkn 1e9t 1ps2 1bf0 1pce 1den 1ueo 1owt
1igl 1hrf 1hp8 1kjs 1c01 1g91 1n69 1hky 1i8n 1pdg 1agq 1ljp
1kat 1ks6 1hn6 1p57 1jfn 1agi 1bw4 1oo9 1ooh 1hcf 1knm 1a9v
1dqg 1cfe 1gz2 1avg 1n1f 1htn 1a0h 1puo 1a7m 1i4u 1ic1 1prt
1iar 1n6v 1lbu 1n10 1jma 1oql 1exw 1ook 1ese 1by8 1nt4 2ant
1dp4 1p8j 1us2 1hcy
Chaînes protéiques contenant 4 ponts disulfures (33)
1ugl 1ayj 1tih 2brz 1ce3 1iw4 1vgh 1vib 2rel 1kbt 1uoy 1kv0
1bcg 1hyp 1whf 1jtb 1fbr 1m58 1k9i 1qo3 1klx 1qnx 1oup 1fg9
1cnv 1p8t 1fhf 1f42 1ogq 1elv 1nhc 1b82 1ivy
Chaînes protéiques contenant 5 ponts disulfures (24)
1eai 1f94 1bxp 1cou 1es7 1tpg 1dz7 1pco 1bfa 1m8n 1ijx 1poc
1nl1 1g1r 1bht 1fjr 1iyb 1ef7 1ism 1phm 1ihp 1mfv 1prh 1kcw

Table A.2. Identifiant PDB des chaînes protéiques de G3D-SS contenant de 2 à 5 ponts disulfures.

Chaînes protéiques contenant 2 ponts salins (182)
1ptr 1g2c 1l8c 1n7l 1tih 1bq8 1c6v 1j75 1p65 1umq 1pce 1eai 1mm3 1uoy 1neg 1fse 2alc
1or7 1pcf 1qgw 1gl5 1b67 1ehj 1jsu 1dz1 1scj 1jw2 1dan 1ljo 1nkn 1l6l 1awj 1pm3 1h95
2occ 1m1a 1bb9 1cxz 1hb8 1nh9 1kwa 1pch 1hbk 1hbx 1cit 1es7 1pyu 3ncm 1gfw 1b34 1pug
1whp 1ecm 1lpl 1cdz 1ihf 1id3 1agq 1vub 1qsz 1jrm 2e2a 1gxq 1kte 1aan 1ej5 1go3 1b0n
1aa2 1gn0 1jer 1byw 1g96 1ibz 1gh6 1q7s 2cy3 1drm 2gmf 1oko 1cot 1c44 2ezs 1mwb 1k78
1bw4 1jhf 1ulk 1n1d 1nvj 1hcf 1knm 1a9v 1m0v 1gqa 1m49 1c7k 2eif 1jat 1b1a 1h3q 1j3a
1n2f 1avg 1b4a 1lql 1n0e 1f08 1j4s 1qsm 1hzm 1jop 1htn 1iry 1ext 1apz 1qxh 1jkn 1au1
1czr 1lf7 1prs 1cnt 1b87 1kgy 1fsi 1fq1 1nqz 1m06 1jt0 1un2 1ki9 1jf0 1gxd 1eje 1l3a
1m1j 1col 1q87 1doa 1jm1 1fll 1r72 1kzn 1tfh 1i8d 1iyb 1n5l 1nbq 1k6y 1lbu 1cpm 1gri
1ng0 4sbv 1l8o 1olr 1cby 2azo 1avq 1cmx 1a41 1l2j 1bev 1lul 1g60 1jma 1frv 1d8v 1oql
1esq 1exj 1cnv 1by8 1r1w 1ogq 1hqd 1j3i 1eiy 1brl 1rhz 1hkg
Chaînes protéiques contenant 3 ponts salins (166)
1a92 1pfb 1l0o 1cxw 1igl 1ftt 1a0o 1l3x 1l9l 1tif 1qgw 1h75 1eo0 1kvj 1lab 1ibu 1lqm
1cou 1n1j 1ycq 1hfe 1g2d 1oqj 1m1r 1eto 1ln4 1j7c 1lwu 1pnx 1i3o 1bwu 1f37 1fyb 1ep8
1lq9 1hx1 1nze 1dw3 1dyn 1ufl 1r26 1fzp 4rhn 1gef 1py9 1gxc 1elw 1ow4 1r75 1i3q 1kx5
1dfx 1ufb 1rfs 1onl 1q05 1dd4 1pa7 1c52 1klx 1ocu 1poc 1dqg 1lqp 1vh9 1j5e 1gz2 1io4
1lo9 1jb0 1k05 1ax8 1kng 1hbg 1vit 1ly1 1bys 1a8m 1ndl 1qvi 1kex 1a0h 1vhm 1lih 1n62
1ggq 1fnx 1hng 1usf 1qab 1cqq 1h0p 1pkh 1iuh 1wbc 1laj 1xbr 2stv 1exc 1m3s 1feu 1i8a
1k2f 1mjs 1gqp 1iar 1klq 1mbm 1rgp 1bp3 1i9b 1a6i 1kk6 1imj 2xat 3gar 1qnx 1l0b 1bye
1qrl 1jmk 1ooe 1h9t 1qcd 1dgl 1bkj 1qhf 1iru 1p4x 1e3k 1qle 1kc6 1jjo 1koo 1f6y 1lxa
1mhm 1ib0 1o4u 1jvf 1b5t 1exw 1klf 1f02 2hwe 1p8t 1p33 1fts 1gaw 1yst 1nm2 1bia 1elv
1p0k 1lq8 1nzb 1p75 1nhc 1j72 1cli 1eag 1a8y 1a26 1d2p 1inp 1dty
Chaînes protéiques contenant 4 ponts salins (136)
1ba5 1gk6 1d66 1igu 1f2i 1ity 1adr 1oqp 1llm 1q1h 1bh9 1jtb 2hts 1iw7 2rgf 2ubp 1tlk
1nlq 1n5p 1p97 1fc3 1mww 1kp4 1bgf 1mvp 1uen 1pqf 1fzr 1hlv 1p8z 1hh1 1k9i 1q5v 1qo3
1m4j 1kgl 1qyn 1iuk 1q2y 1avo 1h6h 1fqk 1lj9 1jzm 1k5g 1fst 1k66 1afv 1dvo 1k2e 1n05
1n1f 1nyk 1ni0 1d1q 1nn4 1f99 1d8u 1v8c 1i1r 1ntg 1ko1 1bvo 1bht 1pn1 1eyx 1hgx 1bef
1jfu 1l1o 1m4o 1i4u 1tyf 1hh4 1di7 1f7c 1khh 1itv 1im9 1jss 1aer 1iru 1b06 1aug 2fua
1lkv 1fg9 1oro 1bam 1noc 1pvy 1a9c 1v7u 1li1 1oy5 1omi 1deo 1qwt 1h21 1eop 1fgu 1p84
1b5d 1jp5 1e6v 1nnw 1mxm 1a8p 1jr2 1k3r 1dj0 1po2 1nzy 1e79 1qle 1r4x 1q7f 1ulb 1egz
1bou 1iaw 1iow 3csu 1qyd 1p8n 1gdh 1khd 1a99 1ipk 1qax 1jny 2cel 1bp1 1ihm 1mio 1mu2
Chaînes protéiques contenant 5 ponts salins (86)
1owt 1ekz 1d8k 1k5k 1tig 3ygs 1hv2 1nlx 1p57 1ejf 1h4y 1qb3 1mai 1ufy 1is6 1ktg 1md0
1qu1 1srv 1lzw 1nl1 1nig 1lqb 1roc 1goc 1dps 1od6 1i1d 1f9a 1dkg 1bgc 1nwa 1np6 1koh
1fnl 1g63 1ak7 1a3c 1hqv 1i9s 1i3a 1qjj 1je5 1mr2 1h4h 1n34 1kzf 2abk 1ff3 1oup 1cn4
1k6d 1o26 1gy0 1cke 1qr0 1ire 1cjb 1nc3 1qaq 1n10 1jj2 1iy2 1kcf 1npy 1nt0 1hx8 1iho
1ook 1jk0 1p7h 1duw 1kpi 1aln 1uox 1ftr 1p35 1p9o 1q2w 1fhf 1le5 1mdw 1fqv 1ujn 1gga
1m57

Table A.3. Identifiant PDB des chaînes protéiques de G3D-SA contenant de 2 à 5 ponts salins.

Chaînes protéiques contenant 2 ponts disulfures (211)
1bso 1bt3 1b2t 1azh 1ava 1atn 1ast 1apy 1apy 1akg 1afa 1ah1 1a5f 1aal 1a8l 1b50 1b45
1a15 1a0m 1bcp 1bcp 1bhu 1bih 154l 1bj7 1brv 1boy 1bp3 1bo0 1buy 1byf 1c0d 1c3y 1cd1
1cd9 1civ 1cn4 1ct2 1cvi 1cx8 1cz6 1d2u 1d6a 1ddt 1de3 1deo 1dg2 1dl2 1dl5 1dlp 1dr9
1dth 1dy1 1e8p 1e8r 1edn 1eej 1efx 1ehs 1ei0 1eig 1es0 1evs 1ew2 1exz 1ezm 1f62 1f8r
1f9p 1faz 1fcv 1fgx 1fi1 1fic 1foa 1frf 1fus 1fwo 1g12 1g26 1g2s 1g8q 1g9m 1g8t 1g96
1ga3 1gcy 1gna 1gnb 1gxy 1h0l 1h0z 1h4p 1h8u 1ha6 1hcc 1hdl 1hje 1hkf 1hnf 1hp9 1hoe
1hpw 1hqq 1hqq 1hqq 1hqq 1hxl 1hxl 1hxm 1hxz 1hxz 1i17 1i1r 1i8x 1ibl 1icw 1ien 1ieo
1iex 1iko 1il6 1ixk 1j5h 1j7v 1j86 1j8h 1jbi 1jbj 1jfm 1jfu 1jhn 1ji8 1jm1 1uya 1js4
1jtg 1k64 1k0x 1k19 1k8i 1kcg 1kcn 1kco 1kfp 1kgy 1kiu 1kjm 1ksi 1kwd 1l3e 1l3w 1l9l
1m2c 1m4r 1ma2 1mkc 1moe 1mtq 1n02 1n0s 1n5h 1nbq 1not 1nr4 1nun 1v54 1o7b 1o80 1oc5
1od5 1ogs 1ohm 1oig 1oqe 1p9m 1pdc 1pen 1pg1 1pre 1pwa 1v7m 1qft 1qmw 1qrn 1qup 1r4m
1r9i 1rhf 1rj1 1rjt 1rni 1rpb 1sfo 1sfs 1spp 1srb 1ter 1thg 1vdu 1vsg 1wbc 1wct 1xga
1xgb 2cut 2gmf 2hgf 2ilk 2vsg 3ssi
Chaînes protéiques contenant 3 ponts disulfures (219)
1av3 1as5 1apf 1ans 1ahk 1ag7 1ab1 1acw 1a7m 1aap 1b8w 1bbn 1bds 1bei 1bgk 1bh4 1bqs
1bnb 1bkc 1bus 1bw3 1bxm 1by2 1c01 1c39 1c6w 1c4e 1c4k 1cgi 1cix 1clv 1cnn 1cns 1cr8
1cti 1d0d 1d1h 1d2l 1d5q 1d6b 1dec 1dfn 1dkc 1dof 1dp4 1dqg 1dt3 1dtk 1du9 1dwa 1dyk
1e03 1e4s 1e4t 1e87 1eb6 1ed0 1egg 1el0 1emx 1erp 1erc 1erd 1ery 1esc 1ete 1ews 1eyo
1f34 1f3k 1f7m 1fd4 1ut3 1fu3 1fyg 1g1p 1g4f 1g9p 1gai 1gm0 1gw0 1gwb 1uv0 1gz2 1h20
1h5o 1ha9 1hcf 1hcy 1hd6 1hic 1hky 1hn6 1hre 1htn 1hvw 1hvz 1hwh 1hy9 1hyl 1i25 1i26
1i2u 1i4u 1i8n 1iar 1uwy 1ica 1ip0 1irh 1ixt 1j0t 1j5j 1j8e 1jjz 1jlp 1jlz 1jma 1jra
1jrf 1ju8 1k36 1k8i 1k9j 1kj0 1kj6 1kjs 1klt 1kma 1koz 1kth 1l3h 1lbt 1lbu 1ldl 1lmm
1lmr 1lnl 1lqr 1lsg 1lsh 1lup 1m12 1m2s 1mb6 1mcp 1mct 1mcv 1mjv 1mkn 1mm0 1mmc 1mwp
1n10 1n1f 1n69 1n6u 1n9d 1n9e 1nep 1nix 1nkl 1o8r 1of9 1omc 1ooh 1oqd 1ow4 1owt 1p1p
1p9j 1p8b 1p8j 1pb5 1pb8 1pce 1pja 1pjv 1pmc 1pnf 1pnh 1poz 1pqr 1pto 1puo 1q2j 1q2k
1q3j 1qab 1qba 1qcx 1qk6 1qk7 1qz1 1r1f 1r0t 1r5r 1rji 1s2b 1s4n 1sco 1sh1 1sop 1t50
1tcg 1tfx 1tsk 1uap 1ueo 1v9u 1vca 1zxq 2crd 2g3p 2psg 2pta 3bmp 3kiv 3lri
Chaînes protéiques contenant 4 ponts disulfures (88)
1ugl 1apj 1aho 1agg 1afh 1b6e 1b9o 1bbg 1bgp 1brz 1bmr 1bk8 1c55 1chl 1clv 1cn2 1cnv
1cq3 1cvo 1dkm 1dtv 1uoy 1eit 1ewt 1f0i 1f42 1fas 1fbr 1ff4 1fg9 1fjn 1g13 1gkn 1gl4
1gp0 1gps 1h1h 1h30 1h8x 1hq8 1hyp 1i1r 1i6f 1ikp 1iq9 1iqq 1ux6 1ivy 1iw4 1j36 1jkz
1jxc 1jzn 1kp6 1kqh 1ksq 1ktw 1kvz 1l3y 1l6h 1lk9 1lz5 1m4u 1mco 1mpz 1mr4 1myn 1mzd
1n8m 1ne5 1oc0 1oup 1p53 1psy 1qno 1qnx 1qo3 1r2m 1rmg 1tfg 1udk 1vgh 1vib 1vmo 1vtx
1whe 2aaa 2rel
Chaînes protéiques contenant 5 ponts disulfures (49)
1ata 1ao5 1bip 1bs9 1bte 1c9t 1ccv 1cdq 1dix 1dqc 1dz7 1eai 1f94 1fjr 1g1q 1g5g 1gh7
1ha8 1hss 1hty 1hx2 1hyk 1ijx 1imt 1isv 1j2e 1jqp 1kcw 1kg0 1kg1 1kpt 1lcs 1lr7 1lsi
1m8n 1mkf 1mn1 1pcn 1pgr 1poc 1prh 1qfx 1qjs 1r0s 1r46 1sdw 1tpg 1wkt 2afp

Table A.4. Identifiant PDB des chaînes protéiques de SPX contenant de 2 à 5 ponts disulfures.

Annexe B

NoTALAP : implémentation Java du protocole de détection d'affinités locales dans les protéines

Nous donnons dans cette annexe une description sommaire d'une implémentation Java du protocole de détection d'affinités locales dans les protéines (Chap. 4) effectuée durant la dernière année de thèse. Ce logiciel, appelé `NoTALAP` pour *Noise Tolerant Algorithms to detect Local Affinities into Proteins*, a été conçu pour atteindre des objectifs simples :

- expérimenter le protocole proposé dans cette thèse sur des données biologiques,
- expérimenter des algorithmes tolérants au bruit CCCN sur des données artificielles ou réelles (biologiques ou non),
- proposer un protocole d'ajout d'algorithmes CCCN simple et rapide,

tout en assurant que l'implémentation des algorithmes d'apprentissage puisse se faire indépendamment du problème biologique et que, de même, leur expérimentation puisse être possible en dehors de tout contexte biologique.

Les expériences présentées dans les chapitres 6 et 7 ont toutes été menées avec cette plate-forme. De même, les figures illustrant, sur des exemples, le comportement de l'algorithme du perceptron CCCN ont été obtenues avec cette application. Nous en présentons ici quelques caractéristiques et fonctionnalités de façon à en donner un aperçu général. Une documentation détaillée n'est pas encore disponible. Ce logiciel est toujours en phase de développement et de stabilisation.

Les premiers travaux menés durant la thèse ont également nécessité des développements pour expérimenter les différents algorithmes proposés. Ces premiers programmes, conçus en langage C, indépendants les uns des autres, restent à intégrer à `NoTALAP`. C'est un des points qui rend cette plate-forme encore incomplète. Toutefois, sa structure générale est implémentée, ses fonctionnalités principales également ; ne manquent que quelques algorithmes et la mise en place de tests standardisés intensifs pour en assurer la stabilité.

Le langage utilisé : Java

Le choix du langage Java pour développer `NoTALAP` se justifie par une multitude de raisons :

- c'est un langage objet évolué,
- il est portable grâce à la JVM (Machine Virtuelle Java),
- il est riche d'un point de vue des possibilités de développement,
- il est polymorphe,
- il permet d'obtenir facilement des applications client-serveur (applets, servlets),
- il permet le partage de librairies de façon simple,
- il évite de nombreuses réécritures de code,
- il permet d'automatiser des séries de tests unitaires qui assurent l'intégrité du code,
- il permet la création rapide de documentations html des classes,
- il est convivial et manipulable assez facilement,
- etc.

On peut toutefois lui opposer un léger manque d'optimalité dans les performances de calculs en comparaison à des langages de plus bas niveau, comme le langage C.

`NoTALAP` est implémenté avec la version 1.5 de Java (J2SE 1.5) mais est compatible avec la version 1.4 (versions antérieures non testées). La plate-forme de développement utilisée est Eclipse.

Structure du code

`NoTALAP` est actuellement constitué de 96 classes réparties dans 4 packages distincts :

- Le premier package contient les classes des formats de données (vecteurs, hyperplans, modèles, protéines, ponts, etc).

- Le second package contient les classes de manipulation des objets de données (lecture et écriture dans des fichiers pour différents formats, génération artificielle de données, extraction des paires d'environnements locaux, codage des paires sous forme de vecteur, statistiques, gestion du bruit, gestion des modèles appris, etc).

- Le troisième package contient toutes les classes relatives aux algorithmes (les algorithmes, la gestion des exceptions liées à l'utilisation d'algorithmes, les interfaces graphiques disponibles pour ces algorithmes, comme pour le perceptron, etc).

- Le dernier package contient les classes d'expériences qui contrôlent le déroulement de celles-ci (expériences à partir de fichiers, de données artificielles, biologiques ou classiques, expériences ponctuelles ou séries répétitives, synthèse des expériences, contrôle du bon déroulement, etc).

Les sections qui suivent décrivent brièvement ces 4 parties du logiciel ainsi que les fonctionnalités associées. Il faut noter qu'il ne possède pas d'interface graphique, le paramétrage des expériences se faisant directement par un fichier prévu à cet effet.

Les objets de données

Les objets destinés à être principalement utilisés comme support de données comme les données d'apprentissage, les protéines, les ponts ou les clusters d'acides aminés, sont regroupés dans un seul package. Nous ne donnons ici que les définitions des deux objets destinés à contenir des informations sur les protéines : les classes `protein` et `bridge`.

Les attributs de la classe `protein` sont :

```
private String Name;
private String PrimarySequence;
private String SecondarySequence;
private Vector StructuralElements;
```

Le nom de la protéine utilisé est généralement son identifiant PDB et les séquences primaires et secondaires sont considérées comme des chaînes de caractères. Le dernier attribut permet de stocker des éléments structuraux de la protéine, comme les ponts disulfures ou salins, mais également les feuillets β ou les hélices α. Leur type n'est pas spécifié afin que cet attribut soit aussi générique que possible. Les constructeurs de cette classe permettent de considérer tous les cas envisageables sur les champs que l'on voudrait ou non remplir. Toutefois, la séquence primaire est requise dans chacun de ces constructeurs. Les méthodes associées à cette classe ne sont pas reportées ici.

La classe `bridge` est destinée à contenir une liaison entre deux acides aminés d'une même protéine (indépendamment du type de liaison considéré). Ces attributs sont :

```
private char AminoAcid1;
private int AminoAcid1Index;
private char AminoAcid2;
private int AminoAcid2Index;
```

La description de ces champs va de soi : les caractères sont ceux qui représentent les acides aminés appariés (ou tout autre alphabet similaire), et les entiers contiennent la position des acides aminés liés par un pont sur la séquence primaire d'une protéine. Les constructeurs et les méthodes de cette classe ne sont pas décrits ici. Pour l'application du protocole, le dernier attribut de la classe `protein` contient donc un ensemble d'instances de la classe `bridges` : une par pont observé dans la protéine.

La gestion des objets de données et l'exploitation des modèles

Cette section présente les classes qui permettent la gestion des données qui circulent entre les différentes étapes des expériences. Ces classes concernent toute la partie qui précède l'exécution d'un algorithme : récupération de données, génération de données, pré-traitements, traitement des données biologiques, statistiques sur les données, etc. De même, elles concernent toute la partie qui succède à l'exécution d'un algorithme : test du modèle appris, statistiques, évaluation des modèles, comparaison, création de fichiers de synthèse des expériences qui se sont déroulées, etc. Par exemple, pour le protocole de détection d'affinités locales proposé dans cette thèse, ces classes sont responsables de la préparation des données et de l'analyse des modèles appris sur ces données.

Pour la préparation des données, les principales fonctionnalités implémentées sont :

- récupération de données biologiques dans des fichiers. Les protéines, avec ou sans pont, doivent alors être décrites avec le format présenté en section 1.3.2
- récupération de données vectorielles dans des fichiers. Le format imposé est le `csv`
- génération de données vectorielles artificielles (paramétrable)
- génération de protéines artificielles avec et sans pont (paramétrable)
- traitement des données relatif au bruit de classification (ajout de bruit sur les données, etc)
- extraction et codage des paires d'environnements locaux sous forme de données vectorielles (protocole et paramètres décrits en section 3.3.1)
- statistiques sur les données (protéines et données vectorielles)

Pour l'analyse des modèles appris par les algorithmes d'apprentissage, les fonctionnalités disponibles sont :

- statistiques sur les performances des modèles appris par les algorithmes
- caractéristiques des fonctions d'affinités apprises par un algorithme sur les données biologiques. Estimation de leur corrélation à la présence de brins β ou d'hélices α
- synthèses de séries d'expériences (cross-validations, répétitions d'expériences, etc)
- Sauvegarde des modèles et des résultats

Toutes les fonctionnalités nécessaires à l'application du protocole proposé dans cette thèse ou à l'expérimentation d'algorithmes sur des données réelles (biologiques ou non) ou artificielles, sont présentes dans cet ensemble de classes. Elles sont toutes terminées, paramétrables lorsque cela est nécessaire, et tout cela, de façon indépendante aux algorithmes implémentés dans `NoTALAP`.

Les algorithmes d'apprentissage

La partie des algorithmes est totalement indépendante du reste du programme. Des algorithmes tolérants ou non au bruit de classification CCCN peuvent y être placés. La seule contrainte sur leur implémentation est de respecter le format d'entrée : un ou deux ensembles de données (classe `dataset`) où chaque donnée est représentée sous la forme d'un vecteur. De même, le format de l'élément retourné en sortie de l'algorithme doit être un objet héritant de la classe abstraite `model`, et doit impérativement implémenter quelques méthodes comme classer une donnée ou un ensemble de données.

Actuellement, les algorithmes implémentés sont le perceptron, les variantes de cet algorithme tolérantes au bruit de classification CN proposées dans [Bylander, 1994] et [Blum et al., 1996] (pour ce dernier, seule la forme simple que nous avons généralisée au cas CCCN (Chapitre 6)) est implémentée), et la variante de cet algorithme au bruit CCCN que nous avons proposée dans cette thèse. D'autres algorithmes restent à ajouter à la version actuelle, par exemple les différentes versions de l'algorithme naïf de Bayes proposées dans cette thèse. C'est une partie encore inachevée, qui fera l'objet d'un travail prochainement.

La gestion globale des expériences

Un dernier ensemble de classes est responsable de la gestion du déroulement des expériences. Il contient peu de classes, chacune d'entre elles permettant de réaliser une catégorie d'expériences particulière, ou partie d'expérience. En effet, gérer des expériences pour expérimenter un algorithme sur des données artificielles est complètement différent de la gestion de l'application du protocole de détection d'affinités locales dans les protéines. Ces classes ne seront pas décrites plus précisément car elles permettent juste d'assurer la coordination correcte des différentes fonctionnalités présentées précédemment.

Le fonctionnement de `NoTALAP`

Le fonctionnement du logiciel est assez simple. Un fichier de configuration permet de choisir les expériences souhaitées puis de régler tous les paramètres de ces expériences. A la suite de l'exécution de ces expériences, le logiciel propose tout un ensemble de fichiers de sortie placés dans un répertoire, organisés selon le type d'expérience demandée. Ces fichiers contiennent aussi bien la sauvegarde de la configuration qui a permis de réaliser l'expérience, que le détail des expériences : statistiques sur les données, détails sur le déroulement des algorithmes sélectionnés pour la série d'expériences, statistiques sur les modèles appris, etc. Enfin quelques fichiers proposent des synthèses intermédiaires ou globales pour les séries d'expériences qui se sont déroulées.

Aucune interface graphique n'est proposée pour le moment. Il n'est pour l'instant pas prévu que cela soit fait car cela nécessite un temps assez important pour un apport concret discutable. Peut-être cela fera-t-il l'objet d'un projet proposé à des étudiants de troisième ou quatrième année dans un cursus informatique.

Perspectives

Le logiciel `NoTALAP` n'est actuellement pas à un stade suffisamment achevé pour le diffuser à la communauté scientifique. Les priorités pour arriver à cela se situent à trois niveaux distincts. Le premier est l'enrichissement du logiciel : proposer quelques fonctionnalités en plus, celles proposées actuellement étant très corrélées aux besoins des travaux menés dans la présente thèse. Par exemple, l'implémentation d'algorithmes supplémentaires est une des priorités. Le second est l'implémentation de séries de tests unitaires intensives pour chacune des fonctionnalités proposées par le logiciel pour en assurer la stabilité. Enfin, la dernière priorité est la conception d'une documentation technique complète du logiciel ainsi que d'un manuel d'utilisation. La documentation technique existe déjà mais demande un effort supplémentaire pour la compléter et la rendre plus compréhensible, le manuel d'utilisation est à concevoir entièrement.

L'objectif actuel est de terminer la conception de `NoTALAP` rapidement pour en proposer la diffusion. De plus, il y a fort à penser que ce logiciel pourra devenir une base logicielle pour les implémentations qui seront engendrées par la suite des travaux présentés dans cette thèse concernant le problème biologique considéré, notamment l'exploitation des fonctions d'affinités apprises pour la prédiction des ponts.

Annexe C

CN = CPCN

Cette annexe de la thèse présente une étude théorique de l'apprentissage en présence de bruit de classification pour différents modèles de bruit : CN (Section 2.1.4), CCCN (Section 4.3.1) et CPCN (Section 2.1.4) dans le cadre PAC de Valiant [Valiant, 1984] décrit à la section 2.1.4. Nous abordons en particulier la question de l'apprenabilité de classes de concepts sous ces différents modèles de bruit de classification.

Le résultat principal de ce travail est l'égalité entre les ensembles de classes de concepts CN-apprenables, CCCN-apprenables et CPCN-apprenables, sous certaines conditions sur les taux de bruit, résultat que nous notons CN=CCCN=CPCN. Les inclusions CPCN $\subseteq$ CCCN $\subseteq$ CN sont triviales de par la définition de la PAC-apprenabilité avec ces différents modèles de bruit. Nous présentons ici les démonstrations des inclusions inverses.

La première partie de ces travaux montre l'inclusion CN $\subseteq$ CCCN : toute classe de concepts CN-apprenable est également CCCN-apprenable. Ce résultat a été annoncé par Avrim Blum et Tom Mitchell dans [Blum and Mitchell, 1998], mais non démontré. Nous apportons ici la démonstration de ce résultat. L'inclusion inverse étant triviale, on a alors la première égalité du résultat : CN=CCCN. La deuxième partie de ce travail montre l'inclusion CCCN $\subseteq$ CPCN : toute classe de concepts CCCN-apprenable est également CPCN-apprenable. On obtient ainsi le résultat annoncé : CN=CPCN.

La présence de ce résultat comme annexe à la thèse s'explique par différentes raisons :

- la première, et la plus importante, est que ce travail a été principalement effectué par Liva Ralaivola (LIF, Marseille) et François Denis (Directeur de thèse), ma participation personnelle à ce travail étant moindre que celle des deux auteurs principaux de ce travail,

- la seconde raison est que ce résultat est de façon naturelle annexe aux travaux présentés dans cette thèse. En effet, le modèle de bruit CCCN introduit dans les travaux de thèse y est étudié, toutefois ce résultat théorique dans le cadre PAC n'a pas encore permis d'établir de nouvelles méthodes permettant d'expérimenter le protocole de détection d'affinités locales impliquées dans la formation de contacts ponctuels entre résidus distants d'une protéine proposé au chapitre 4.

Néanmoins, ce résultat est important pour les travaux entrepris dans cette thèse. En effet, l'application du protocole décrit au chapitre 4 nécessite des méthodes capables d'apprendre des classifieurs en présence de bruit CCCN sur les données. Le résultat présenté dans cette annexe apporte une information importante sur cette contrainte imposée par le protocole : il autorise à chercher les fonctions d'affinités g dans les classes de concepts CN-apprenables, largement étudiées dans la littérature.

Ces travaux ont été publiés dans les actes des conférences ICML 2006 [Ralaivola et al., 2006b] et CAp 2006 [Ralaivola et al., 2006a].

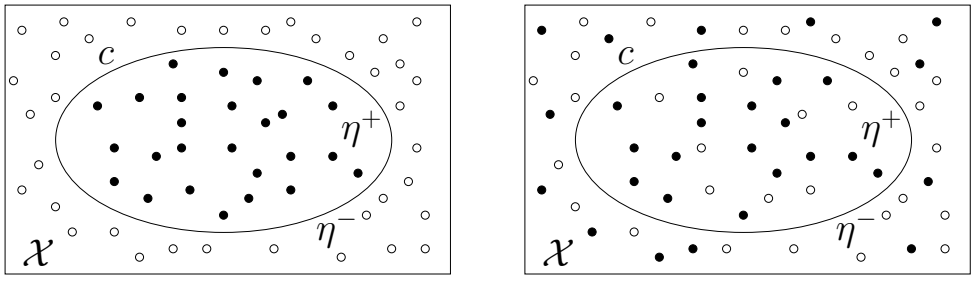
Cette annexe se décompose en deux parties. La première donne une définition formelle de l'apprentissage PAC avec bruit de classification CCCN de façon similaire aux définitions de la CN-apprenabilité et de la CPCN-apprenabilité données en section préliminaire 2.1.4. La seconde présente la démonstration du théorème CN=CPCN sous la forme publiée de ces travaux dans [Ralaivola et al., 2006b] (ICML).

PAC-apprentissage avec bruit CCCN

Cette section présente une définition formelle de l'apprentissage PAC avec bruit de classification CCCN. Les notations utilisées, spécifiques au cadre d'apprentissage PAC, sont présentées à la section 2.1.4. L'apprentissage de classes de concepts avec bruit de classification conditionnel à chaque classe (Figure C.1) considère que l'algorithme d'apprentissage a accès à un oracle $EX^{\overrightarrow{\eta}}_{CCCN}$ défini de la façon suivante :

Définition C.1. *Soit $\overrightarrow{\eta} = [\eta^+ \ \eta^-]$ un vecteur de bruit CCCN tel que $\eta^+, \eta^- \in [0; 1[$ et $\eta^+ + \eta^- \neq 1$. Soient $c \in \mathcal{C}$ et $P \in \mathcal{P}$, l'oracle $EX^{\overrightarrow{\eta}}_{CCCN}(c, P)$ fournit à chaque appel une paire $(x, c^{\overrightarrow{\eta}}(x))$ telle que x est tiré selon la distribution P et $c^{\overrightarrow{\eta}}(x)$ est défini par :*

$$\textit{si } c(x) = 1 \textit{ alors } c^{\overrightarrow{\eta}}(x) = \begin{cases} 1 & \textit{avec probabilité } 1 - \eta^+ \\ -1 & \textit{avec probabilité } \eta^+ \end{cases}$$



(a) (b)

Figure C.1. (a) Un exemple classique d'apprentissage d'un concept c à partir de 26 exemples positifs ($c(x) = 1$) et 37 exemples négatifs ($c(x) = -1$). Dans ce cas, $\eta^+ = 0$ et $\eta^- = 0$. (b) Le même exemple en présence de bruit de classification CCCN $\eta^+ = 8/26$, $\eta^- = 13/37$.

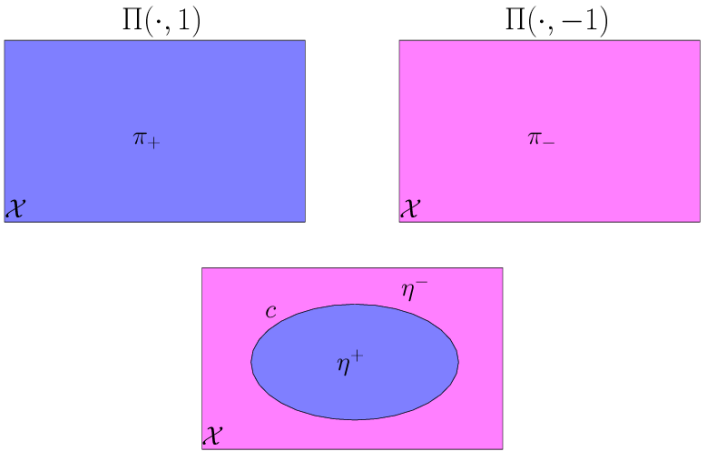


Figure C.2. Partition Π de $\mathcal{X} \times \mathcal{Y}$ induite par le modèle CCCN.

$$\textit{si } c(x) = -1 \textit{ alors } c^{\overrightarrow{\eta}}(x) = \begin{cases} -1 & \textit{avec probabilité } 1 - \eta^- \\ 1 & \textit{avec probabilité } \eta^- \end{cases}$$

Remarquons que c'est un modèle de bruit proche du modèle CPCN présenté à la section 2.1.4, pour lequel la partition Π sur $\mathcal{X} \times \mathcal{Y}$ considérée vaut $\Pi = \{\pi_+, \pi_-\}$ (*cf* Figure C.2) et le vecteur de bruit associé $\overrightarrow{\eta} = [\eta^+, \eta^-]$.

On définit alors la CCCN-apprenabilité d'une classe de concepts $\mathcal{C}$ de la façon suivante :

Définition C.2. *une classe de concepts $\mathcal{C}$ est efficacement CCCN-apprenable par une classe de représentation $\mathcal{H}$ ssi il existe un algorithme $\mathcal{A}$ et deux polynômes p et q tels que $\forall c \in \mathcal{C}$, $\forall P \in \mathcal{P}$, $\forall \epsilon, \delta > 0$ et $\forall \overrightarrow{\eta} = [\eta^+, \eta^-]$ tel que $\eta^+, \eta^- \in [0; 1[$ et $\eta^+ + \eta^- \neq 1$, en ayant accès à $EX^{\overrightarrow{\eta}}_{CCCN}(c, P)$, ϵ et δ, $\mathcal{A}$ calcule avec une probabilité $> 1 - \delta$ une hypothèse $h \in \mathcal{H}$ telle que $R(h) \leq \epsilon$. $\mathcal{A}$ requiert au plus $p(\frac{1}{\epsilon}, \frac{1}{\delta}, \frac{1}{1-\eta^+-\eta^-})$ appels à $EX^{\overrightarrow{\eta}}_{CCCN}(c, P)$ et s'arrête en temps $q(\frac{1}{\epsilon}, \frac{1}{\delta}, \frac{1}{1-\eta^+-\eta^-})$.*

CN=CPCN

Les pages qui suivent donnent la démonstration du résultat CN=CPCN sous sa forme publiée dans [Ralaivola et al., 2006b].

CN=CPCN

Liva Ralaivola LIVA.RALAIVOLA@LIF.UNIV-MRS.FR
François Denis FRANCOIS.DENIS@LIF.UNIV-MRS.FR
Christophe N. Magnan CHRISTOPHE.MAGNAN@LIF.UNIV-MRS.FR
Laboratoire d'Informatique Fondamentale de Marseille
UMR 6166 CNRS, 39, rue F. Joliot-Curie, F-13453 Marseille cedex 13, France

Abstract

We address the issue of the learnability of concept classes under three classification noise models in the *probably approximately correct* framework.

After introducing the Class-Conditional Classification Noise (CCCN) model, we investigate the problem of the learnability of concept classes under this particular setting and we show that concept classes that are learnable under the well-known uniform classification noise (CN) setting are also CCCN-learnable, which gives CN=CCCN.

We then use this result to prove the equality between the set of concept classes that are CN-learnable and the set of concept classes that are learnable in the Constant Partition Classification Noise (CPCN) setting, or, in other words, we show that CN=CPCN.

1. Introduction

This paper presents a study in the probably approximately correct (PAC) framework. In particular, we investigate the equality of concept classes in different classification noise settings from the learnability standpoint.

More precisely, we study three different noise settings: the *uniform classification noise setting* CN (Angluin & Laird, 1988), the *class-conditional classification noise setting* CCCN and the *constant partition classification noise setting* CPCN (Decatur, 1997). The second setting is a particular case of the latter and it is characterized by a by a noise process that flips the label of an example according to unifom classification noise processes defined on each (positive or negative) class. This setting is therefore a generalization of the uniform classification noise setting where noise is added independently of the class of the examples.

Our first contribution is the formal proof that CN = CCCN, that is, the concept classes that are learnable (in the PAC sense) under the CN framework (these classes are said to be CN-learnable) are also learnable under the CCCN (they are therefore CCCN-learnable) framework, and conversely. The idea to prove this result is that it is possible to bring a CCCN learning problem down to a CN learning problem by an appropriate addition of noise to the labeled examples of the CCCN problem.

Our second contribution is the proof that CN = CPCN, that is, the concept classes that are CN-learnable are CPCN-learnable, and conversely. The underlying idea of the proof is that a CPCN learning problem can be decomposed into several CCCN learning problems.

The paper is organized as follows. Section 2 briefly recalls the notion of PAC-learnability and formally presents and/or recalls the different noise settings together with the corresponding definitions of CN-learnability, CCCN-learnability and CPCN-learnability. Section 3 gives the proof of CN=CCCN while section 4 develops that of CN=CPCN. A short discussion on possible relaxation of noise constraints is provided in section 5.

2. Preliminaries

2.1. Learning in the PAC Framework

In the classical PAC learning framework, the problem of concept learning can be stated as follows (Valiant, 1984). Let $\mathcal{X}$ be a space (e.g. $\mathbb{R}^n$ or $\{0,1\}^d$), subsequently referred to as the *input space*. Let c be some *concept* from a *concept class* $\mathcal{C}$ (basically, $\mathcal{C}$ is a subset

Appearing in *Proceedings of the 23rd International Conference on Machine Learning*, Pittsburgh, PA, 2006.

of $\mathcal{X}$) and D some fixed but unknown distribution on $\mathcal{X}$ from $\mathcal{D}$, the set of all the distributions on $\mathcal{X}$. The task of learning is that of identifying c given access only to a *sampling oracle* $EX(c, D)$, such that each call to $EX(c, D)$ outputs a pair $\langle \mathbf{x}, t(\mathbf{x})\rangle$, with $\mathbf{x} \in \mathcal{X}$ drawn randomly according to D and $t(\mathbf{x}) = 1$ if $\mathbf{x} \in c$ and $t(\mathbf{x}) = 0$ otherwise (i.e. t is the indicator function of c). $\mathcal{C}$ is said to be *efficiently PAC-learnable*, if, there is an algorithm $\mathcal{A}$ such that for every concept c in $\mathcal{C}$, for every distribution D over $\mathcal{X}$, for every $\varepsilon > 0$ and for every $\delta > 0$, $\mathcal{A}$, when given access to $EX(c, D)$, outputs with probability at least $1 - \delta$ a hypothesis $h \in \mathcal{H}$, where $\mathcal{H}$ is a representation class over $\mathcal{X}$, such that the probability $\mathrm{err}_D(h) := P_{\mathbf{x}\sim D}(h(\mathbf{x}) \neq t(\mathbf{x}))$ of disagreement between h and t on instances randomly drawn from D is lower than ε (Kearns & Vazirani, 1994); $\delta > 0$ is referred to as the *confidence* parameter (although the confidence is actually $1 - \delta$), $\varepsilon > 0$ as the *precision* and $\mathrm{err}_D(h)$ is the *error* of h. There must be two polynomials $p(\cdot, \cdot)$ and $q(\cdot, \cdot)$, such that in order to draw a hypothesis h, $\mathcal{A}$ needs at most $p(\frac{1}{\varepsilon}, \frac{1}{\delta})$ training examples and it runs in at most $q(\frac{1}{\varepsilon}, \frac{1}{\delta})$ time. These two polynomials should also take as another argument the size of the concept c to be learned, but as it will not play any explicit role in our discussion, we have decided for sake of clarity not to mention it in the sample size and time requirements.

2.2. CN, CPCN and CCCN Learnability

In the framework of uniform Classification Noise (CN) concept learning (Angluin & Laird, 1988), the oracle to which the learning procedure has access is defined as follows (Angluin & Laird, 1988).

Definition 1 (CN oracle $EX_{\mathrm{CN}}(c, D)$). *Let $\eta \in [0; 1]$. Given $c \in \mathcal{C}$ and $D \in \mathcal{D}$, the uniform Classification Noise oracle $EX_{\mathrm{CN}}(c, D)$ outputs a pair $\langle \mathbf{x}, t^\eta(\mathbf{x})\rangle$ according to the following procedure:*

(a) $\mathbf{x}$ is drawn randomly according to D;

(b) $t^\eta(\mathbf{x})$ is set as

$$t^\eta(\mathbf{x}) := \begin{cases} t(\mathbf{x}) \text{ with prob. } 1 - \eta \\ \neg t(\mathbf{x}) \text{ with prob. } \eta. \end{cases}$$

The notion of CN-learnability, defined by Angluin and Laird (1988) readily follows.

Definition 2 (CN-learnability). *A concept class $\mathcal{C}$ is efficiently CN-learnable by representation class $\mathcal{H}$* iff *there exist an algorithm $\mathcal{A}$ and polynomials $p(\cdot, \cdot, \cdot)$ and $q(\cdot, \cdot, \cdot)$ such that for any $c \in \mathcal{C}$, for any $D \in \mathcal{D}$, for any $\varepsilon > 0$, for any $\delta \in]0; 1]$ and for any $\eta \in [0; 0.5[$, when given access to $EX^\eta_{\mathrm{CN}}(c, D)$ and given inputs ε, δ and an upper bound $\eta_b < 0.5$ on η, $\mathcal{A}$ outputs with probability at least $1 - \delta$ a hypothesis $h \in \mathcal{H}$ such that $err_D(h) \leq \varepsilon$.*

To output such an hypothesis $\mathcal{A}$ requires at most $p(\frac{1}{\varepsilon}, \frac{1}{\delta}, \frac{1}{1-2\eta_b})$ training samples and it runs in $q(\frac{1}{\varepsilon}, \frac{1}{\delta}, \frac{1}{1-2\eta_b})$ time.

Remark 1. *Here, we have assumed the knowledge of an upper bound η_b on the actual noise η. As stated in (Angluin & Laird, 1988) and (Kearns & Vazirani, 1994), this assumption is not restrictive since it is possible to guess a value for η_b when none is provided. The classes of concepts that can be CN-learned with a provided bound η_b are therefore exactly the same as the ones that can be CN-learned without any knowledge on an upperbound on η.*

As for CPCN-learnability, introduced in (Decatur, 1997), this setting assumes a set of partition functions $\Pi = \{\pi_1, \ldots, \pi_k\}$ defined on the labeled space $\mathcal{X} \times \mathcal{Y}$ and taking values in $\{0, 1\}$ such that $\sum_{i=1}^k \pi_i(\langle \mathbf{x}, y\rangle) = 1$ for any pair $\langle \mathbf{x}, y\rangle$ from $\mathcal{X}\times\mathcal{Y}$ and it assumes a CPCN oracle as defined by Decatur (1997).

Definition 3 (CPCN oracle $EX^{\Pi,\boldsymbol{\eta}}_{\mathrm{CPCN}}(c, D)$). *Let $\Pi = \{\pi_1, \ldots, \pi_k\}$ a set of partition functions over $\mathcal{X} \times \mathcal{Y}$ and $\boldsymbol{\eta} = [\eta_1 \ldots \eta_k]$, with $\eta_i \in [0; 1]$. Given $c \in \mathcal{C}$ and $D \in \mathcal{D}$, the CPCN oracle $EX^{\Pi,\boldsymbol{\eta}}_{\mathrm{CPCN}}(c, D)$ outputs a labeled example $\langle \mathbf{x}, t^{\boldsymbol{\eta}}(\mathbf{x})\rangle$ as follows:*

(a) $\mathbf{x}$ is drawn according to D;

(b) if i is the index such that $\pi_i(\langle \mathbf{x}, t(\mathbf{x})\rangle) = 1$ then

$$t^{\boldsymbol{\eta}}(\mathbf{x}) := \begin{cases} t(\mathbf{x}) \text{ with prob. } 1 - \eta_i \\ \neg t(\mathbf{x}) \text{ with prob. } \eta_i. \end{cases}$$

The next definition is that of CPCN-learnability (Decatur, 1997).

Definition 4 (CPCN-learnability). *A concept class $\mathcal{C}$ is efficiently CPCN-learnable by representation class $\mathcal{H}$* iff *there exist an algorithm $\mathcal{A}$ and polynomials $p(\cdot, \cdot, \cdot)$ and $q(\cdot, \cdot, \cdot)$ such that for any set $\Pi = \{\pi_1, \ldots, \pi_k\}$ of partition functions, for any $\boldsymbol{\eta} = [\eta_1 \cdots \eta_k]$, with $\eta_i \in [0; 1/2[$, for any $c \in \mathcal{C}$, for any $D \in \mathcal{D}$, for any $\varepsilon > 0$ and for any $\delta \in]0; 1]$, when given access to $EX^{\Pi,\boldsymbol{\eta}}_{\mathrm{CPCN}}(c, D)$ and given inputs ε, δ and an upper bound $\eta_b < 0.5$ on the noise rates η_i, $\mathcal{A}$ outputs with probability at least $1 - \delta$ a hypothesis $h \in \mathcal{H}$ such that $err_D(h) \leq \varepsilon$.*

To output such an hypothesis $\mathcal{A}$ requires at most $p(\frac{1}{\varepsilon}, \frac{1}{\delta}, \frac{1}{1-2\eta_b})$ training samples and it runs in $q(\frac{1}{\varepsilon}, \frac{1}{\delta}, \frac{1}{1-2\eta_b})$ time.

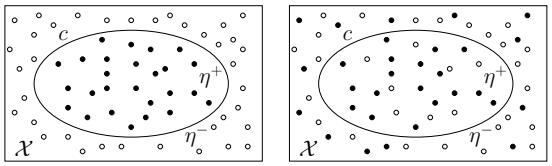


Figure 1. Left: classical (noise free) concept learning setting showing 26 positive examples (black discs) and 37 negative examples (white discs); $\eta^1 = 0$ and $\eta^0 = 0$. Right: Class-Conditional Noise concept learning setting; the values of η^1 and η^0 might be $\eta^1 = 8/26$ and $\eta^0 = 13/37$.

In order to prove our main result, that is, CN = CPCN, we will focus on the specific CPCN case where $\Pi = \{\pi_+, \pi_-\}$ and $\boldsymbol{\eta} = [\eta^1\ \eta^0]$ with $\pi_+(\mathbf{x}, y) = y$ and $\pi_-(\mathbf{x}, y) = 1 - y$. A CPCN oracle $EX^{\Pi,\boldsymbol{\eta}}_{\text{CPCN}}$ defined along this setting corresponds to the case where different classification noises are applied to positive and negative examples, as illustrated on Figure 1. From now on, we refer to the problem of learning in this particular framework, i.e., with $\Pi = \{\pi_+, \pi_-\}$, $\boldsymbol{\eta} = [\eta^1\ \eta^0]$ and the corresponding CPCN oracle, as the problem of learning under Class-Conditional Classification Noise (CCCN); the corresponding oracle will hence be denoted as $EX^{\boldsymbol{\eta}}_{\text{CCCN}}$. CCCN-learnability is obviously defined as in Definition 4.

3. CN=CCCN

The main theorem of this section states that the class CN of concepts that are learnable under the uniform classification noise model is the same as the class CCCN of concepts that are learnable under the class-conditional classification noise model:

Theorem 1. *CCCN=CN*

CCCN $\subseteq$ CN is obvious: if $c \in \mathcal{C}$ is a concept from a class $\mathcal{C}$ that is CCCN-learnable with any noise vector $\boldsymbol{\eta} = [\eta^1\ \eta^0]$ given a noise upper bound on η_b then it is still learnable when $\eta^1 = \eta^0$, i.e. it is CN-learnable (with the same noise upper bound η_b).

3.1. Sketch of the Proof

The proof of Theorem 1 proceeds in three steps. First, we show (Lemma 1) that from noisy oracle $EX^{\boldsymbol{\eta}}_{\text{CCCN}}$, it is possible to construct another CCCN noisy oracle $EX^{\bar{\boldsymbol{\eta}}}_{\text{CCCN}}$ whose noise vector $\bar{\boldsymbol{\eta}} = [\bar{\eta}^1\ \bar{\eta}^0]$ depends on two 'renoising' control parameters ρ and s. In addition, we observe that there exists a specific pair (ρ_{opt}, s_{opt}) of values that allows to turn a CCCN learning problem into CN learning problem.

Secondly, we show (Lemma 2) that it suffices to know a sufficiently accurate approximation ρ to ρ_{opt} (with the correct setting of the corresponding s) to 'almost' meet the requirements for PAC-learnability from the CCCN-oracle.

Then, it is proved that knowing $\eta_b < 0.5$ such that $\eta^1, \eta^0 \leq \eta_b$ makes it possible to learn any CCCN concept that is CN-learnable (Proposition 1). This concludes the proof of Theorem 1.

Algorithm 1 FlipLabel

Input: $\rho \in [0;1]$, $s \in \{0,1\}$, $t \in \{0,1\}$
Output: $t^{\rho,s} \in \{0,1\}$

```
Draw a random number r uniformly in [0;1]
if s = t then
  t^{ρ,s} := t
else
  if r ≤ ρ then
    t^{ρ,s} := 1 − t
  else
    t^{ρ,s} := t
  end if
end if
return t^{ρ,s}
```

3.2. Formal Proof

Lemma 1. *Let $c \in \mathcal{C}$ and $D \in \mathcal{D}$. Let $EX^{\boldsymbol{\eta}}_{\text{CCCN}}(c, D)$ be the CCCN oracle with noise vector $\boldsymbol{\eta} = [\eta^1\ \eta^0]$ with $\eta^1, \eta^0 \in [0;1]$. Given parameters $\rho \in [0;1]$ and $s \in \{0,1\}$, the procedure that returns a pair $\langle \mathbf{x}, t^{\bar{\boldsymbol{\eta}}}(\mathbf{x})\rangle$ by (1) polling a labeled example $\langle \mathbf{x}, t^{\boldsymbol{\eta}}(\mathbf{x})\rangle$ from $EX^{\boldsymbol{\eta}}_{\text{CCCN}}$ and (2) setting $t^{\bar{\boldsymbol{\eta}}}(\mathbf{x})$ to* FlipLabel$(\rho, s, t^{\boldsymbol{\eta}}(\mathbf{x}))$ *(cf. Algorithm 1), simulates a call to a CCCN-oracle $EX^{\bar{\boldsymbol{\eta}}}_{\text{CCCN}}(c, D)$ of noise vector $\bar{\boldsymbol{\eta}} = [\bar{\eta}^1\ \bar{\eta}^0]$ with $\bar{\eta}^1, \bar{\eta}^0 \in [0;1]$ and such that*

$$\bar{\eta}^1 = (1-\rho)\eta^1 + (1-s)\rho \ \text{ and } \ \bar{\eta}^0 = (1-\rho)\eta^0 + s\rho.$$

Proof. Let $c \in \mathcal{C}$, $D \in \mathcal{D}$, $\rho \in [0;1]$ and, for sake of exposition, suppose that $s = 1$.

The procedure described in the lemma together with the way FlipLabel is defined (cf. Algorithm 1) are such that $P(t^{\bar{\boldsymbol{\eta}}}(\mathbf{x}) = 1 | t^{\boldsymbol{\eta}}(\mathbf{x}) = 1) = 1$ and $P(t^{\bar{\boldsymbol{\eta}}}(\mathbf{x}) = 1 | t^{\boldsymbol{\eta}}(\mathbf{x}) = 0) = \rho$. Therefore, the probabilities of flipping the class $t(\mathbf{x})$ of a random example $\mathbf{x}$ to the opposite class $1 - t(\mathbf{x})$ are given by (dropping the dependence on $\mathbf{x}$)

$$\begin{aligned}
\bar{\eta}^1 &= P(t^{\bar{\boldsymbol{\eta}}} = 0 | t = 1) \\
&= P(t^{\bar{\boldsymbol{\eta}}} = 0, t^{\boldsymbol{\eta}} = 0 | t = 1) + P(t^{\bar{\boldsymbol{\eta}}} = 0, t^{\boldsymbol{\eta}} = 1 | t = 1) \\
&= P(t^{\bar{\boldsymbol{\eta}}} = 0 | t^{\boldsymbol{\eta}} = 0) P(t^{\boldsymbol{\eta}} = 0 | t = 1) \\
&\qquad + P(t^{\bar{\boldsymbol{\eta}}} = 0 | t^{\boldsymbol{\eta}} = 1) P(t^{\boldsymbol{\eta}} = 1 | t = 1) \\
&= (1-\rho)\eta^1,
\end{aligned}$$

and

$$\begin{aligned}\bar{\eta}^0 &= P(t^{\bar{\boldsymbol{\eta}}} = 1 | t = 0)\\ &= P(t^{\bar{\boldsymbol{\eta}}} = 1, t^{\boldsymbol{\eta}} = 1 | t = 0) + P(t^{\bar{\boldsymbol{\eta}}} = 1, t^{\boldsymbol{\eta}} = 0 | t = 0)\\ &= P(t^{\bar{\boldsymbol{\eta}}} = 1 | t^{\boldsymbol{\eta}} = 1) P(t^{\boldsymbol{\eta}} = 1 | t = 0)\\ &\qquad + P(t^{\bar{\boldsymbol{\eta}}} = 1 | t^{\boldsymbol{\eta}} = 0) P(t^{\boldsymbol{\eta}} = 0 | t = 0)\\ &= \rho + \eta^0 (1 - \rho),\end{aligned}$$

which corresponds to the expressions for $\bar{\eta}^1$ and $\bar{\eta}^0$ provided in the lemma. It is straightforward to check that when $s = 0$ we do also recover these expressions.

Checking that $\bar{\eta}^1, \bar{\eta}^0$ are in $[0; 1]$ is straightforward: both $\bar{\eta}^1$ and $\bar{\eta}^0$ are bounded from above by $(1 - \rho)\max(\eta^1, \eta^0) + \rho$, which, since $1 - \rho \geq 0$ and $\max(\eta^1, \eta^0) \in [0; 1]$, is upper bounded by $(1 - \rho) + \rho = 1$.

□

Remark 2. *This lemma has the direct consequence that it is possible to get a CN oracle from a CCCN oracle as soon as the noise parameters of the CCCN oracle are known. Indeed, if $EX^{\boldsymbol{\eta}}_{\mathrm{CCCN}}(c, D)$ is a CCCN oracle of known noise vector $\boldsymbol{\eta} = [\eta^1\ \eta^0]$ then using $\rho_{opt} := \frac{|\eta^1 - \eta^0|}{1 + |\eta^1 - \eta^0|}$ and setting $s_{opt} := 1$ if $\eta^1 > \eta^0$ and 0 otherwise allows to obtain a CN oracle. This CN oracle has its noise equal to $\eta_{opt} := \bar{\eta}^1 = \bar{\eta}^0 = \frac{\max(\eta^1, \eta^0)}{1 + |\eta^1 - \eta^0|}$.*

Remark 3. *If only an upper bound $\eta_b < 0.5$ is known on the noise rates η^1 and η^0 of a CCCN oracle $EX^{\boldsymbol{\eta}}_{\mathrm{CCCN}}(c, D)$ then it is straightforward to see that $\rho_{opt} \leq \eta_b$ and that the noise of the CN oracle obtained from $EX^{\boldsymbol{\eta}}_{\mathrm{CCCN}}(c, D)$ by adding noise to one of the classes is also upper bounded by η_b.*

Lemma 2. *Let $\mathcal{C}$ be a concept class on $\mathcal{X}$ that is CN-learnable by representation class $\mathcal{H}$. Let $\mathcal{A}^\eta$ be an algorithm that CN-learns $\mathcal{C}$ (with any noise rate $\eta \in [0; 1/2[$); $p(\cdot, \cdot, \cdot)$ and $q(\cdot, \cdot, \cdot)$ are polynomials (in $1/\varepsilon, 1/\delta, 1/(1 - 2\eta)$, respectively) for $\mathcal{A}^\eta$'s sample size and time requirements.*

Let $\eta^1, \eta^0 \in [0; 0.5[$ be the (actual) unkown *noise levels for the positive class and the negative class. Assume that we know a value $\eta_b < 0.5$ such that $\eta^1 \leq \eta_b$ and $\eta^0 \leq \eta_b$ and that we know whether $\eta^1 \geq \eta^0$.*

There exists an algorithm $\mathcal{A}$ such that for any $c \in \mathcal{C}$, for any $D \in \mathcal{D}$, for any $\varepsilon > 0$, for any $\delta > 0$, for any $\Delta \in]0; 1]$, for $\ell := p(1/\varepsilon, 1/\delta, 1/(1 - 2\eta_b))$ and $\tau := \frac{\Delta}{2\ell}$, for any $\rho \in [0; 1]$ verifying $|\rho - \rho_{opt}| < \tau$, for $s := \mathbf{1}_{(\eta^1 \geq \eta^0)}$, $\mathcal{A}$, when given inputs ε, δ, ρ, s, $\mathcal{A}^\eta$, ℓ and η_b and given access to $EX^{\boldsymbol{\eta}}_{\mathrm{CCCN}}(c, \mathcal{D})$, outputs with probability $1 - \delta - \Delta$ a hypothesis $h \in \mathcal{H}$ verifying

$$err_D(h) \leq \varepsilon.$$

Algorithm 2 ApproximateLearn

Input: $\varepsilon > 0$, $\delta > 0$, $\rho \in [0; 1]$, $s \in \{0, 1\}$, $\mathcal{A}^\eta$ that CN-learns $\mathcal{C}$ with $q(\cdot, \cdot, \cdot)$ running time, $\ell \in \mathbb{N}$, $\eta_b \in [0; 1/2[$

Output: $h \in \mathcal{H}$

Build the CCCN oracle $EX^{\bar{\boldsymbol{\eta}}}_{\mathrm{CCCN}}(c, D)$ using ρ and s as in Lemma 1
Draw a sample $\mathcal{S} = \{\langle \mathbf{x}_1, t^{\bar{\boldsymbol{\eta}}}(\mathbf{x}_1)\rangle, \ldots, \langle \mathbf{x}_\ell, t^{\bar{\boldsymbol{\eta}}}(\mathbf{x}_\ell)\rangle\}$ of size ℓ from $EX^{\bar{\boldsymbol{\eta}}}_{\mathrm{CCCN}}(c, D)$
Input $\mathcal{S}$, ε and δ to $\mathcal{A}^\eta$ with the upper bound on η set to η_b
if the running time of $\mathcal{A}^\eta$ gets longer than $q(1/\varepsilon, 1/\delta, 1/(1 - 2\eta_b))$ **then**
  stop $\mathcal{A}^\eta$ and return $\emptyset$
else
  return the hypothesis $h \in \mathcal{H}$ output by $\mathcal{A}^\eta$
end if

In order to output h, $\mathcal{A}$ requires a polynomial number of labeled data and runs in polynomial time.

Proof. The idea of the proof is that if ρ is not too far from ρ_{opt} and s is set to the correct value (either 0 or 1) then the oracle resulting from the procedure specified in Lemma 1 is 'almost' a CN oracle and c can therefore be learned under D by $\mathcal{A}^\eta$.

Let us fix $\varepsilon > 0$, $\delta > 0$, $c \in \mathcal{C}$ and $D \in \mathcal{D}$. We assume, without loss of generality, that $\eta^1 \geq \eta^0$ and that, as a consequence, the indicator function $\mathbf{1}_{(\eta^1 \geq \eta^0)}$ takes the value 1. We also fix ρ such that $|\rho - \rho_{opt}| < \tau$.

We show that a call to ApproximateLearn (cf. Algorithm 2) with the inputs ε, δ, ρ, s, $\mathcal{A}^\eta$, ℓ and η_b outputs with probabily $1 - \delta - \Delta$ a hypothesis $h \in \mathcal{H}$ such that $\mathrm{err}(h) \leq \varepsilon$.

We know from Remark 3 that η_{opt}, the noise of the CN oracle obtained when using the procedure of Lemma 1 with ρ_{opt} and s_{opt}, is bounded from above by η_b. Therefore, ℓ set as in the lemma ensures that if $\mathcal{A}^\eta$ is provided with ℓ labeled sample data from a CN-oracle with noise lower than η_b, then it outputs with probability $1 - \delta$ a hypothesis having error not larger than ε. In addition, the running time to output such an hypothesis will not exceed $q(1/\varepsilon, 1/\delta, 1/(1 - 2\eta_b))$. The following analysis, which builds on an idea of Goldberg (2005), shows that it is possible with high probability to draw from $EX^{\boldsymbol{\eta}}_{\mathrm{CCCN}}$ samples of size ℓ that can be interpreted as samples drawn from the CN oracle having noise η_{opt}.

Given a sample $\mathcal{S} = \{(\mathbf{x}_1, t(\mathbf{x}_1)), \ldots, (\mathbf{x}_\ell, t(\mathbf{x}_\ell))\}$ drawn from noise free oracle $EX(c, D)$ and param-

Algorithm 3 CorruptSample

Input: $\mathcal{S} = \{(\mathbf{x}_1, t_1), \ldots, (\mathbf{x}_\ell, t_\ell)\} \in (\mathcal{X} \times \{0,1\})^\ell$, η, $\bar{\boldsymbol{\eta}} = [\bar{\eta}^1\ \bar{\eta}^0]$
Output: $\mathcal{S}^\eta = \{(\mathbf{x}_1, t_1^\eta), \ldots, (\mathbf{x}_\ell, t_\ell^\eta)\}$,
$\mathcal{S}^{\bar{\boldsymbol{\eta}}} = \{(\mathbf{x}_1, t_1^{\bar{\boldsymbol{\eta}}}), \ldots, (\mathbf{x}_\ell, t_\ell^{\bar{\boldsymbol{\eta}}})\}$

```
for i = 1, ..., ℓ do
    Draw a random number u uniformly in [0; 1]
    /* noise process 1 */
    if u ≤ η then
        t_i^η := 1 − t_i
    else
        t_i^η := t_i
    end if
    /* noise process 2 */
    if u ≤ η̄^{t_i} then
        t_i^η̄ := 1 − t_i
    else
        t_i^η̄ := t_i
    end if
end for
return (S^η, S^η̄)
```

eters η_{opt} and $\bar{\boldsymbol{\eta}}$ as inputs, algorithm CorruptSample (cf. Algorithm 3) produces two labeled samples $\mathcal{S}^{\eta_{opt}} = \{(\mathbf{x}_1, t^{\eta_{opt}}(\mathbf{x}_1)), \ldots, (\mathbf{x}_\ell, t^{\eta_{opt}}(\mathbf{x}_\ell)\}$ and $\mathcal{S}^{\bar{\boldsymbol{\eta}}} = \{(\mathbf{x}_1, t^{\bar{\boldsymbol{\eta}}}(\mathbf{x}_1)), \ldots, (\mathbf{x}_\ell, t^{\bar{\boldsymbol{\eta}}}(\mathbf{x}_\ell)\}$, which may be noisy versions of $\mathcal{S}$. It is important to note that

(a) the two processes that $\mathcal{S}$ undergoes in order to give rise to $\mathcal{S}^{\eta_{opt}}$ and $\mathcal{S}^{\bar{\boldsymbol{\eta}}}$ (cf. Algorithm 3) precisely simulate a CN noise process with noise parameter η_{opt} and a CCCN noise process with noise parameter $\bar{\boldsymbol{\eta}} = [\bar{\eta}^1\ \bar{\eta}^0]$, respectively (see Definitions 1 and 3);

(b) the only way for $\mathcal{S}^{\eta_{opt}}$ and $\mathcal{S}^{\bar{\boldsymbol{\eta}}}$ to differ is that there exists (at least) an i such that $t^{\eta_{opt}}(\mathbf{x}_i) \neq t^{\bar{\boldsymbol{\eta}}}(\mathbf{x}_i)$: this means that during the i-th iteration of the main loop of CorruptSample, the random uniform variate u fell either between $\bar{\eta}^1$ and η_{opt} if $t(\mathbf{x}_i)$ was equal to 1 (hence, only one of $t^{\eta_{opt}}(\mathbf{x}_i)$ and $t^{\bar{\boldsymbol{\eta}}}(\mathbf{x}_i)$ would be a flipped version of $t(\mathbf{x}_i)$) or between $\bar{\eta}^0$ and η_{opt} if $t(\mathbf{x}_i)$ was equal to 0; dropping the dependence on i and using $p := P(t(\mathbf{x}) = 1)$, the probability of observing $t^{\eta_{opt}}(\mathbf{x}) \neq t^{\bar{\boldsymbol{\eta}}}(\mathbf{x})$ is therefore

$$P(t^{\eta_{opt}}(\mathbf{x}) \neq t^{\bar{\boldsymbol{\eta}}}(\mathbf{x})) = p|\bar{\eta}^1 - \eta_{opt}| + (1-p)|\bar{\eta}^0 - \eta_{opt}|. \tag{1}$$

From a broader perspective, $EX(c, D)$ in combination with CorruptSample – applied on labeled samples of size ℓ drawn from $EX(c, D)$ –, define a distribution $D_{\eta_{opt},\bar{\boldsymbol{\eta}}}(\mathcal{S}^{\eta_{opt}}, \mathcal{S}^{\bar{\boldsymbol{\eta}}})$ on $(\mathcal{X} \times \{0,1\})^\ell \times (\mathcal{X} \times \{0,1\})^\ell$. According to what we have just noted, the marginal distributions $D_{\eta_{opt}}(\mathcal{S}^{\eta_{opt}})$ and $D_{\bar{\boldsymbol{\eta}}}(\mathcal{S}^{\bar{\boldsymbol{\eta}}})$ derived from $D_{\eta_{opt},\bar{\boldsymbol{\eta}}}(\mathcal{S}^{\eta_{opt}}, \mathcal{S}^{\bar{\boldsymbol{\eta}}})$ are *exactly* the distributions of samples of size ℓ from $EX_{\mathrm{CN}}^{\eta_{opt}}(c, D)$ and $EX_{\mathrm{CCCN}}^{\bar{\boldsymbol{\eta}}}(c, D)$, respectively.

If we define $\omega(\mathcal{S})$ as the binary random variable that is equal to 1 if $\mathcal{S}$ allows $\mathcal{A}^\eta$ to output an hypothesis having error less than ε within the running time specified in ApproximateLearn (cf. Algorithm 2) and 0 otherwise, we have:

$$\begin{aligned}
&P_{\mathcal{S}^{\bar{\boldsymbol{\eta}}} \sim D_{\bar{\boldsymbol{\eta}}}}\left(\omega(\mathcal{S}^{\bar{\boldsymbol{\eta}}}) = 0\right) \\
&\quad = P_{(\mathcal{S}^\eta, \mathcal{S}^{\bar{\boldsymbol{\eta}}}) \sim D_{\eta_{opt},\bar{\boldsymbol{\eta}}}}\left(\omega(\mathcal{S}^{\bar{\boldsymbol{\eta}}}) = 0, \omega(\mathcal{S}^{\eta_{opt}}) = 1\right) \\
&\qquad + P_{(\mathcal{S}^\eta, \mathcal{S}^{\bar{\boldsymbol{\eta}}}) \sim D_{\eta_{opt},\bar{\boldsymbol{\eta}}}}\left(\omega(\mathcal{S}^{\bar{\boldsymbol{\eta}}}) = 0, \omega(\mathcal{S}^{\eta_{opt}}) = 0\right) \\
&\quad \leq P_{(\mathcal{S}^\eta, \mathcal{S}^{\bar{\boldsymbol{\eta}}}) \sim D_{\eta_{opt},\bar{\boldsymbol{\eta}}}}\left(\omega(\mathcal{S}^{\bar{\boldsymbol{\eta}}}) = 0, \omega(\mathcal{S}^{\eta_{opt}}) = 1\right) + \delta \\
&\quad \leq P_{(\mathcal{S}^\eta, \mathcal{S}^{\bar{\boldsymbol{\eta}}}) \sim D_{\eta_{opt},\bar{\boldsymbol{\eta}}}}(\mathcal{S}^{\bar{\boldsymbol{\eta}}} \neq \mathcal{S}^{\eta_{opt}}) + \delta.
\end{aligned}$$

However, for $(\mathcal{S}^\eta, \mathcal{S}^{\bar{\boldsymbol{\eta}}})$ distributed according to $D_{\eta_{opt},\bar{\boldsymbol{\eta}}}$, the following holds true:

$$\begin{aligned}
P(\mathcal{S}^{\eta_{opt}} \neq \mathcal{S}^{\bar{\boldsymbol{\eta}}}) &= P(t_1^{\eta_{opt}} \neq t_1^{\bar{\boldsymbol{\eta}}} \vee \ldots \vee t_\ell^{\eta_{opt}} \neq t_\ell^{\bar{\boldsymbol{\eta}}}) \\
&\leq \ell P(t^{\eta_{opt}}(\mathbf{x}) \neq t^{\bar{\boldsymbol{\eta}}}(\mathbf{x})) \quad \text{(union bound)} \\
&= \ell \left(p|\eta_{opt} - (1-\rho)\eta^1| \right. \quad \text{(cf. (1))} \\
&\qquad \left. +(1-p)|\eta_{opt} - \eta^0 - \rho(1-\eta^0)|\right) \\
&\leq \ell \left(|\rho - \rho_{opt}|\eta^1 + |\rho - \rho_{opt}|(1-\eta^0)\right) \\
&\qquad \text{(Remark 2)} \\
&\leq \ell\,(\tau + \tau) \quad \text{(assumption)} \\
&\leq \ell \cdot 2\frac{\Delta}{2\ell} \quad \text{(definition of } \tau) \\
&= \Delta,
\end{aligned}$$

where $t_i^{\eta_{opt}} := t^{\eta_{opt}}(\mathbf{x}_i)$, $t_i^{\bar{\boldsymbol{\eta}}} := t^{\bar{\boldsymbol{\eta}}}(\mathbf{x}_i)$.

The probability with which a random sample $\mathcal{S}^{\bar{\boldsymbol{\eta}}}$ of size ℓ generated from $D_{\bar{\boldsymbol{\eta}}}$, that is, polled from $EX_{\mathrm{CCCN}}^{\bar{\boldsymbol{\eta}}}(c, D)$, prevents a successful learning is therefore bounded by $\delta + \Delta$. In other words, when given access to a sample of size ℓ from $EX_{\mathrm{CCCN}}^{\bar{\boldsymbol{\eta}}}(c, D)$, as well as other input parameters, $\mathcal{A}^\eta$ has henceforth a probability at least $1 - \delta - \Delta$ to output a hypothesis having error lower than ε. □

Proposition 1. *Any concept class that is efficiently CN-learnable is also efficiently CCCN-learnable: CN ⊆ CCCN.*

More precisely, for every CN-learnable class there is an algorithm $\mathcal{A}$ such that for any concept c and any distribution D, for any $\epsilon > 0$ and $\delta > 0$, for any noise vector $\boldsymbol{\eta} = [\eta^1\ \eta^0]$ with $\eta^1, \eta^0 \leq \eta_b < 0.5$, when given access to $EX_{\mathrm{CCCN}}^{\boldsymbol{\eta}}(c, D)$, $\mathcal{A}$ outputs with probability $1-\delta$ a hypothesis h such that $err_D(h) \leq \epsilon$.

Proof. We note that this proposition closes the proof of Theorem 1.

Algorithm 4 LearnWithUpperBound

Input: $\varepsilon > 0$, $\delta > 0$, $\eta_b < 0.5$, $\mathcal{A}^\eta$ that CN-learns $\mathcal{C}$ with $p(\cdot,\cdot,\cdot)$ sample size
Output: a hypothesis $h \in \mathcal{H}$

$H := \emptyset$
$\Delta := \frac{\delta}{4}$
$\varepsilon' := \frac{\varepsilon}{4}(1-2\eta_b)$
$\delta' := \frac{\delta}{4}$
$\ell := p(\frac{1}{\varepsilon'}, \frac{1}{\delta'}, \frac{1}{1-2\eta_b})$
$\tau := \frac{\Delta}{2\ell}$
for all $s \in \{0,1\}$ and $i \in \mathbb{N}$ such that $i\tau < \eta_b$ **do**
  $\rho_i := i\tau$
  $H := H \cup \{\mathsf{ApproximateLearn}(\varepsilon', \delta', \rho_i, s, \mathcal{A}^\eta, \ell, \eta_b)\}$
end for
$m := \frac{8}{\varepsilon^2(1-2\eta_b)^2} \ln \frac{16\ell}{\delta^2}$
draw a sample $\mathcal{S}_m^{\boldsymbol{\eta}}$ of m labeled examples from $EX_{\mathrm{CCCN}}^{\boldsymbol{\eta}}(c, D)$
return $\mathrm{argmin}_{h\in H} \mathrm{err}_{\mathcal{S}_m^{\boldsymbol{\eta}}}(h)$

In order to prove this lemma, it suffices to see that algorithm LearnWithUpperBound (cf. Algorithm 4) can CCCN-learn any concept c from a class that is CN-learnable under any distribution D when given an upper bound $\eta_b < 0.5$ on η^1 and η^0.

Let us fix $c \in \mathcal{C}$, $D \in \mathcal{D}$, $\epsilon > 0$, $\delta > 0$ and let us assume that we know η_b.

From the way τ is set, the double loop of LearnWithUpperBound ensures that there is a pair (ρ^*, s^*) of values such that ρ^* is within a distance of τ from ρ_{opt} and $s^* = s_{opt}$. Hence, by applying lemma 2, we know that there is with probability at least $1-\delta/4-\delta/4 = 1-\delta/2$ a hypothesis h^* such that $\mathrm{err}(h^*) \leq \frac{\varepsilon}{4}(1-2\eta_b)$.

There is a need for a strategy capable with high probability to pick from H a hypothesis that has error lower than ε. Simple calculations give, for any h:

$$\begin{aligned} P(h(\mathbf{x}) \neq t^{\boldsymbol{\eta}}(\mathbf{x})) = p\eta^1 + (1-p)\eta^0 \\ + (1-2\eta^1)P(h(\mathbf{x})=1, t(\mathbf{x})=0) \\ + (1-2\eta^0)P(h(\mathbf{x})=0, t(\mathbf{x})=1), \end{aligned}$$

where, as earlier, p stands for $P(t(\mathbf{x})=1)$. Consequently, for any ε-bad hypothesis h, that is, any hypothesis having error larger than ε, we have

$$P(h(\mathbf{x}) \neq t^{\boldsymbol{\eta}}(\mathbf{x})) > p\eta^1 + (1-p)\eta^0 + \varepsilon(1-2\eta_b). \quad (2)$$

Besides, $P(h^*(\mathbf{x}) \neq t(\mathbf{x})) \leq \frac{\varepsilon}{4}(1-2\eta_b)$ implies

$$\begin{aligned} P(h^*(\mathbf{x}) = 1 \neq t(\mathbf{x}) = 0) \leq \frac{\varepsilon}{4}(1-2\eta_b) \\ P(h^*(\mathbf{x}) = 0 \neq t(\mathbf{x}) = 1) \leq \frac{\varepsilon}{4}(1-2\eta_b), \end{aligned}$$

and, therefore

$$\begin{aligned} P(h^*(\mathbf{x}) \neq t^{\boldsymbol{\eta}}(\mathbf{x})) \\ \leq p\eta^1 + (1-p)\eta^0 + \frac{\varepsilon}{4}(1-2\eta_b)\cdot 2(1-\eta^1-\eta^0) \\ \leq p\eta^1 + (1-p)\eta^0 + \frac{\varepsilon}{2}(1-2\eta_b). \end{aligned} \quad (3)$$

Equations (2) and (3) say that there is a gap of at least $\frac{\varepsilon}{2}(1-2\eta_b)$ between the error (on noisy patterns) of any ε-bad hypothesis and the error (on noisy patterns) of h^*. There is henceforth a size m of test sample $\mathcal{S}_m^{\boldsymbol{\eta}}$ such that the empirical errors measured on $\mathcal{S}_m^{\boldsymbol{\eta}}$ of all ε-bad hypotheses are far enough from the empirical error of h^*, i.e., for any $\varepsilon_{\mathrm{cut}}$ (strictly) within the bounds of (2) and (3), there is a size m of test sample that guarantees (with high probability) that the empirical errors on $\mathcal{S}_m^{\boldsymbol{\eta}}$ of all ε-bad hypotheses are above $\varepsilon_{\mathrm{cut}}$ while the empirical error of h^* on $\mathcal{S}_m^{\boldsymbol{\eta}}$ is below $\varepsilon_{\mathrm{cut}}$.

Letting $\varepsilon_{\mathrm{cut}} := p\eta^1 + (1-p)\eta^0 + \frac{3\varepsilon}{4}(1-2\eta_b)$, $H_{bad} := \{h \in H : \mathrm{err}(h) > \varepsilon\}$ and $G_{m,\varepsilon_{cut}}^{\boldsymbol{\eta}} := \{h \in H : \mathrm{err}_{\mathcal{S}_m^{\boldsymbol{\eta}}}(h) \leq \varepsilon_{\mathrm{cut}}\}$, we have

$$\begin{aligned} P(\exists h \in H_{bad} \cap G_{m,\varepsilon_{cut}}^{\boldsymbol{\eta}}) \leq |H_{bad}| P(h \in H_{bad} \cap G_{m,\varepsilon_{cut}}^{\boldsymbol{\eta}}) \\ \text{(union bound)} \\ \leq |H| \exp\left(-\frac{m\varepsilon^2(1-2\eta_b)^2}{8}\right) \\ \text{(Chernoff bound)} \\ \leq \frac{1}{2\tau} \exp\left(-\frac{m\varepsilon^2(1-2\eta_b)^2}{8}\right). \end{aligned}$$

In order to have $P(\exists h \in H_{bad} \cap G_{m,\varepsilon_{cut}}^{\boldsymbol{\eta}}) \leq \delta/4$, it suffices to choose m so that the right-hand side of the last inequation is bounded from above by $\delta/4$, i.e., it suffices to have

$$m = \frac{8}{\varepsilon^2(1-2\eta_b)^2} \ln \frac{16\ell}{\delta^2}$$

as it is set in LearnWithUpperBound.

Likewise, for h^*, we have

$$\begin{aligned} P(h^* \notin G_{m,\varepsilon_{cut}}^{\boldsymbol{\eta}}) \leq \exp\left(-\frac{m\varepsilon^2(1-2\eta_b)^2}{8}\right) \\ \text{(Chernoff bound)} \\ \leq 2\tau\frac{\delta}{4} = 2\cdot\frac{\delta}{8\ell}\cdot\frac{\delta}{4} \\ \leq \frac{\delta}{4} \end{aligned}$$

for the specific choice of m made.

It directly follows that the hypothesis $h_{\min}$ from H that minimizes the empirical error on $\mathcal{S}_m^{\boldsymbol{\eta}}$ – for the given value of m – is, with probability at least $1 - \delta/4 - \delta/4 = 1 - \delta/2$, a hypothesis that has *true* error

lower than ε. (We note that, though it may possibly be the case, $h_{\min}$ need not be h^*.)

All in all, we have that when given an upper bound on η^1 and η^0, and given access to a polynomial number of labeled data, LearnWithUpperBound outputs with probability at least $1-\delta$ a hypothesis with error at most ε. In addition, since ApproximateLearn controls its running time the running time of LearnWithUpperBound is polynomial as well. This closes the proof of Proposition 1. □

4. CPCN=CCCN=CN

In this section we provide a result showing the equality between CPCN and CCCN. This directly gives the main result of this paper, namely CN = CPCN.

The idea of the proof is that it is possible to build a partition of the input space $\mathcal{X}$ from the partition functions of a CPCN oracle (which define a partition over $\mathcal{S}\times\{0,1\}$): this partition is such that the noise process that corrupts the data within each part is a CCCN noise process. Given a CCCN learning algorithm $\mathcal{A}$ and some condition as for the number of data to draw from the CPCN oracle to be sure that each part contains enough (or no) data to be CCCN-learned, hypotheses are learned on each part. These hypotheses are used to relabel a CPCN sample of an appropriate size, which is in turn input to $\mathcal{A}$ to output with high probability a hypothesis having small error.

Lemma 3. *Let $c\in\mathcal{C}$ and $D\in\mathcal{D}$. Let h be a classifier that has error $err_D(h)\leq\varepsilon$. Then, for any $\alpha\in]0;1]$ and any integer $\ell\leq\alpha/\varepsilon$, the probability that h correctly predicts the labels of the elements of a sample of size ℓ drawn according to D is greater than $1-\alpha$.*

Proof. The probability that h correctly predicts the class of ℓ elements independently drawn according to D is greater than $(1-\varepsilon)^\ell$. It can be easily checked that for any $0\leq\varepsilon\leq 1$, $(1-\varepsilon)^\ell\geq 1-\ell\varepsilon\geq 1-\alpha$. □

Lemma 4. *Let $D\in\mathcal{D}$. Let $\overline{\pi}_1,\ldots,\overline{\pi}_k$ be a partition of $\mathcal{X}$, let $0<\varepsilon,\delta\leq 1$ be two parameters, let m be an integer and let $\ell\geq\max(2m/\varepsilon,-2\log(\delta/k)/\varepsilon^2)$. Then, with a probability greater than $1-\delta$, any sample $\mathcal{S}$ of $\mathcal{X}$ containing ℓ examples independently drawn according to D will contain at least m elements of each part $\overline{\pi}_i$ that satisfies $D(\overline{\pi}_i)\geq\varepsilon$, with $D(\overline{\pi}_i):=P_{\mathbf{x}\sim D}(\overline{\pi}_i(\mathbf{x})=1)$.*

Proof. We note that the partition $\overline{\pi}_1,\ldots,\overline{\pi}_k$ is defined with respect to the unlabeled space $\mathcal{X}$.

Let $\ell\geq\max(2m/\varepsilon,-2\log(\delta/k)/\varepsilon^2)$, let $\mathcal{S}$ be a sample containing ℓ examples independently drawn according to D, and let $m_i:=|\mathcal{S}\cap\overline{\pi}_i)|$. It comes from Chernoff bound and the way ℓ is chosen that, for any $1\leq i\leq k$,

$$P\left(\frac{m_i}{\ell}\leq D(\overline{\pi}_i)-\frac{\varepsilon}{2}\right)\leq\exp\left(-\frac{\ell\varepsilon^2}{2}\right)\leq\frac{\delta}{k}.$$

Hence, if $\overline{\pi}_i$ is such that $D(\overline{\pi}_i)\geq\varepsilon$,

$$P(m_i\leq m)\leq P\left(m_i\leq\ell\frac{\varepsilon}{2}\right)\leq\frac{\delta}{k}.$$

By the union bound

$$\begin{aligned}P(\exists i: m_i\leq m, D(\overline{\pi}_i)\geq\varepsilon)&\leq kP((m_i\leq m),D(\overline{\pi}_i)\geq\varepsilon)\\&=k\cdot\frac{\delta}{k}=\delta.\end{aligned}$$

Therefore, with probability greater than $1-\delta$, any part $\overline{\pi}_i$ such that $D(\overline{\pi}_i)\geq\varepsilon$ satisfies $m_i>m$. □

Proposition 2. *Let $\mathcal{C}$ be a class of concepts over $\mathcal{X}$ which is in CCCN. Then $\mathcal{C}$ is in CPCN. Stated otherwise: $CCCN\subseteq CPCN$.*

Proof. Let $\mathcal{A}$ be a CCCN learning algorithm for $\mathcal{C}$ and let $p(\cdot,\cdot,\cdot)$ be a polynomial such that for any target concept c in $\mathcal{C}$, any distribution $D\in\mathcal{D}$, any accuracy parameter ε, any confidence parameter δ and any noise rate bound η_b, if $\mathcal{A}$ is given as input a sample $\mathcal{S}$ drawn according to $EX^{\boldsymbol{\eta}}_{\text{CCCN}}(c,D)$ (where $\boldsymbol{\eta}=[\eta^1\ \eta^0]$ and $\eta^1,\eta^0\leq\eta_b$) and satisfying $|S|\geq p(1/\varepsilon,1/\delta,1/(1-2\eta_b))$, then $\mathcal{A}$ outputs a hypothesis whose error rate is lower than ε with probability at least $1-\delta$.

Let $\Pi=\{\pi_1,\ldots,\pi_k\}$ be a partition of $\mathcal{X}\times\{0,1\}$ and let $\eta=[\eta_1\cdots\eta_k]$ be a vector of noise rates satisfying $0\leq\eta_i\leq\eta_b$ for $1\leq i\leq k$. We deduce from Π a partition $\overline{\Pi}=(\overline{\pi}_1,\ldots,\overline{\pi}_l)$ of $\mathcal{X}$ based on the parts π_{ij} defined for $1\leq i,j\leq k$, by $\pi_{ij}=\{\mathbf{x}\in\mathcal{X}|\langle\mathbf{x},1\rangle\in\pi_i$ and $\langle\mathbf{x},0\rangle\in\pi_j\}$. (It is straightforward to check that for any $\mathbf{x}\in\mathcal{X}$, there exist i and j such that $\mathbf{x}\in\pi_{ij}$ and that $\pi_{ij}\cap\pi_{uv}\neq\emptyset$ implies $i=u$ and $j=v$.) For any $\overline{\pi}_i\in\overline{\Pi}$ such that $\overline{\pi}_i=\pi_{uv}$, define $\eta^1_i:=\eta_u$ and $\eta^0_i:=\eta_v$.

Let $c\in\mathcal{C}$, let $D\in\mathcal{D}$ and let $0<\varepsilon,\delta\leq 1$ be accuracy and confidence parameters.

Let $n_1\geq p(1/\varepsilon,4/\delta,1/(1-2\eta_b))$, let $\varepsilon_1:=\delta/(4ln_1)$, let $m:=p(1/\varepsilon_1,4l/\delta,1/(1-2\eta_b))$ and let $n_2\geq\max(2m/\varepsilon_1,-2\log(\delta/(4l))/\varepsilon_1^2)$. Note that n_2 is polynomial in $1/\varepsilon,1/\delta$ and $1/(1-2\eta_b)$.

Let $\mathcal{S}_2$ be a sample of size n_2 drawn according to $EX^{\Pi,\boldsymbol{\eta}}_{\text{CPCN}}(c,D)$. From Lemma 4, with probability at least $1-\delta/4$, any part $\overline{\pi}_i$ such that $D(\overline{\pi}_i)\geq\varepsilon_1$ satisfies $|\mathcal{S}_2\cap\overline{\pi}_i|>m$. Let $I:=\{i:|\mathcal{S}_2\cap\overline{\pi}_i|>m\}$.

For each $i\in I$, run algorithm $\mathcal{A}$ on each sample $\mathcal{S}_2\cap\overline{\pi}_i$ and let h_i be the output classifier. With a

probability greater than $1-\delta/4$, each h_i is such that $P_{D|\pi_i(\mathbf{x})=1}(h_i(\mathbf{x}) \neq t(\mathbf{x})) \leq \varepsilon_1$.

Now, let $\mathcal{S}_1$ be a new sample of size n_1 drawn according to $EX^{\Pi,\boldsymbol{\eta}}_{\text{CPCN}}(c, D)$.

- Let $\overline{\pi}_i$ be a part such that $D(\overline{\pi}_i) < \varepsilon_1$. The probability that $\mathcal{S}_1$ contains no element of $\overline{\pi}_i$ is $\geq (1-\varepsilon_1)^{n_1} \geq 1-\delta/(4l)$.
- Let $\overline{\pi}_i$ be a part such that $D(\overline{\pi}_i) \geq \varepsilon_1$. From Lemma 3, the probability that h_i computes the correct label of each example of $\mathcal{S}_1 \cap \overline{\pi}_i$ is greater than $1-\delta/(4l)$.

That is (using the union bound) the probability that $\mathcal{S}_1$ contains no element of a part $\overline{\pi}_i$ satisfying $D(\overline{\pi}_i) < \varepsilon_1$ and that all elements of $\mathcal{S}_1$ are correctly labeled by hypotheses $(h_i)_{i\in I}$ is greater than $1-\delta/4$.

Finally, relabel the examples of $\mathcal{S}$ using the predictions given by hypotheses $(h_i)_{i\in I}$ and run algorithm $\mathcal{A}$ on the relabeled sample $\tilde{\mathcal{S}}_1$. With a probability greater than $1-\delta/4$, it will output a hypothesis h such that $\text{err}_D(h) \leq \varepsilon$.

Taking everything together, the overall procedure outputs with probability $1-4\cdot\delta/4 = 1-\delta$ a hypothesis h that has error $\text{err}_D(h) \leq \varepsilon$. □

We can therefore state the main result of this paper:

Theorem 2. *CN=CCCN=CPCN.*

Proof. From the previous section, we know that CN = CCCN. In this section, we showed that CCCN ⊆ CPCN and, since CPCN ⊆ CCCN (the CCCN framework is a particular case of the CPCN framework), CPCN = CCCN. This trivially gives CN = CPCN. □

5. Bounds on the Noise Rates

In this study, we have restricted ourselves to the case where the upper bound η_b on the noise rates is stricly lower than 1/2. It can be shown that it is possible to address the case where instead of having an upper bound on the noise rates, a lower bound $\eta^l_b > 0.5$ is provided. Whichever the oracle to which the learning procedure is given access, it suffices to flip all the labels of the labeled examples it produces. Doing that brings the learning problem considered back to the classical setting where the upper bound on the noise rates is now $1-\eta^l_b$. The question of the learnability in the CPCN (or CCCN) framework in the more general case where some noise rates may be above 0.5 and some other below is still an open problem.

Another open question is that of the need of having an upper bound on the noise rates. Even though it is known that in the CN framework such a bound can be estimated (in polynomial time) through the learning process, it is not clear whether such an (distribution free) estimation can be carried out for the CCCN and CPCN cases.

6. Conclusion and Outlook

This paper presents a particular case of the learnability in the PAC-framework, where classes of examples are subject to various classification noise settings and we give two important results, namely, CN=CCCN and CN=CPCN.

An interesting outlook of this work is that of its application to the learning of noisy perceptrons (Blum et al., 1996; Cohen, 1997) in the CPCN framework. The question of the consequences of our work on the problem of learning optimal separating hyperplanes (in finite dimension as well as in infinite dimensional spaces) with soft-margins is also a problem we are interested in and that we plan to investigate.

References

Angluin, D., & Laird, P. (1988). Learning from Noisy Examples. *Machine Learning, 2.*

Blum, A., Frieze, A. M., Kannan, R., & Vempala, S. (1996). A Polynomial-Time Algorithm for Learning Noisy Linear Threshold Functions. *Proc. of 37th IEEE Symposium on Foundations of Computer Science* (pp. 330–338).

Cohen, E. (1997). Learning Noisy Perceptrons by a Perceptron in Polynomial Time. *Proc. of 38th IEEE Symposium on Foundations of Computer Science* (pp. 514–523).

Decatur, S. E. (1997). Pac Learning with Constant-Partition Classification Noise and Applications to Decision Tree Induction. *Proc. of the 14th Int. Conf. on Machine Learning.*

Goldberg, P. (2005). Some Discriminant-based PAC Algorithm. Personal communication.

Kearns, M. J., & Vazirani, U. V. (1994). *An Introduction to Computational Learning Theory.* MIT Press.

Valiant, L. (1984). A theory of the learnable. *Communications of the ACM, 27,* 1134–1142.

Bibliographie

[Altschul et al., 1997] Altschul, S., Madden, T., Schaffer, A., Zhang, J., Zhang, Z., Miller, W., and Lipman, D. (1997). Gapped BLAST and PSI-BLAST : a new generation of protein database search programs. *Nucleic Acids Res.*, 25(17) :3389–3402.

[Anfinsen, 1973] Anfinsen, C. B. (1973). Principles that Govern the Folding of Protein Chains. *Science*, 181(96) :223–230.

[Anfinsen et al., 1961] Anfinsen, C. B., Haber, E., Sela, M., and White, F. H. (1961). The Kinetics of Formation of Native Ribonuculease During Oxidation of the Reduced Polypeptide Chain. *PNAS*, 47 :1309–1314.

[Angluin and Laird, 1988] Angluin, D. and Laird, P. (1988). Learning From Noisy Examples. *Mach. Learn.*, 2(4) :343–370.

[Aslam and Decatur, 1994] Aslam, J. A. and Decatur, S. E. (1994). Improved Noise-Tolerant Learning and Generalized Statistical Queries. Technical report, Harvard University.

[Asuncion and Newman, 2007] Asuncion, A. and Newman, D. (2007). UCI machine learning repository.

[Balcan and Blum, 2005] Balcan, M.-F. and Blum, A. (2005). A PAC-Style Model for Learning from Labeled and Unlabeled Data. In *Proceedings of the 18th Annual Conference on Computational Learning Theory*, pages 111–126, London, UK. Springer-Verlag.

[Baldi et al., 2005] Baldi, P., Cheng, J., and Vullo, A. (2005). Large-Scale Prediction of Disulphide Bond Connectivity. In *Proceedings of NIPS' 05, Advances in Neural Information Processing Systems*, pages 97–104, Cambridge, MA. MIT Press.

[Baldi et al., 2000] Baldi, P., Pollastri, G., Andersen, C. A., and Brunak, S. (2000). Matching Protein β-Sheet Partners by Feedforward and Recurrent Neural Networks. In *Proceedings of the Eighth International Conference on Intelligent Systems for Molecular Biology*, pages 25–36. AAAI Press.

[Baldi et al., 2003] Baldi, P., Pollastri, G., Frasconi, P., and Vullo, A. (2003). New machine learning methods for the prediction of protein topologies. In Frasconi, P. and Shamir, R., editors, *Artificial Intelligence and Heuristic Methods in Bioinformatics*. IOS Press.

[Bayas et al., 2007] Bayas, M. V., Kearney, A., Avramovic, A., Van Der Merwe, P. A., and Leckband, D. E. (2007). Impact of Salt Bridges on the Equilibrium Binding and Adhesion of Human CD2 and CD58. *J. Biol. Chem.*, 282(8) :5589–5596.

[Berman et al., 2000] Berman, H. M., Westbrook, J., Feng, Z., Gilliland, G., Bhat, T. N., Weissig, H., Shindyalov, I. N., and Bourne, P. E. (2000). The Protein Data Bank. *Nucleic Acids Res.*, 28(1) :235–242.

[Blum et al., 1996] Blum, A., Frieze, A., Kannan, R., and Vempala, S. (1996). A Polynomial-time Algorithm for Learning Noisy Linear Threshold Functions. In *FOCS*

'96 : Proceedings of the 37th Annual Symposium on Foundations of Computer Science, pages 330–338, Washington, DC, USA. IEEE Computer Society.

[Blum and Mitchell, 1998] Blum, A. and Mitchell, T. (1998). Combining Labeled and Unlabeled Data with Co-Training. In *COLT' 98 : Proceedings of the eleventh annual conference on Computational learning theory*, pages 92–100, New York, NY, USA. ACM.

[Boeckmann et al., 2003] Boeckmann, B., Bairoch, A., Apweiler, R., Blatter, M.-C., Estreicher, A., Gasteiger, E., Martin, M. J., Michoud, K., O'Donovan, C., Phan, I., Pilbout, S., and Schneider, M. (2003). The SWISS-PROT protein knowledgebase and its supplement TrEMBL in 2003. *Nucleic Acids Res.*, 31(1) :365–370.

[Bonneau and Baker, 2001] Bonneau, R. and Baker, D. (2001). Ab initio Protein Structure Prediction : Progress and Prospects. *Annual Rev. Biophys. Biomol. Struct.*, 30 :173–189.

[Bonneau et al., 2001] Bonneau, R., Tsai, J., Ruczinski, I., Chivian, D., Rohl, C., Strauss, C. E. M., and Baker, D. (2001). Rosetta in CASP4 : Progress in ab initio protein structure prediction. *Proteins*, 45(Suppl. 5) :119–126.

[Bowie et al., 1991] Bowie, J., Luthy, R., and Eisenberg, D. (1991). A method to identify protein sequences that fold into a known three-dimensional structure. *Science*, 253(5016) :164–170.

[Bradley et al., 2003] Bradley, P., Chivian, D., Meiler, J., Misura, K. M., Rohl, C. A., Schief, W. R., Wedemeyer, W. J., Schueler-Furman, O., Murphy, P., Schonbrun, J., Strauss, C. E., and Baker, D. (2003). Rosetta predictions in CASP5 : Successes, failures, and prospects for complete automation. *Proteins*, 53(Suppl. 6) :457–468.

[Bradley et al., 2001] Bradley, P., Cowen, L., Menke, M., King, J., and Berger, B. (2001). Predicting the β-Helix Fold from Protein Sequence Data. In *RECOMB 2001 : Proceedings of the 5th annual international conference on Computational biology*, pages 59–67, New York, NY, USA. ACM.

[Branden et al., 1996] Branden, C., Tooze, J., and Lubochinsky, B. (1996). *Introduction à la structure des protéines.* Editions DeBoeck Université, Bruxelles.

[Breiman et al., 1984] Breiman, L., Friedman, J., Olshen, R., and Stone, C. J. (1984). Classification and Regression Trees. Technical report, Wadsworth & Brooks/Cole Advanced Books & Software, Pacific Grove, CA.

[Bylander, 1994] Bylander, T. (1994). Learning Linear Threshold Functions in the Presence of Classification Noise. In *COLT '94 : Proceedings of the seventh annual conference on Computational learning theory*, pages 340–347, New York, NY, USA. ACM.

[Bylander, 1998] Bylander, T. (1998). Learning noisy linear threshold functions.

[Ceroni et al., 2003] Ceroni, A., Frasconi, P., Passerini, A., and Vullo, A. (2003). Predicting the Disulfide Bonding State of Cysteines with Combinations of Kernel Machines. *J. VLSI Signal Processing Systems*, 35(3) :287–295.

[Ceroni et al., 2006] Ceroni, A., Passerini, A., Vullo, A., and Frasconi, P. (2006). DISULFIND : a Disulfide Bonding State and Cysteine Connectivity Prediction Server. *Nucleic Acids Res.*, 34(Suppl. 2) :177–181.

[Chapelle et al., 2003] Chapelle, O., Scholkopf, B., and Weston, J. (2003). Semi-Supervised Learning Through Principal Directions Estimation. In *Proceedings of the ICML Workshop : The Continuum from Labeled to Unlabeled Data.*

[Cheng and Baldi, 2005] Cheng, J. and Baldi, P. (2005). Three-Stage Prediction of Protein β-Sheets by Neural Networks, Alignments and Graph Algorithms. *Bioinformatics*, 21(Suppl. 1) :75–84.

[Cheng and Baldi, 2007] Cheng, J. and Baldi, P. (2007). Improved Residue Contact Prediction Using Support Vector Machines and a Large Feature Set. *Bioinformatics*, 8(1) :113.

[Cheng et al., 2006] Cheng, J., Saigo, H., and Baldi, P. (2006). Large-Scale Prediction of Disulphide Bridges Using Kernel Methods, Two-Dimensional Recursive Neural Networks, and Weighted Graph Matching. *Proteins*, 62(3) :617–629.

[Chothia and Lesk, 1986] Chothia, C. and Lesk, A. (1986). The relation between the divergence of sequence and structure in proteins. *EMBO J.*, 5(4) :823–826.

[Corduneanu and Jaakkola, 2001] Corduneanu, A. and Jaakkola, T. (2001). Stable Mixing of Complete and Incomplete Information. Technical report, Massachusetts Institute of Technology - Artificial Intelligence Laboratory.

[Cornuéjols et al., 2002] Cornuéjols, A., Miclet, L., Kodratoff, Y., and Mitchell, T. (2002). *Apprentissage artificiel : Concepts et algorithmes*. Eyrolles.

[Cozman et al., 2003] Cozman, F. G., Cohen, I., and Cirelo, M. C. (2003). Semi-supervised Learning of Mixture Models and Bayesian Networks. In *Proceedings of the Twentieth International Conference on Machine Learning*, pages 99–106. AAAI Press.

[De Comité et al., 1999a] De Comité, F., Denis, F., Gilleron, R., and Letouzey, F. (1999a). Comment améliorer l'apprentissage en utilisant des exemples positifs et non étiquetés. In *Actes de la Conférence Francophone sur l'Apprentissage*, pages 133–144.

[De Comité et al., 1999b] De Comité, F., Denis, F., Gilleron, R., and Letouzey, F. (1999b). Positive and Unlabeled Examples Help Learning. In *ALT '99 : Proceedings of the 10th International Conference on Algorithmic Learning Theory*, pages 219–230, London, UK. Springer-Verlag.

[Decatur, 1997] Decatur, S. E. (1997). PAC Learning With Constant-Partition Classification Noise and Applications to Decision Tree Induction. In *ICML '97 : Proceedings of the Fourteenth International Conference on Machine Learning*, pages 83–91, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

[DeLano, 2002] DeLano, W. (2002). The PyMOL Molecular Graphics System.

[Dempster et al., 1977] Dempster, A., Laird, N., and Rubin, D. (1977). Maximum Likelihood from Incomplete Data via the EM Algorithm. *J. Royal Stat. Soc.*, 39(1) :1–38.

[Denis, 1998] Denis, F. (1998). PAC Learning from Positive Statistical Queries. In *ALT '98 : Proceedings of the 9th International Conference on Algorithmic Learning Theory*, pages 112–126, London, UK. Springer-Verlag.

[Denis et al., 2002a] Denis, F., Gilleron, R., and Tommasi, M. (2002a). Classification de textes et co-training à partir de textes positifs et non étiquetés. In *Actes de la Conférence Francophone sur l'Apprentissage*, pages 205–220.

[Denis et al., 2002b] Denis, F., Gilleron, R., and Tommasi, M. (2002b). Text classification from positive and unlabeled examples. In *Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems : IPMU 2002*, pages 1927–1934.

[Denis et al., 2003] Denis, F., Laurent, A., Gilleron, R., and Tommasi, M. (2003). Text Classification and co-training from Positive and Unlabeled Examples. In *Proceedings of the ICML 2003 Workshop : The Continuum from Labeled to Unlabeled Data*, pages 80–87.

[Denis et al., 2006a] Denis, F., Magnan, C. N., and Ralaivola, L. (2006a). Apprentissage de classifieurs naïfs de Bayes à partir de données soumises à un bruit de classification conditionnel à chaque classe. In *CAp '06 : Actes de la Conférence Francophone sur l'Apprentissage*, pages 251–266, Grenoble. PUG.

[Denis et al., 2006b] Denis, F., Magnan, C. N., and Ralaivola, L. (2006b). Efficient learning of Naive Bayes classifiers under class-conditional classification noise. In *ICML '06 : Proceedings of the 23rd International Conference on Machine Learning*, pages 265–272, New York, NY, USA. ACM Press.

[Dietterich, 1998] Dietterich, T. G. (1998). Approximate Statistical Test For Comparing Supervised Classification Learning Algorithms. *Neural Computation*, 10(7) :1895–1923.

[Domingos and Pazzani, 1996] Domingos, P. and Pazzani, M. J. (1996). Beyond Independence : Conditions for the Optimality of the Simple Bayesian Classifier. In *International Conference on Machine Learning*, pages 105–112.

[Domingos and Pazzani, 1997] Domingos, P. and Pazzani, M. J. (1997). On the Optimality of the Simple Bayesian Classifier under Zero-One Loss. *Machine Learning*, 29(2-3) :103–130.

[Dunagan and Vempala, 2004] Dunagan, J. and Vempala, S. (2004). A Simple Polynomial-time Rescaling Algorithm for Solving Linear Programs. In *STOC '04 : Proceedings of the thirty-sixth annual ACM symposium on Theory of computing*, pages 315–320, New York, NY, USA. ACM.

[Errami et al., 2003] Errami, M., Geourjon, C., and Deléage, G. (2003). Conservation of amino acids into multiple alignments involved in pairwise interactions in three-dimensional protein structures. *J. Bioinform. Comput. Biol.*, 1(3) :505–520.

[Eyheramendy et al., 2003] Eyheramendy, S., Lewis, D. D., and Madigan, D. (2003). On the Naive Bayes Model for Text Categorization. In *Ninth International Workshop on Artificial Intelligence and Statistics*.

[Fariselli and Casadio, 2001] Fariselli, P. and Casadio, R. (2001). Prediction of Disulfide Connectivity in Proteins. *Bioinformatics*, 17(10) :957–964.

[Fariselli et al., 2002] Fariselli, P., Martelli, P. L., and Casadio, R. (2002). A Neural Network-Based Method for Predicting the Disulfide Connectivity in Proteins. In *Proceedings of KES 2002, Knowledga based intelligent information engineering systems and allied technologies*, pages 464–468, Amsterdam. IOS Press.

[Fariselli et al., 1999] Fariselli, P., Riccobelli, P., and Casadio, R. (1999). Role of Evolutionary Information in Predicting the Disulfide-Bonding State of Cysteine in Proteins. *Proteins*, 36(3) :340–346.

[Ferrè and Clote, 2005a] Ferrè, F. and Clote, P. (2005a). DiANNA : a Web Server for Disulfide Connectivity Prediction. *Nucleic Acids Res.*, 33(Suppl. 2) :230–232.

[Ferrè and Clote, 2005b] Ferrè, F. and Clote, P. (2005b). Disulfide Connectivity Prediction using Secondary Structure Information and Diresidue Frequencies. *Bioinformatics*, 21(10) :2336–2346.

[Ferrè and Clote, 2006] Ferrè, F. and Clote, P. (2006). DiANNA 1.1 : an Extension of the DiANNA Web Server for Ternary Cysteine Classification. *Nucleic Acids Res.*, 34(Suppl. 2) :182–185.

[Fiser and Simon, 2000] Fiser, A. and Simon, I. (2000). Predicting the Oxidation State of Cysteines by Multiple Sequence Alignment. *Bioinformatics*, 16(3) :251–256.

[Frasconi et al., 2002] Frasconi, P., Passerini, A., and Vullo, A. (2002). A Two-Stage SVM Architecture for Predicting the Disulfide Bonding State of Cysteines. In *Proc. of the IEEE Workshop on Neural Networks for Signal Processing*, pages 25–34. IEEE Press.

[Frishman and Argos, 1996] Frishman, D. and Argos, P. (1996). Incorporation of Non-Local Interactions in Protein Secondary Structure Prediction from the Amino Acid Sequence. *Protein Eng.*, 9(2) :133–142.

[Fung and Lu, 2006] Fung, G. P. C. and Lu, H. (2006). Text Classification without Negative Examples Revisit. *IEEE Transactions on Knowledge and Data Engineering*, 18(1) :6–20.

[Geiger et al., 2001] Geiger, D., Heckerman, D., King, H., and Meek, C. (2001). Stratified Exponential Families : Graphical Models and Model Selection. *Ann. Statist.*, 29(2) :505–529.

[Geourjon, 2005] Geourjon, C. (2005). Communication personnelle.

[Gibrat, 2005] Gibrat, J.-F. (2005). Communication personnelle.

[Goldberg, 2006] Goldberg, P. W. (2006). Some Discriminant-Based PAC Algorithms. *J. Machine Learning Res.*, 7 :283–306.

[Hardin et al., 2002] Hardin, C., Pogorelov, T. V., and Luthey-Schulten, Z. (2002). Ab initio protein structure prediction. *Current Opinion in Structural Biology*, 12(2) :176–181.

[Hastie et al., 2001] Hastie, T., Tibshirani, R., and Friedman, J. (2001). *The Elements of Statistical Learning*. Springer-Verlag, New York, NY, USA.

[Hendsch and Tidor, 1994] Hendsch, Z. S. and Tidor, B. (1994). Do Salt Bridges Stabilize Proteins ? A Continuum Electrostatic Analysis. *Protein Science*, 3(2) :211–226.

[Higgins and Taylor, 2000] Higgins, D. and Taylor, W. (2000). *Bioinformatics : Sequence, Structure and Databanks : A Practical Approach*. Oxford University Press, USA.

[Honig and Hubbell, 1984] Honig, B. H. and Hubbell, W. L. (1984). Stability of « Salt Bridges »in Membrane Proteins. *PNAS*, 81(17) :5412–5416.

[Human Genome Sequencing Consortium International, 2004] Human Genome Sequencing Consortium International (2004). Finishing the euchromatic sequence of the human genome. *Nature*, 431 :931–945.

[Humphrey et al., 1996] Humphrey, W., Dalke, A., and Schulten, K. (1996). VMD : visual molecular dynamics. *J. Mol. Graph.*, 14(1) :33–38.

[Hutchinson et al., 1998] Hutchinson, E., Sessions, R., Thornton, J., and Woolfson, D. (1998). Determinants of Strand Register in Antiparallel β-Sheets of Proteins. *Protein Science*, 7(11) :2287–2300.

[Kearns, 1993] Kearns, M. (1993). Efficient Noise-Tolerant Learning from Statistical Queries. In *STOC '93 : Proceedings of the twenty-fifth annual ACM symposium on Theory of computing*, pages 392–401, New York, NY, USA. ACM.

[Kearns and Li, 1988] Kearns, M. and Li, M. (1988). Learning in the Presence of Malicious Errors. In *STOC '88 : Proceedings of the twentieth annual ACM symposium on Theory of computing*, pages 267–280, New York, NY, USA. ACM.

[Kearns and Vazirani, 1994] Kearns, M. J. and Vazirani, U. V. (1994). *An Introduction to Computational Learning Theory*. MIT Press, Cambridge, MA.

[Kohavi, 1995] Kohavi, R. (1995). A Study of Cross-Validation and Bootstrap for Accuracy Estimation and Model Selection. In *Proceedings of the International Joint Conference on Artificial Intelligence - IJCAI'95'*, pages 1137–1145.

[Krogh and Riis, 1996] Krogh, A. and Riis, S. K. (1996). Prediction of Beta Sheets in Proteins. In Touretzky, D. S., Mozer, M. C., and Hasselmo, M. E., editors, *Advances in Neural Information Processing Systems*, pages 917–923, Cambridge, MA. MIT Press.

[Kulikova et al., 2007] Kulikova, T., Akhtar, R., Aldebert, P., Althorpe, N., Andersson, M., Baldwin, A., Bates, K., Bhattacharyya, S., Bower, L., Browne, P., Castro, M., Cochrane, G., Duggan, K., Eberhardt, R., Faruque, N., Hoad, G., Kanz, C., Lee, C., Leinonen, R., Lin, Q., Lombard, V., Lopez, R., Lorenc, D., McWilliam, H., Mukherjee, G., Nardone, F., Pilar Garcia Pastor, M., Plaister, S., Sobhany, S., Stoehr, P., Vaughan, R., Wu, D., Zhu, W., and Apweiler, R. (2007). EMBL Nucleotide Sequence Database in 2006. *Nucleic Acids Res.*, 35 :D16–D20.

[Kumar and Nussinov, 1999] Kumar, S. and Nussinov, R. (1999). Salt Bridge Stability in Monomeric Proteins. *J. Mol. Biol.*, 293(5) :1241–1255.

[Kumar and Nussinov, 2001] Kumar, S. and Nussinov, R. (2001). Fluctuations in Ion Pairs and their Stabilities in Proteins. *Proteins*, 43(4) :433–454.

[Langley et al., 1992] Langley, P., Iba, W., and Thompson, K. (1992). An Analysis of Bayesian Classifiers. In *Proceedings of the Tenth National Conference on Artificial Intelligence*, pages 223–228, San Jose, CA. AAAI Press.

[Letouzey et al., 2000] Letouzey, F., Denis, F., and Gilleron, R. (2000). Learning from Positive and Unlabeled Examples. In *ALT '00 : Proceedings of the 11th International Conference on Algorithmic Learning Theor*, pages 71–85, London, UK. Springer-Verlag.

[Lewis and Ringuette, 1994] Lewis, D. D. and Ringuette, M. (1994). A Comparison of Two Learning Algorithms for Text Categorization. In *SDAIR-94, 3rd Annual Symposium on Document Analysis and Information Retrieval*, pages 81–93.

[Li and Liu, 2003] Li, X. and Liu, B. (2003). Learning to Classify Texts Using Positive and Unlabeled Data. In *IJCAI-03, Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence*, pages 587–594. Morgan Kaufmann.

[Li and Liu, 2005] Li, X. and Liu, B. (2005). Learning from Positive and Unlabeled Examples with Different Data Distributions. In *ECML 2005, 16th European Conference on Machine Learning*, pages 218–229, London, UK. Springer-Verlag.

[Liu et al., 2003] Liu, B., Dai, Y., Li, X., Lee, W. S., and Yu, P. S. (2003). Building Text Classifiers Using Positive and Unlabeled Examples. In *ICDM '03 : Proceedings of the Third IEEE International Conference on Data Mining*, pages 179–186, Washington, DC, USA. IEEE Computer Society.

[Liu et al., 2002] Liu, B., Lee, W. S., Yu, P. S., and Li, X. (2002). Partially Supervised Classification of Text Documents. In *ICML '02 : Proceedings of the Nineteenth International Conference on Machine Learning*, pages 387–394, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

[Magnan, 2005] Magnan, C. N. (2005). Apprentissage semi-supervisé asymétrique et estimations d'affinités locales dans les protéines. In *CAp '05 : Actes de la Conférence Francophone sur l'Apprentissage*, pages 297–312, Grenoble. PUG.

[Magnan, 2006] Magnan, C. N. (2006). Asymmetrical Semi-Supervised Learning and Prediction of Disulfide Connectivity in Proteins. *Revue d'Intelligence Artificielle*, 20(6) :673–695.

[Magnan et al., 2007a] Magnan, C. N., Capponi, C., and Denis, F. (2007a). A Protocol to Detect Local Affinities Involved in Proteins Distant Interactions. In *2007 IEEE International Conference on Bioinformatics and Biomedicine*, pages 252–257. IEEE Computer Society.

[Magnan et al., 2007b] Magnan, C. N., Capponi, C., and Denis, F. (2007b). Un protocole de détection d'affinités locales dans les protéines. In *CAp '07 : Actes de la Conférence Francophone sur l'Apprentissage*, pages 299–300. Cépaduès Editions.

[Mandel-Gutfreund et al., 2001] Mandel-Gutfreund, Y., Zaremba, S. M., and Gregoret, L. M. (2001). Contributions of Residue Pairing to β-Sheet Formation : Conservation and Covariation of Amino Acid Residue Pairs on Antiparallel β-Strands. *J. Mol. Biol.*, 305(5) :1145–1159.

[Martelli et al., 2002a] Martelli, P. L., Fariselli, P., Malaguti, L., and Casadio, R. (2002a). Prediction of the Disulfide-Bonding State of Cysteines in Proteins at 88% Accuracy. *Protein Science*, 11(11) :2735–2739.

[Martelli et al., 2002b] Martelli, P. L., Fariselli, P., Malaguti, L., and Casadio, R. (2002b). Prediction of the Disulfide Bonding State of Cysteines in Proteins with Hidden Neural Networks. *Protein Eng.*, 15(12) :951–953.

[Mc Callum and Nigam, 1998] Mc Callum, A. and Nigam, K. (1998). A Comparison of Event Models for Naive Bayes Text Classification. In *Proceedings of the AAAI-98 Workshop on Learning for Text Categorization.*

[Merkel and Regan, 1998] Merkel, J. S. and Regan, L. (1998). Aromatic Rescue of Glycine in β Sheets. *Folding & Design*, 3(6) :449–456.

[Merkel et al., 1999] Merkel, J. S., Sturtevant, J. M., and Regan, L. (1999). Sidechain Interactions in Parallel β-Sheets : the Energetics of Cross-Strand Pairings. *Structure Fold Desc.*, 7(11) :1333–1343.

[Miller et al., 1996] Miller, R., Jones, D., and Thornton, J. (1996). Protein fold recognition by sequence threading : tools and assessment techniques. *FASEB J.*, 10(1) :171–178.

[Minor and Kim, 1994a] Minor, D. L. and Kim, P. S. (1994a). Context is a major determinant of β-sheet propensity. *Nature*, 371(6494) :264–267.

[Minor and Kim, 1994b] Minor, D. L. and Kim, P. S. (1994b). Measurement of the β-sheet-forming propensities of amino acids. *Nature*, 367(6404) :660–663.

[Minor and Kim, 1996] Minor, D. L. and Kim, P. S. (1996). Context-Dependent Secondary Structure Formation of a Designed Protein Sequence. *Nature*, 380(6576) :730–734.

[Mitchell, 1997] Mitchell, T. (1997). *Machine Learning.* McGraw-Hill International Ed.

[Moreno and Agarwal, 2003] Moreno, P. J. and Agarwal, S. (2003). An Experimental Study of EM-Based Algorithms for Semi-Supervised Learning in Audio Classification. In *Proceedings of the ICML Workshop : The Continuum from Labeled to Unlabeled Data.*

[Mucchielli-Giorgi et al., 2002] Mucchielli-Giorgi, M., Hazout, S., and Tufféry, P. (2002). Predicting the Disulfide Bonding State of Cysteines using Protein Descriptors. *Proteins*, 46(3) :243–249.

[Nigam and Ghani, 2000] Nigam, K. and Ghani, R. (2000). Analyzing the Effectiveness and Applicability of Co-Training. In *CIKM '00 : Proceedings of the ninth international conference on Information and knowledge management*, pages 86–93, New York, NY, USA. ACM.

[Nigam et al., 1998] Nigam, K., Mc Callum, A., Thrun, S., and Mitchell, T. (1998). Learning to Classify Text from Labeled and Unlabeled Documents. In *AAAI '98 : Proceedings of the fifteenth Conference of the American Association for Artificial Intelligence*, pages 792–799, Menlo Park, CA, USA. American Association for Artificial Intelligence.

[Nigam et al., 2000] Nigam, K., McCallum, A., Thrun, S., and Mitchell, T. (2000). Text Classification from Labeled and Unlabeled Documents Using E.M. *Machine Learning*, 39(2-3) :103–134.

[Quinlan, 1979] Quinlan, J. (1979). Discovering Rules by Induction from Large Collections of Examples. *Expert Systems in the Microelectronic age*, pages 168–201.

[Quinlan, 1993] Quinlan, J. R. (1993). *C4.5 : Programs for Machine Learning*. Morgan Kaufmann, San Mateo, CA.

[Ralaivola et al., 2006a] Ralaivola, L., Denis, F., and Magnan, C. N. (2006a). Bruits de classification constant et constant par morceaux : égalité des classes de concepts apprenables. In *CAp '06 : Actes de la Conférence Francophone sur l'Apprentissage*, pages 235–250, Grenoble. PUG.

[Ralaivola et al., 2006b] Ralaivola, L., Denis, F., and Magnan, C. N. (2006b). CN = CPCN. In *ICML '06 : Proceedings of the 23rd International Conference on Machine Learning*, pages 721–728, New York, NY, USA. ACM Press.

[Rama et al., 2005] Rama, J. G., Shilton, A., Parker, M., and Palaniswami, M. (2005). Disulphide Bridge Prediction using Fuzzy Support Vector Machines. In *IEEE Proceedings of Third International Conference on Intelligent Sensing and Information Processing*, pages 49–54.

[Rish, 2001] Rish, I. (2001). An Empirical Study of the Naive Bayes Classifier. In *IJCAI 2001 Workshop on Empirical Methods in Artificial Intelligence*.

[Rosenblatt, 1958] Rosenblatt, F. (1958). The Perceptron : a probabilistic model for information storage and organization in the brain. *Psychol. Rev.*, 65(6) :386–408.

[Ruan et al., 2005] Ruan, J., Wang, K., Yang, J., Kurgan, L. A., and Cios, K. J. (2005). Highly Accurate and Consistent Method for Prediction of Helix and Strand Content from Primary Protein Sequences. *Artif. Intell. Med.*, 35(1-2) :19–35.

[Sanger and Thompson, 1953] Sanger, F. and Thompson, E. O. (1953). The amino-acid sequence in the glycyl chain of insulin. I. The identification of lower peptides from partial hydrolysates. II. The investigation of peptides from enzymic hydrolysates. *Biochem. J.*, 53(3) :353–374.

[Sen, 2003] Sen, S. (2003). Statistical Analysis of Pair-Wise Compatibility of Spatially Nearest Neighbor and Adjacent Residues in α-Helix and β-Strands : Application to a Minimal Model for Secondary Structure Prediction. *Biophysical Chem.*, 103(1) :35–49.

[Shawe-Taylor and Cristianini, 2000] Shawe-Taylor, J. and Cristianini, N. (2000). *An introduction to Support Vector Machines and other kernel-based learning methods*. Cambridge University Press, Cambridge, U.K.

[Simons et al., 1999] Simons, K., Bonneau, R., Ruczinski, I., and Baker, D. (1999). Ab initio protein structure prediction of CASP III targets using ROSETTA. *Proteins*, Suppl. 3 :171–176.

[Smith and Regan, 1995] Smith, C. K. and Regan, L. (1995). Guidelines for Protein Design : The Energetics of β-Sheet Side Chain Interactions. *Science*, 270(5238) :980–982.

[Smith and Regan, 1997] Smith, C. K. and Regan, L. (1997). Construction and Design of β-Sheets. *Acc. Chem. Res.*, 30(4) :153–161.

[Smith et al., 1994] Smith, C. K., Withka, J. M., and Regan, L. (1994). A Thermodynamic Scale For the β-Sheet Forming Tendencies of the Amino Acids. *Biochemistry*, 33(18) :5510–5517.

[Smola et al., 2000] Smola, A., Bartlett, P., Scholkopf, B., and Schuurmans, D. (2000). *Advances in Large Margin Classifiers*. MIT Press, Cambridge, MA.

[Song et al., 2004] Song, J.-N., Wang, M.-L., Li, W.-J., and Xu, W.-B. (2004). Prediction of the Disulfide-Bonding State of Cysteines in Proteins based on Dipeptide Composition. *Biochemical and Biophysical Research Communications*, 318(1) :142–147.

[Stevens, 2003] Stevens, R. (2003). The cost and value of three-dimensional protein structure. *Drug Discovery World*, 4 :35–48.

[Steward and Thornton, 2002] Steward, R. E. and Thornton, J. M. (2002). Prediction of Strand Pairing in Antiparallel and Parallel β-Sheets Using Information Theory. *Proteins*, 48(2) :178–191.

[The UniProt Consortium, 2007] The UniProt Consortium (2007). The Universal Protein Resource (UniProt). *Nucleic Acids Res.*, 35(Database Issue) :D193–197.

[Tsai et al., 2005] Tsai, C.-H., Chen, B.-J., Chan, C.-h., Liu, H.-L., and Kao, C.-Y. (2005). Improving disulfide connectivity prediction with sequential distance between oxidized cysteines. *Bioinformatics*, 21(24) :4416–4419.

[Valiant, 1984] Valiant, L. G. (1984). A Theory of the Learnable. *Commun. ACM*, 27(11) :1134–1142.

[Vapnik, 1995] Vapnik, V. N. (1995). *The nature of statistical learning theory*. Springer-Verlag, New York, NY, USA.

[Vapnik, 1998] Vapnik, V. N. (1998). *Statistical Learning Theory*. Wiley Inter-Science, New-York, NY, USA.

[Vijayakumar and Zhou, 2001] Vijayakumar, M. and Zhou, H.-X. (2001). Salt Bridges Stabilize the Folded Structure of Barnase. *J. Phys. Chem.*, 105(30) :7334–7340.

[Vullo and Frasconi, 2003] Vullo, A. and Frasconi, P. (2003). A Recursive Connectionist Approach for Predicting Disulfide Connectivity in Proteins. In *SAC '03 : Proceedings of the 2003 ACM symposium on Applied computing*, pages 67–71, New York, NY, USA. ACM.

[Vullo and Frasconi, 2004] Vullo, A. and Frasconi, P. (2004). Disulfide Connectivity Prediction using Recursive Neural Networks and Evolutionary Information. *Bioinformatics*, 20(5) :653–659.

[Whiley and Titterington, 2002] Whiley, M. and Titterington, D. (2002). Model Identifiability in Naive Bayesian Networks. Technical report, University of Glasgow.

[Wouters and Curmi, 1995] Wouters, M. A. and Curmi, P. M. (1995). An Analysis of Side Chain Interactions and Pair Correlations Within Antiparallel β-Sheets : The Differences Between Backbone Hydrogen-Bonded and Non-Hydrogen-Bonded Residue Pairs. *Proteins*, 22(2) :119–131.

[Xu et al., 1997] Xu, D., Tsai, C.-J., and Nussinov, R. (1997). Hydrogen Bonds and Salt Bridges Across Protein-Protein Interfaces. *Protein Eng.*, 10(9) :999–1012.

[Yakowitz and Spragins, 1968] Yakowitz, S. J. and Spragins, J. D. (1968). On the Identifiability of Finite Mixtures. *Ann. Math. Statist.*, 39(1) :209–214.

[Yang et al., 2003] Yang, Y., Xia, Y., Chi, Y., and Muntz, R. R. (2003). Learning naive bayes classifier from noisy data. Technical report, UCLA.

[Yu et al., 2002] Yu, H., Han, J., and Chang, K. C.-C. (2002). PEBL : Positive Example Based Learning for Web Page Classification Using SVM. In *KDD '02 : Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 239–248, New York, NY, USA. ACM Press.

[Yu et al., 2004] Yu, H., Han, J., and Chang, K. C.-C. (2004). PEBL : Web Page Classification without Negative Examples. *IEEE Transactions on Knowledge and Data Engineering*, 16(1) :70–81.

[Yu et al., 2003] Yu, H., Zhai, C. X., and Han, J. (2003). Text Classification from Positive and Unlabeled Documents. In *CIKM '03 : Proceedings of the twelfth international conference on Information and knowledge management*, pages 232–239, New York, NY, USA. ACM Press.

[Zaremba and Gregoret, 1999] Zaremba, S. M. and Gregoret, L. M. (1999). Context-dependence of Amino Acid Residue Pairing in Antiparallel β-Sheets. *J. Mol. Biol.*, 291(2) :463–479.

[Zhang et al., 1998] Zhang, C.-T., Lin, Z.-S., Zhang, Z., and Yan, M. (1998). Prediction of the Helix/Strand Content of Globular Proteins Based on their Primary Sequences. *Protein Eng.*, 11(11) :971–979.

[Zhao et al., 2005] Zhao, E., Liu, H.-L., Tsai, C.-H., Tsai, H.-K., Chan, C.-h., and Kao, C.-Y. (2005). Cysteine separations profiles on protein sequences infer disulfide connectivity. *Bioinformatics*, 21(8) :1415–1420.

[Zhu et al., 2003a] Zhu, X., Lafferty, J., and Ghahramani, Z. (2003a). Combining Active Learning and Semi-Supervised Learning Using Gaussian Fields and Harmonic Functions. In *Proceedings of the ICML Workshop : The Continuum from Labeled to Unlabeled Data.*

[Zhu et al., 2003b] Zhu, X., Wu, X., and Chen, Q. (2003b). Eliminating Class Noise in Large Datasets. In *Proceedings of the 20th ICML International Conference on Machine Learning*, pages 920–927.

« Apprentissage à partir de données diversement étiquetées pour l'étude du rôle de l'environnement local dans les interactions entre acides aminés »

Résumé. Nous étudions le problème bioinformatique de la prédiction de contacts ponctuels entre résidus distants sur la séquence d'une protéine, tels que les ponts disulfures ou salins, une étape encore non résolue du problème plus général de la prédiction de la structure 3D d'une protéine à partir de sa séquence primaire. L'étude de l'état de l'art sur ce problème a fait ressortir des questions sur la modélisation de ce problème ainsi que sur le rôle de l'environnement local des acides aminés appariés dans la formation de ces contacts. Plusieurs considérations biologiques d'une part, et des expérimentations d'autres part, montrent la nécessité d'étudier des contextes d'apprentissage jusqu'ici peu connus et peu étudiés pour répondre à ces questions. Le premier est un cas particulier de l'apprentissage semi-supervisé binaire dans lequel on suppose que les exemples classés dont on dispose appartiennent uniquement à une seule classe, nous l'appelons *apprentissage semi-supervisé asymétrique*. Le second cadre d'apprentissage étudié est une extension de l'apprentissage avec bruit de classification, noté *CN*, dans lequel on suppose que les données de chacune des deux classes sont corrompues par un bruit de classification constant par classe avant d'être observées, nous notons ce modèle de bruit *CCCN*. Nous montrons que ces deux contextes d'apprentissage sont mal posés dans le cadre général de l'apprentissage statistique, mais que certaines hypothèses sur les distributions sous-jacentes permettent de les rendre bien posés, comme par exemple l'hypothèse que les distributions conditionnelles à chacune des classes sont des distributions produits. Des adaptations de méthodes connues de l'apprentissage à ces contextes sont proposées : l'algorithme naïf de Bayes et l'algorithme du perceptron. Ces nouveaux algorithmes ont été expérimentés puis utilisés pour tenter de répondre aux questions biologiques initialement posées.

Mots clés : prédiction de la structure 3D des protéines, ponts disulfures et salins, affinité locale, apprentissage statistique supervisé et semi-supervisé, bruit de classification.

Abstract. The tridimensional structure of proteins is constrained or stabilized by some interactions between distant amino acids in the primary sequences. An accurate prediction of these bonds by machine learning methods may reduce the set of feasible conformations for a protein and may be an important step forward for the prediction of the 3D structure from primary sequences. A review of the literature raises questions about the role of the neighbourhood of bonded amino acids in the formation of these bonds and about the modeling of this problem. Some biological considerations and experiments show that we have to investigate uncommon learning frameworks to answer these questions. The first one, that we call *asymmetrical semi-supervised learning*, is a particular case of binary semi-supervised learning, in which the only labelled data to learn from belong to one class, and the second one considers that the data at hand are subject to *class-conditional classification noise* (*CCCN*), a generalization of the *uniform classification noise* (*CN*). We show that learning in these frameworks leads to ill-posed problems. We give some assumptions that make these problems well-posed. We propose adaptations of well-known learning methods, namely naive Bayes algorithm and the perceptron, to these learning frameworks. We test these methods and apply them to try to answer the questions on the biological problem considered in this study.

Keywords : proteins 3D structure prediction, disulfide and salt bridges, local affinities, statistical supervised and semi-supervised learning, classification noise.

Thèse préparée par Christophe N. Magnan au sein du
Laboratoire d'Informatique Fondamentale de Marseille
39, rue F. Joliot-Curie, F-13453 Marseille Cedex 13.