UNIVERSITÉ DE LORRAINE

École doctorale IAEM Lorraine

Impact du Bruit d'Annotation sur l'Évaluation de Classifieurs

THÈSE

présentée et soutenue publiquement le 22 Novembre 2021

pour l'obtention du

Doctorat de l'Université de Lorraine

(mention informatique)

par

Ilias Benjelloun

Composition du jury

Rapporteurs :	Véronique Eglin Céline Hudelot
Examinateurs :	Marianne Clausel Benoît Frénay
Directeurs de thèse :	Efoevi Angelo Koudou Bart Lamiroy

Laboratoire Lorrain de Recherche en Informatique et ses Applications — UMR 7503

Mis en page avec la classe thesul.

Remerciements

Je tiens avant tout à remercier chaleureusement mon directeur de thèse, Bart Lamiroy, ainsi que mon codirecteur de thèse, Angelo Efoevi Koudou, pour leur soutien et leur bonne humeur durant ces années, et en particulier pour leurs précieux conseils lors de la rédaction de ce document. J'adresse des remerciements particuliers à Marianne Clausel, pour l'aide qu'elle m'a apportée au moment où j'en avais besoin.
Ce travail n'aurait pas vu le jour sans les financements de la région Grand-Est et de la Fédération Charle-Hermite, qui m'ont permis de me consacrer à ma recherche. Je remercie également en ce sens l'Université de Lorraine, pour avoir accepté d'étendre mon financement pendant quelques mois en réaction à la situation sanitaire que nous avons subie.
J'adresse toute ma reconnaissance à mes collègues de travail ; Louis Viard, pour son assistance dans les moments critiques, et sans qui je n'aurais certainement pas atteint la ligne d'arrivée ; Pierre-Antoine Rault, pour les heures studieuses passées dans notre bureau commun, et surtout pour celles qui l'étaient moins ; Christophe Cerisara, Claire Gardent, Yannick Parmentier pour leur présence rassurante et les avis pleins de recul dont ils ont pu me faire part ; Anastasia Shimorina, Hoa Thien Le, Hubert Nourtel, Guillaume Leberre, Emilie Colin et tous les autres membres de l'équipe Synalp pour les bons moments que nous avons passés ensemble. J'exprime enfin une pensée particulière envers Maxime, Samy, Rayan, et les autres amis précieux qui continuent à vivre dans mes souvenirs, malgré que nos chemins se soient séparés.

Je dédie cette thèse
à ma famille.
Mes parents, ma soeur,
mon oncle et ma tante,
et ceux qui m'auront permis de me relever.

Sommaire

Introduction Générale

Cet ouvrage compile les réflexions et travaux menés sur trois années de doctorat pour l'obtention du grade de docteur en informatique et statistiques, effectué au Loria, centre de recherche en informatique et applications de Nancy, dans le cadre du projet de financement de thèses multidisciplinaires de la Fédération Charles-Hermite. Cette thèse, dirigée par Bart Lamiroy, Professeur membre associé du Loria, et codirigée par Angelo Efoevi-Koudou, Maître de Conférences HDR à l'Institut Elie-Cartan, porte sur la problématique de l'évaluation de la performance des systèmes de classification construits par apprentissage lorsque la qualité des données de test est entachée par des erreurs d'annotation, et par conséquent pose la question de la fiabilité des décisions s'appuyant sur cette évaluation.

Ces systèmes sont généralement construits par apprentissage automatique, un champ de recherche qui a connu ses débuts lors du siècle dernier, et qui n'eut de cesse de se développer à une vitesse impressionnante jusqu'à nos jours. Il regroupe désormais de nombreux sous-domaines, et des applications diverses dans quasiment tous les secteurs. De ce fait, proposer une étude de la question posée par le sujet de thèse qui serait valable pour l'ensemble de ces domaines et applications dépasse largement les ambitions de cet ouvrage. Dans le chapitre 1, nous commençons par introduire les principes généraux de l'apprentissage automatique, puis nous posons les frontières du domaine sur lequel notre réflexion porte, i.e. la classification supervisée.

Un élément essentiel à la construction de systèmes de classification efficaces est d'avoir une idée suffisamment précise de leur performance. De nombreuses techniques ont été mises au point pour estimer cette performance de différentes manières. La grande majorité de ces techniques entre dans ce que l'on appelle l'évaluation empirique, une pratique n'étant pas restreinte au champ de l'apprentissage automatique. Dans le chapitre 2, nous présentons ce qu'est l'évaluation empirique, pourquoi et sous quelles formes elle est généralement appliquée, puis nous insistons sur son utilisation dans le cadre de l'évaluation de classifieurs.

Nous nous plaçons dans le contexte de l'apprentissage supervisé. La construction d'un système par apprentissage automatique se fond sur l'analyse d'un ensemble de données spécifiant la tâche devant être apprise. Lorsque ces données sont accompagnées d'étiquettes correspondant à une interprétation vers laquelle le système doit idéalement tendre, l'apprentissage est dit « supervisé ». Un problème de longue date qui accompagne l'apprentissage supervisé est le bruit d'annotation : les annotateurs peuvent en effet fournir la mauvaise interprétation pour certaines données, ce qui modifie en partie la spécification de la tâche qui transparaît de l'ensemble de données obtenu. Dans le chapitre 3, nous présentons en quoi cela peut être problématique, non seulement pour l'apprentissage des systèmes mais également pour l'évaluation que l'on en fait, et quelles solutions sont envisageables pour combattre ces effets négatifs.

Nous abordons aussi notre sujet du point de vue du cadre législatif d'application des systèmes d'IA. Notre société connaît depuis quelques années une transformation progressive très rapide due à la mise en place de tels systèmes dans différents secteurs. Le comportement de ces systèmes a par conséquent un impact grandissant sur le quotidien des citoyens. Cela pose des questions

éthiques et légales quant à leur utilisation, et les réponses actuellement apportées sont loin d'être satisfaisantes, étant donné que nos sociétés ont fait le choix de privilégier le profit naissant de la multiplication de ces systèmes au détriment du développement de garde-fous. Une conséquence directe est la construction rapide de tels systèmes à l'aide de données en grande quantité, mais dont la qualité n'a pas été contrôlée, et pouvant donc présenter un nombre non-négligeable d'erreurs d'annotation. Le chapitre 4 tente de présenter ces problématiques en faisant le lien entre bruit d'annotation dans les données d'un côté, et comportement indésirable des systèmes construits *et évalués* à l'aide de ces données de l'autre.

Nous présentons ensuite dans le chapitre 5 les pistes envisageables pour mener l'évaluation de classifieurs en prenant en compte la présence de bruit d'annotation. Puis, dans le chapitre 6, nous développons l'un de ces points : la connaissance formelle de l'impact du bruit d'annotation sur le processus d'évaluation. Nous illustrons l'apport de cette connaissance sur le processus d'évaluation par un cas applicatif de comparaison d'un grand nombre de classifieurs différents entraînés pour de la classification d'images. Enfin, nous consacrons le chapitre 7 à la présentation de travaux annexes, menés dans le but d'analyser l'impact du bruit d'annotation sur des indicateurs utilisés pour l'élagage de réseaux de neurones. En particulier, nous montrons qu'un indicateur n'étant pas calculé de manière supervisé, i.e. ne nécessitant pas l'utilisation des annotations des données, n'est pas forcément robuste à la présence d'un bruit d'annotation.

Chapitre 1

L'apprentissage automatique

Sommaire

Ce chapitre introductif pose une partie du contexte dans lequel se place cette thèse. En effet, nous nous intéressons en particulier à une classe d'objets en apprentissage automatique : les classifieurs. Il nous parait nécessaire de bien définir le terme tel que nous l'entendons puisqu'il existe beaucoup d'approches de natures variées qui peuvent entrer sous la dénomination de « classifieur ». Notre problématique se restreint à de la classification dite simple, et ce chapitre a aussi pour objectif de présenter les motivations de ce choix. De plus, nous introduisons des concepts et définitions nécessaires au développement de cette thèse.

1.1 Évolution de la réflexion sur l'IA dans l'Histoire

L'apprentissage automatique est une discipline qui a vu le jour au milieu du 20$^{\text{ème}}$ siècle, grâce aux travaux de penseurs comme, entre autres, Allan Turing dans son essai « Computing Machinery and Intelligence » [Tur09], ainsi qu'à l'invention de l'ordinateur et sa puissance de calcul. De nos jours, allant de la Médecine à la Défense, du Droit à la Finance, de l'Industrie au quotidien du citoyen, les applications de cette discipline sont légions. Elle joue un rôle majeur dans ce qu'on appelle l'industrie 4.0, terme apparu depuis quelques années [Hen13].

Le concept de « machine intelligente » n'est pas récent. Dès l'antiquité, le thème de l'automate était déjà abordé par des auteurs tels qu'Homère, ou même véritablement simulé en Égypte par, entre autres, la statue du dieu Amon [Mas10], manipulée de l'intérieur en secret par des prêtres pour désigner du bras le nouveau pharaon. De nombreux essais et travaux sur les automates, et plus généralement sur la pensée automatique, ont vu le jour avant les réflexions d'Allan Turing [Lur+20 ; Tru15].

De la même manière, un autre concept essentiel à l'apprentissage automatique a commencé à occuper l'esprit de l'homme il y a longtemps : l'exploration de données. Les premiers recen-

sements, respectivement de la population et des récoltes, ont été effectué en Égypte antique et en Chine impériale, bien avant l'an 1 du calendrier grégorien. La volonté d'utiliser des données pour en extraire des caractéristiques communes apparut plus récemment au 17ème siècle. Une fois extraites, ces caractéristiques forment un modèle, permettant par conséquent de satisfaire une autre vieille ambition de l'homme : être capable de prédire son avenir.

L'apprentissage automatique est la synthèse de ces trois éléments principaux :

— la discipline constitue une avancée majeure pour les réflexions sur le concept de machines pensantes ;
— son application est fondée sur de l'exploration automatique de données grâce à un ordinateur ;
— elle permet de construire un modèle prédictif fondé sur les données analysées.

Le terme de *prédiction* sera par conséquent utilisé régulièrement pour parler de ce qu'accomplit une machine autonome. Cette appellation doit bien sûr être prise avec recul : nous ne parlons pas d'entité surnaturelle connaissant l'avenir, simplement d'un agrégat d'informations permettant de faire des prévisions sur un aspect du monde a priori inconnu.

1.2 Qu'est-ce que l'apprentissage automatique ?

Proposer une définition générale de l'apprentissage automatique n'est pas une mince affaire. Le problème se situe dans la signification des termes. *Qu'est-ce qu'apprendre* ? Peut-on dire d'une machine qu'elle apprend ? Le Larousse donne entre autres les définitions suivantes :

— *acquérir par l'étude*, par la pratique, par l'expérience une connaissance, un savoir-faire, *quelque chose d'utile* ;
— *enseigner* à quelqu'un *quelque chose*, lui faire acquérir une connaissance, un savoir-faire, une expérience.

Ces définitions, particulièrement les parties en italique, sont suffisamment vagues pour pouvoir a priori imaginer une machine étant le sujet qui apprend dans la première, ou l'objet à qui on apprend dans la seconde. Cependant, on peut argumenter que ces définitions concernent implicitement des entités douées de conscience, ce qui pour une machine est une question philosophique ardue, sujette à réflexion depuis plusieurs décennies. En particulier, apprendre est en général associé à l'intention d'apprendre, ce qui pose la question de l'existence d'une quelconque intention chez une machine. Sans aller jusqu'à donner un avis sur cette dernière question, nous pensons tout de même que le fait d'apprendre peut, même pour un humain, s'opérer sans que cette personne en ait nécessairement eu l'intention.

Plutôt que de poursuivre dans un débat philosophique sans fin, nous allons prendre un point de vue analogue à Witten et. al. dans [Wit+16a], et définir de manière opérationnelle l'apprentissage automatique par « le processus accompli par toutes choses ayant changé son *comportement* à l'aide *d'observations* pour améliorer sa *performance future* ». Cela ne constitue certainement pas une définition dépourvue de défauts, mais le fait qu'elle soit liée à la notion de performance est intéressant pour la suite de notre propos.

Malgré tout, cette définition reste pour le moins générale, et peut concerner des processus de natures très différentes. Pour préciser les choses, il est nécessaire de définir la nature des observations dont il est question, la forme que prend le comportement de la machine, ainsi que cette notion de performance. Cette dernière notion sera abordée en détail dans le chapitre 2. Pour l'instant, nous allons nous concentrer sur la définition des deux premiers points, ce qui nous permettra de montrer la variété des systèmes [1] existants au sein même de la discipline, et

1. L'utilisation du mot « système » peut paraître ambiguë, le terme étant assez vague. Dans le cadre de cette

par conséquent la nécessité de focaliser l'étude sur un domaine plus restreint. En premier lieu, nous allons préciser ce que sont les « données d'un problème d'apprentissage automatique », puis nous nous intéresserons dans un second temps au large panel de techniques développées pour le traitement de ces données.

1.3 La matière première de l'apprentissage automatique : la donnée

Une entité qui apprend a besoin d'observations, et par conséquent de capteurs lui permettant d'observer. Un nouveau-né se sert de ses sens pour observer son environnement, et apprendre à y vivre. De la même manière, les végétaux possèdent de nombreuses capacités sensorielles, et s'en servent pour apprendre et anticiper au sein de leur environnement d'évolution. Une machine peut également être dotée de capteurs sensoriels : ils peuvent être d'ordre visuels ou laser, sonores, haptiques ; les possibilités sont nombreuses. Cette variété de capteurs montre à quel point les observations effectuées peuvent être de types différents. Dans le contexte de l'apprentissage automatique, ces observations sont appelées de manière assez sobre des *données*. De nos jours, de tels données existent sous forme numérisées en grand nombre, et sont effectivement de types variés : texte ou image, audio ou vidéo, continu ou discret, numérique ou nominal.

La nomenclature utilisée pour ces différents types de données n'est pas toujours correctement définie au sein de la discipline. Différents auteurs ont parfois différentes appellations pour un même type de donnée, et les classifications de ces types peuvent être trop réduites pour représenter correctement la diversité des applications. Nous allons donc choisir d'utiliser une nomenclature que nous pensons pertinente, proposée dans [Hal18], qui sépare les données en 7 types de base :

inutile : les données qui ne portent pas de sens particulier pour le problème à résoudre, et qui peuvent être ignorées (e.g. les numéros d'identification unique générés aléatoirement, comme les numéros de comptes bancaires) ;

nominal : les données à valeurs discrètes parmi différentes catégories, lesquelles ne présentant aucune relation numérique, rendant le calcul d'une moyenne ou d'une médiane insensé (e.g. les espèces animales) ;

ordinal : des données numériques discrètes qui peuvent être ordonnées, mais pour lesquelles une notion de distance entre chaque valeur ne peut pas forcément être définie (e.g. le classement des coureurs du 100 mètres, où l'écart entre le premier et le deuxième peut être très différent de celui entre le deuxième et le troisième) ;

binaire : les données n'ayant que deux valeurs possibles, typiquement 0 ou 1 ; bien que pouvant être vu comme un cas particulier des types nominal, ordinal, cardinal ou continu, le type binaire est très répandu dans les problèmes d'apprentissage automatique, d'où le mérite de le traiter comme un type à part entière ;

cardinal : les données à valeurs entières positives permettant de compter une caractéristique particulière (e.g. le nombre d'occurrences d'un même mot dans un document) ;

temporel : les données évoluant selon une caractéristique temporelle (jour, semaine, année ...) ; communément appelées séries temporelles, elles sont souvent présentes dans des problèmes concernant la finance ou le marketing ;

thèse, lorsque nous parlons de système, nous désignons précisément un programme informatique construit par apprentissage automatique. Cela peut donc se rapporter à un classifieur, mais pas seulement.

continu : les données numériques pouvant prendre des valeurs arbitraires dans un intervalle particulier ;

Par ailleurs, on ajoute généralement 4 types supplémentaires à cette classification, étant donné l'ensemble de techniques spécialement développées pour les traiter :

texte : données discrètes présentant une notion de proximité spatiale selon une dimension, d'où la possibilité de les représenter sous une forme similaire aux séries temporelles, bien que d'autres représentations existent ;

audio : données continues avec une notion de proximité temporelle ;

image : données présentant une notion de proximité spatiale selon deux, voire trois dimensions ;

vidéo : séries temporelles de données de type image et audio ;

Un principe fondamental au sein de la discipline est de connaître au mieux les données d'un problème d'apprentissage automatique avant de se lancer dans sa résolution. Ce sont ces données, leur forme, leur nombre, leur distribution et les potentiels biais qu'elles contiennent qui déterminent aussi bien les techniques adaptées à leur traitement que le type de questions auxquelles il est envisageable de répondre en les utilisant. Ce dernier aspect est essentiel pour nos réflexions car c'est précisément la question posée qui fixe la forme de la réponse, et par conséquent la manière d'évaluer si cette réponse est satisfaisante ou non.

En plus des types de données employés, un autre élément possède une importance particulière sur la forme que prend l'évaluation à terme : le modèle d'apprentissage mis en œuvre. La prochaine partie fera la distinction entre les principaux modèles, et détaillera celui sur lequel nous focaliserons la suite de notre réflexion : le modèle d'apprentissage supervisé.

1.4 L'apprentissage supervisé

Les problèmes d'apprentissage sont en général divisés en quatre catégories : apprentissage *supervisé*, apprentissage *non supervisé*, apprentissage *semi-supervisé* et apprentissage *par renforcement*. L'apprentissage supervisé désigne ces problèmes où les données sont étiquetées de sorte que la réponse du système idéal soit connue au préalable pour chaque donnée. Dans le cas des problèmes d'apprentissage non supervisés et semi-supervisés, toutes ou au moins une partie des données ne sont pas étiquetées : cette réponse idéale n'est pas connue. Les méthodes appliquées visent alors en général à déterminer la structure sous-jacente de ces données non étiquetées, par exemple en les classant dans des *clusters*, i.e. des groupes de données, selon une mesure de proximité particulière. Enfin, l'apprentissage par renforcement désigne une catégorie de problèmes où la réponse idéale n'est pas nécessairement connue au préalable : la viabilité des réponses données par le système est plutôt examinée après coup, à partir de l'évolution de la situation, selon le comportement qu'il adopte.

Les manières d'évaluer l'efficacité de systèmes entraînés par apprentissage semi- ou non supervisé, ou par renforcement, peuvent être assez spécifiques et différentes de celles utilisées pour des systèmes construits par apprentissage supervisé. En effet, la nature des réponses fournies varie en fonction de la nature du problème d'apprentissage considéré. Dans le cadre de notre sujet de thèse, nous nous concentrons uniquement sur des problèmes d'apprentissage supervisé. Nous allons par ailleurs voir que, même au sein de ce champ particulier, la variété de techniques et procédés ayant été développés pour l'évaluation de modèles est suffisamment conséquente.

1.4.1 La classification

L'apprentissage supervisé est un modèle d'apprentissage où toutes les données à disposition sont annotées, c'est-à-dire qu'une étiquette précise la réponse que le système est censé fournir pour chaque donnée. Les réponses possibles sont définies au préalable. Elles appartiennent soit à un ensemble de valeurs continu, auquel cas on parle de *régression*, soit à un ensemble de valeurs discret : on parle alors de *classification*. Il existe également des applications où la réponse prend la forme de données structurées de manière complexe. On parle de *prédiction structurée*. A titre d'exemple, prédire la température qu'il fera dans les jours à venir est un problème de régression, reconnaître différents objets ou être vivants dans des images est un problème de classification, et prédire des réseaux d'interactions entre gènes est un problème de prédiction structurée. Bien que l'on parle dans tous les cas d'apprentissage supervisé, ces trois situations présentent tout de même des différences importantes, entre autres par rapport aux métriques d'apprentissage et d'évaluation utilisées, ou encore sur les formes de biais pouvant apparaître lors des différentes étapes du problème d'apprentissage. Pour cette raison, nous allons essentiellement nous concentrer dans la suite sur les problèmes de classification.

Commençons tout d'abord par définir avec un peu plus de précision certains termes. En apprentissage automatique, l'objectif est de construire un programme informatique en mesure d'effectuer une tâche, ou plus généralement répondre à une question, de manière autonome. Cette question est autrement appelée le *concept* à apprendre. Le programme est appelé, dans le cas le plus général, un *modèle d'inférence* ou simplement *modèle*. En effet, le but est d'apprendre à décrire le concept à l'aide d'un modèle capable d'expliquer les données du problème. En classification, expliquer les données correspond à les classifier correctement, ce programme est donc plus souvent nommé *classifieur*[2]. Un classifieur reçoit une question en entrée pour produire une réponse en sortie. Jusqu'à présent, nous appelions cette entrée une donnée, de manière assez imprécise. Nous allons dorénavant utiliser autant que possible l'appellation *instance*, bien que nous continuerons à nous servir de la première dans certains cas. Une instance du problème d'apprentissage est un ensemble d'informations ou *caractéristiques*, en pratique encodées sous la forme d'un vecteur, qui définit le contenu de la question posée au classifieur. On considère généralement que ces instances proviennent d'une même distribution, et sont indépendantes. Le classifieur choisit sa réponse parmi un ensemble fini et discret de *classes* ou *catégories*, *l'ensemble d'interprétation*. A chaque instance du problème est associée une *étiquette*, appartenant également à cet ensemble, et représentant l'interprétation de cette instance par un *oracle*, entité détenant la description du concept vers laquelle celle du classifieur doit tendre dans l'idéal. Cet oracle consiste en général en un ou plusieurs humains, que l'on nomme des *annotateurs*, et l'interprétation que font ces annotateurs de l'ensemble des instances du problème d'apprentissage forme ce qu'on appelle la *vérité-terrain*. Le couple instance-étiquette est appelé un *exemple du problème d'apprentissage*, ou tout simplement *exemple*.

Les étapes du problème d'apprentissage sont divisibles en trois phases : une première phase de collecte et préparation des exemples du problème d'apprentissage, une phase d'apprentissage où le modèle affine sa description du concept grâce aux exemples et à un *algorithme d'apprentissage*, et une phase de test où l'aptitude du modèle à *généraliser* est évaluée. Plus un modèle est capable d'interpréter correctement des instances du problème qui n'ont pas servi lors de la phase d'apprentissage, plus son aptitude à généraliser est bonne. À l'inverse, lorsqu'un modèle n'obtient pas de bons résultats sur de nouveaux exemples, alors qu'il montre une performance élevée sur

2. Dans la suite de cet ouvrage, il arrive que nous nous référions à un classifieur par le terme « modèle ». De plus, nous utilisons également l'appellation générale « système » lorsque le propos s'applique aussi bien à un classifieur qu'à un autre type de modèle

les exemples qu'il a déjà vus, on parle communément de *sur-apprentissage*. Ce phénomène peut survenir pour différentes raisons, qui varient selon le modèle d'inférence considéré [Haw04 ; Die95]. On remarque par ailleurs que l'élément central d'un problème d'apprentissage est l'ensemble des exemples : en effet, le bon déroulement des deux dernières phases dépend de la réussite de la première, i.e. de l'abondance des instances et la qualité de leurs annotations.

En ce qui concerne les modèles d'inférences, ils sont nombreux. En choisir un correspond à faire des hypothèses précises sur la manière d'aborder le problème, et cela implique d'être soumis à des biais particuliers. C'est pourquoi il n'existe en général pas de modèle qui soit meilleur que les autres une fois entraîné sur les exemples, tous problèmes confondus. Chaque ensemble de données peut être plus ou moins adapté au biais d'un modèle d'inférence. L'algorithme d'apprentissage diffère du modèle d'inférence car il correspond à la spécification de la manière dont l'apprentissage s'effectue. En ce sens, différents modèles d'inférence peuvent être appliqués dans le cadre d'un même algorithme d'apprentissage. Pour en nommer quelques-uns, voici différents modèles connus et largement utilisés :

— les machines à vecteur de support ;
— les arbres de décision ;
— la régression logistique ;
— la méthode des plus proches voisins ;
— les modèles de mixture gaussienne et autres méthodes statistiques ;
— les réseaux de neurones et leurs extensions, méthodes de l'apprentissage *profond*.

Nous ne détaillons pas le fonctionnement de chacun de ces modèles, le lecteur intéressé peut se référer au livre de Witten et. al. [Wit+16a]. Il est vrai que de nos jours, le succès des modèles provenant de l'apprentissage profond est remarquable, ayant permis la résolution de certains problèmes très complexes, même pour un humain. Les avantages de cette pratique sont indéniables, mais des méthodes différentes restent supérieures sur certains aspects. En particulier, le caractère « boîte noire » des réseaux de neurones est un sujet de recherche d'actualité : il est en effet difficile d'obtenir une explication de la réponse ou du comportement adopté par un tel réseau, ce qui peut être préjudiciable dans de nombreux cas réel. De plus, en pratique, l'entraînement de ce type de modèles nécessite souvent une très grande quantité de données, ce qui les rend non applicables dans certaines situations où la collecte est ardue. En général, le contexte et les contraintes d'un problème d'apprentissage déterminent quels sont le ou les modèles d'inférence les plus intéressants pour le résoudre.

1.4.2 Exemple d'un problème de classification binaire

Nous allons à présent introduire quelques notations via un exemple concret de classification. Supposons que l'on dispose d'une base de données de renseignements anamnestiques [3] sur plusieurs patients, ainsi que le diagnostic établi pour chaque patient par un professionnel. L'objectif est de construire, à partir de ces informations, un processus capable de déterminer de manière autonome le diagnostic de futurs patients. En pratique, un tel problème s'accompagne de nombreuses difficultés, e.g. plusieurs informations sont manquantes pour chaque patient, ou encore d'autres sont imprécises ou fausses. Nous allons cependant faire abstraction de ces points par soucis de simplicité. Le problème se modélise alors de la manière suivante :

— chaque patient correspond à une instance notée $\mathbf{x}$;
— cette instance est concrètement représentée par un vecteur de caractéristiques $(\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(d)})$; Ces caractéristiques peuvent être de type nominal (e.g. adresse), ordinal (e.g. classes

3. Terme médical désignant l'ensemble des signes permettant de connaître ce qui a précédé, donc d'établir un diagnostic.

d'âge), binaire (e.g. fumeur), ou plus généralement continu (e.g. taux d'hémoglobine) ;
— d est la *dimension* du problème de classification ;
— les diagnostics envisageables constituent l'ensemble d'interprétation $\mathcal{I}$ des instances du problème ;
— l'étiquette d'une instance $\mathbf{x}$ est notée $\mathbf{y}$, et le couple $(\mathbf{x}, \mathbf{y})$ forme un exemple du problème d'apprentissage ;
— le nombre d'exemples disponibles est noté N ;
— le modèle d'inférence à entraîner est noté $\mathcal{C}$;
— un ensemble d'exemples $\mathcal{E}$;
— l'algorithme d'apprentissage est une fonction $\mathcal{A}$ qui prend en entrée un modèle d'inférence $\mathcal{C}_1$ ainsi que $\mathcal{E}$, et qui donne en sortie un second modèle d'inférence $\mathcal{C}_2$, possédant naturellement une meilleure description du concept que $\mathcal{C}_1$.

Supposons que le concept à apprendre soit la grippe. Dans ce cas, $\mathcal{I}$ est un ensemble de deux catégories : une catégorie c_0 dans laquelle les patients non atteints de la grippe doivent se retrouver, et une catégorie c_1 pour les patients effectivement atteints. On parle alors de *classification binaire*.

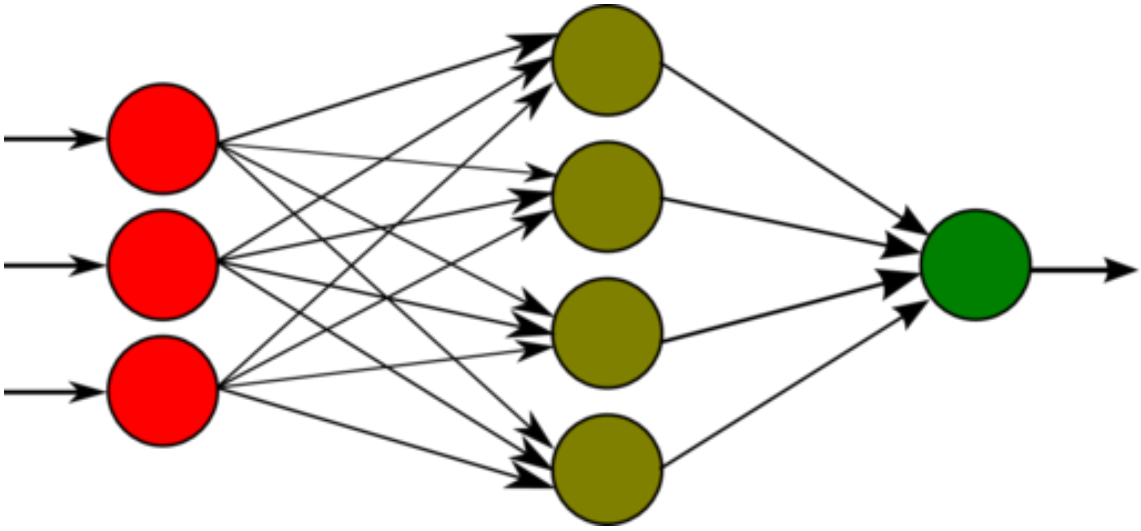


FIGURE 1.1 – Schéma d'un perceptron multicouches munie d'une couche cachée de quatre neurones, de trois neurones d'entrée et un neurone de sortie.

Nous considérons un réseau de neurones simple, un perceptron multicouches [Ros58], en guise de modèle d'inférence. Un perceptron multicouches (figure 1.1) est une structure en réseau où des « neurones » sont ordonnés en couches successives, chacun étant connecté à l'ensemble des neurones de la couche suivant la sienne. Chaque neurone est une fonction prenant un vecteur de nombre réels en entrée, et produisant une valeur réelle en sortie. Le vecteur d'entrée d'un neurone sur la couche k est composé des sorties des neurones qui lui sont connectés sur la couche $k-1$. Les arcs établissant la connexion entre la sortie des neurones de la couche $k-1$ et l'entrée d'un neurone sur la couche k sont munis d'une valeur numérique nommée *poids*. Ces valeurs servent à pondérer les sorties des neurones situés sur la couche $k-1$ dans le calcul de la sortie du neurone de la couche k. Ce sont les *paramètres* du réseau. Tout modèle d'inférence possède des paramètres de ce genre, qui déterminent entièrement ses réponses aux instances qui lui sont présentées. Ces instances $\mathbf{x}$ prennent en pratique la forme de vecteurs de caractéristiques numériques, et sont présentées au réseau en attribuant respectivement à chaque neurone sur la première couche la

valeur de chaque caractéristique. Ces valeurs sont alors propagées par l'application successive des fonctions correspondant à chaque neurone suivant, et fournissent à terme un vecteur numérique composé des sorties des neurones sur la couche finale. Les composantes de ce vecteur sont vues comme étant les scores respectifs des catégories de l'ensemble d'interprétation $\mathcal{I}$. Pour la situation de classification binaire entre les patients grippés et les patients saints, la dernière couche doit ainsi contenir deux neurones, un pour chaque score. Pour une instance $\mathbf{x}$, la catégorie au score le plus élevé correspond à la réponse donnée par le classifieur.

1.4.3 Extension à des problèmes de classification complexes

En plus de la classification binaire, il y a des problèmes de classification beaucoup moins simples. Supposons par exemple que la question soit de prédire si le patient est atteint d'une pathologie parmi n, disons une grippe, une gastro-entérite ou une angine. L'ensemble $\mathcal{I}$ possède alors trois catégories, ou plutôt quatre en incluant une dernière catégorie pour les patients atteints d'aucune de ces trois maladies. Ce type de problème est appelé *classification multi-classes*. La classification binaire en est un cas simplifié. Les problèmes de classification multi-classes présentent des difficultés supplémentaires, en particulier par rapport à la phase d'évaluation et au sens des métriques utilisées (cf. chapitre 2).

Le contexte peut encore devenir plus complexe. Par exemple, il semble assez irréaliste de catégoriser des patients malades sous l'étiquette d'une seule pathologie. Très souvent, un patient est atteint de plusieurs maux en même temps, ce qui nécessite un traitement adapté. Il est donc désirable d'être capable de prédire qu'un patient appartient à une catégorie ou plusieurs, selon les pathologies qui l'affectent. Dans ce cas, un exemple du problème n'est plus un couple instance-étiquette, mais plutôt un n-uplet regroupant l'instance et toutes les étiquettes pouvant lui être attribuées. Un patient pourrait donc être atteint d'une grippe et une gastro-entérite, un autre d'une angine et un dernier d'aucune de ces trois maladies. Ce problème est nommé *classification multi-étiquettes* et dispose également de ses particularités, aussi bien pour la phase d'apprentissage que pour la phase d'évaluation.

Pour finir, il existe des problèmes de classification encore plus sophistiqués, entre autres la *classification hiérarchique* et la *classification multi-instances*. Cette dernière, en particulier, concerne les problèmes où les instances ne peuvent être séparées en individus indépendants. Un exemple est alors formé de multiples instances dépendantes les unes des autres, accompagnées de leurs éventuelles annotations. On peut se retrouver dans ce cas lorsqu'on traite par exemple des données textuelles. Une tâche habituelle du traitement automatique du langage est de déterminer la catégorie grammaticale de chaque mot et particule dans une phrase. Si l'on considère qu'un mot représente une instance, avec son étiquette, i.e. sa catégorie grammaticale, il est en effet insensé d'essayer de prédire cette catégorie avec un modèle d'inférence qui considère chaque mot de la même phrase de manière indépendante. La tâche de prédiction doit être effectué en prenant en compte le contexte d'apparition de chaque instance. Bien que ce problème corresponde techniquement à de la classification, les modèles d'inférence et les métriques d'évaluation qui lui sont appliqués n'ont rien à voir avec un problème de classification classique.

1.5 Conclusion

Dans ce chapitre, nous avons vu une image d'ensemble du domaine de l'apprentissage automatique, et insisté sur la variété existante pour différents aspects : les types de données, les algorithmes d'apprentissage, les modèles d'inférence, ainsi que les situations d'apprentissage supervisé et de classification. Nous nous sommes concentrés en particulier sur les problèmes de

classification, pour lesquels nous avons introduit, à travers un exemple simple, une nomenclature et de multiples notations dont nous aurons l'usage tout au long de cet ouvrage. Par la suite, nous nous concentrons uniquement sur les problèmes de classification binaire et multi-classes, pour pouvoir aborder la problématique d'évaluation de modèles dans le cadre défini par notre thèse. Dans le prochain chapitre, nous verrons en détails pourquoi cette phase d'évaluation est nécessaire et comment elle est menée pour ces problèmes de classification.

Chapitre 2

L'évaluation

Sommaire

Un élément central de notre problématique de thèse est l'évaluation de classifieur, et ce chapitre en présente les fondamentaux. Nous abordons le sujet par des généralités sur la notion d'évaluation, et nous expliquons la différence entre évaluation empirique et rationnelle. Nous entrons ensuite en détails dans le sujet de l'évaluation en classification, en présentant le déroulement des procédures de test usuelles, ainsi que quelques mesures de performances fréquemment utilisées. Enfin, nous introduisons des outils statistiques permettant d'intégrer une notion de confiance (ou de risque) à l'évaluation de classifieurs : les intervalles de confiance et les tests d'hypothèse statistiques.

2.1 Notion d'évaluation dans différents domaines

L'évaluation est une notion omniprésente dans notre société, et a toujours été source de débat. Elle peut prendre de nombreuses formes, et servir différents objectifs, selon le cas d'application, mais elle est communément acceptée comme nécessaire au maintien et à l'évolution de toute structure ou organisme. Elle est parfois réalisée et subie par la même entité, auquel cas on parle de *self-évaluation*, ou alors appliquée par des entités d'autorité équivalentes dans le cas d'une *évaluation par des pairs*, ou enfin par des entités d'autorité supérieure. L'évaluation est spécifiée par un certain nombre de normes ou critères, pouvant représenter soit des *règles de conduite*, soit des *normes d'excellence*. Chaque individu possède des critères de self-évaluation, éventuellement implicites, lui permettant de maintenir autant que possible une direction de vie lui convenant. Il en est de même pour un groupe, une société, un pays, un organisme, lesquels mettent également en place des critères d'évaluation pour leurs membres et sous-systèmes. En somme, des entités de

nature très différente peuvent être sujette à évaluation. Même une cellule, ou plus généralement un système biologique, possède des propriétés d'homéostasie [4], procédure que l'on peut qualifier de self-évaluation, pour réguler son comportement. Un point essentiel sur lequel nous insistons tout au long de notre démarche est le caractère subjectif des critères de l'entité qui évalue, et nous le montrerons par plusieurs exemples, citations et développements dans les chapitres suivants (cf. chapitres 2, 3 et 4).

Les points de vue sur la notion d'évaluation varient selon les domaines. Plusieurs d'entre eux la réduisent en général à une mesure, entachée d'éventuels biais. C'est le cas de la docimologie, terme introduit par Henri Piéron [Pié54] pour parler de « l'étude systématique des examens (modes de notation, variabilité interindividuelle et intra-individuelle des examinateurs, facteurs subjectifs, etc.) ». En psychologie, en médecine ou encore en science des données, les procédures d'évaluation prennent cette forme, dans la grande majorité des cas. Dans d'autres cas, le point de vue est plus large, comme en sociologie, où le caractère arbitraire inhérent aux procédures d'évaluation est fréquemment considéré, ainsi que l'éventualité des contestations qui en résultent [Per89]. Si certains sociologues, comme Garcia et Montagne, insistent sur la nécessité globale des procédures d'évaluation, qui « peuvent aussi bien constituer un dispositif de contrôle et de mise en concurrence des individus au travail qu'un outil de connaissance de la réalité sociale [. . .] » [GM11], d'autres s'interrogent [Cha12] sur la pertinence de la forme sous laquelle elle est appliquée. C'est le cas, entre autres, de Michel Chauvière, lorsqu'il écrit, à propos de nouvelles formes d'évaluation qui se généralisent dans le domaine de l'intervention sociale : « Tellement saturée d'utilitarisme, la néo évaluation crée une illusion sur la réalité des pratiques et leur signification. Elle clôt à bon prix ce qui devrait rester ouvert à l'incertitude de la pensée » [Cha12]. Quoi qu'il en soit, si le point de vue porté sur l'évaluation dépend en grande partie du contexte dans lequel elle s'applique, deux aspects importants ressortent. Le premier est bien exprimé par les spécialistes des politiques publiques, qui envisagent l'évaluation comme « une des dimensions habituelles des processus de décision ». Le second se réfère à l'imperfection inévitable de toute procédure d'évaluation. En effet, il arrive malheureusement que les critères d'évaluation considérés ne soient pas en phase avec l'objectif initial. Pour peu que les décideurs n'en aient pas conscience, cela peut avoir de mauvaises conséquences, e.g. une entité passant avec succès l'évaluation mais comprenant des défauts restés invisibles.

Dans certains domaines, l'évaluation est pratiquée de manière empirique, c'est-à-dire qu'elle se fonde sur la confrontation de l'entité évaluée à des observations ou des résultats obtenues via des expériences précises. Dans ce cas, il est fréquent que l'évaluation soit réduite à une simple mesure. On parle d'*évaluation empirique*. En apprentissage automatique, nous appliquons essentiellement ce type d'évaluation. Mais avant d'aborder ce sujet, nous allons nous pencher plus en détails sur la signification du mot empirique.

2.2 Évaluation empirique ou rationnelle

En philosophie, on parle de « preuve empirique » pour désigner une information ou une connaissance acquise au moyen des sens, donc par l'observation et l'expérimentation [Pic11]. On peut également parler de connaissance a posteriori, par opposition aux connaissances a priori que l'on peut acquérir par l'utilisation de la pensée seule [KH05]. Le point de vue empirique est largement adopté dans toute discipline scientifique où une preuve empirique est nécessaire pour l'acception d'une hypothèse. Par opposition, le point de vue rationaliste place la raison ou la

4. Stabilisation, réglage chez les organismes vivants, de certaines caractéristiques physiologiques (pression artérielle, température, etc.).

réflexion comme seule preuve de la vérité ou de la fausseté d'une proposition [AA99].

Bien sûr, la pertinence de ces points de vue peut être questionnée. En effet, on peut argumenter que pour interpréter les résultats obtenus par un processus qualifié d'empirique, le raisonnement intervient. De la même manière, dans un processus dit rationnel, les observations et expériences passées ne sont pas indépendantes des mécanismes de raisonnement mis en œuvre. Il est néanmoins certain que le point de vue empirique accorde bien plus de poids aux observations, tandis que le point de vue rationaliste privilégie essentiellement la réflexion. Cette dichotomie de points de vue peut également s'appliquer à la notion d'évaluation.

Une évaluation peut en effet être menée principalement de manière rationnelle. C'est le cas pour un examinateur évaluant un étudiant par le dialogue pour déterminer la qualité de son raisonnement sur un sujet donné, ou encore pour un responsable des ressources humaines dans une entreprise qui interroge un candidat pour un poste quelconque. Dans les deux cas, le jugement de l'évaluateur est fortement dépendant de sa pensée, qui, bien sûr, reste influencée par les observations faites au cours de l'échange. À l'inverse, l'évaluation peut être essentiellement de nature empirique. Par exemple, il est fréquent, dans l'enseignement supérieur, pour certaines disciplines, de faire passer des examens ou des concours aux étudiants sous la forme d'un questionnaire à choix multiples dans le but de faciliter la correction, qui peut ainsi être réalisée automatiquement par ordinateur. Le jugement porté sur chaque étudiant dépend alors principalement des observations obtenues par la confrontation entre ses réponses et un corrigé établi au préalable.

Une différence majeure entre les deux points de vue est le nombre d'informations accessibles au sujet de l'entité évaluée. En effet, lorsqu'un processus empirique fixe au préalable des critères d'évaluation bien précis, et examine les entités évalués uniquement à travers les lunettes que sont ces critères, il est possible de manquer une information essentielle par rapport à l'entité évaluée, mais invisible à travers lesdites lunettes. À l'inverse, les informations à traiter dans le cadre d'une évaluation rationnelle sont bien plus nombreuses. D'un côté, cela limite le risque de manquer un aspect important qui n'avaient pas été prévus par les critères d'évaluation définis au préalable. En contrepartie, l'abondance de ces informations rend la tâche d'évaluation bien plus complexe, demandant du temps et des efforts considérables par rapport à la manière empirique.

De plus, pour les deux points de vue, des biais peuvent perturber le processus d'évaluation. En effet, le résultat d'une évaluation menée de manière rationnelle est vulnérable aux biais de pensée de l'évaluateur. Ces biais de pensée relèvent de la subjectivité inhérente à tout être humain, mais parfois, ils sont inadéquats par rapport à l'objectif initial du processus d'évaluation concerné. Par exemple, un recruteur pourrait décider de favoriser un candidat plutôt qu'un autre en se fondant sur des traits distinctifs de genre ou d'ethnie. Cet aspect peut être atténué en établissant des critères d'évaluation précis, et en limitant la tâche de l'évaluateur à *observer* la concordance entre l'évalué et ces critères. Dans ce cas, l'évaluation initialement rationnelle intègre un certain degré d'empirisme. Malgré tout, pour peu que les critères soient trop imprécis, laissant trop de libertés à l'évaluateur, ce dernier peut trouver le moyen d'affirmer ses avis biaisés, qui transparaîtront dans le résultat du processus d'évaluation. De plus, il n'est pas impossible que lesdits critères reflètent eux-mêmes ce genre de biais.

De l'autre côté, une évaluation essentiellement empirique peut conduire à des erreurs d'interprétation qui aurait été plus aisément évitées par une évaluation rationnelle. Par exemple, prenons le cas d'un QCM dans le cadre d'un examen de médecine, où les questions présentent des patients avec une série de symptômes, et les réponses proposent des diagnostics potentiels. Il peut arriver qu'une ou plusieurs questions soient en apparence évidente, mais manquent de donner certaines informations sur le patient complexifiant largement la situation, et susceptibles de modifier le diagnostic du tout au tout. Le cas échéant, l'étudiant qui n'a pas connaissance de cette complexité sous-jacente choisirait la réponse évidente, tandis que l'étudiant envisageant

tous les cas potentiels redouterait un piège, et choisirait donc une réponse différente. Dans le cas où la réponse évidente correspond à celle des correcteurs, l'étudiant le moins bon serait alors favorisé par l'évaluation. En résumé, les observations faites dans le cadre de cette évaluation empirique ne reflètent pas la réalité de la compétence des deux étudiants, et cela est dû à des critères d'évaluation mal posés ou incomplets, ici les questions du QCM. Ainsi, le résultat d'une évaluation empirique peut inclure des erreurs d'interprétation.

On remarque par ailleurs que les erreurs d'interprétation dans l'exemple donné ci-avant sur l'évaluation empirique auraient pu être évitées par un dialogue en face à face entre les étudiants et un examinateur, i.e. par une évaluation plus rationnelle. À l'inverse, dans l'exemple sur l'évaluation rationnelle, c'est plutôt la mise en place et l'utilisation de critères d'évaluation précis qui réduiraient l'influence des biais de pensée du recruteur, i.e. une évaluation plus empirique. Déterminer un juste milieu entre empirisme et rationalisme semble donc être l'idéal.

Pourtant, au fil du temps, la pratique de l'évaluation a clairement évoluée vers un point de vue empirique dans plusieurs secteurs. Cela s'explique en premier lieu par la difficulté de mettre au point un processus rationnelle dans certains domaines. Par exemple, en médecine, lors de la mise au point d'un nouveau traitement contre une maladie, la seule manière de vérifier l'adéquation de ce traitement aux normes préétablies est de le tester dans le cadre d'une étude expérimentale menée pour recueillir le plus d'observations possibles sur ses effets quand il est administré à des sujets, animaux ou humains. De manière analogue, en apprentissage automatique, la question de l'évaluation d'un système autonome et de la qualité de la tâche qu'il accomplit, est difficilement réalisable de manière rationnelle. La pratique d'une évaluation empirique présente également des avantages indéniables, en commençant par la rapidité d'application des procédures de test, qui peuvent ainsi être réalisées à grande échelle. Mais comme nous l'avons mentionné, les biais et erreurs de jugement affectant ce genre de pratique restent problématiques. Nous préciserons cette notion de biais dans le chapitre 4, et aborderons les méthodes permettant d'y remédier dans le chapitre 5. Pour le moment, nous allons nous concentrer sur la forme que prend l'évaluation en apprentissage automatique, et plus particulièrement en classification.

2.3 L'évaluation en classification

Dans ce qui précède, nous avons mis en évidence la différence et la complémentarité entre deux points de vue pour la notion d'évaluation : l'empirisme et le rationalisme. Nous avons également mentionné que certains domaines sont difficilement en mesure de choisir, étant restreint à une pratique essentiellement empirique de l'évaluation. C'est le cas de l'apprentissage automatique. L'objectif de l'évaluation étant de mesurer la qualité de la description du concept acquise par un système autonome, le plus évident est d'observer son comportement de manière empirique dans un grand nombre de situations différentes, et de synthétiser le tout par une mesure, représentant ainsi la performance du système. S'il était possible d'adopter un autre point de vue, e.g. en extrayant la description du concept du système sous un format facilement interprétable ou analysable par la réflexion humaine, la pratique de l'évaluation serait peut être différente. Malheureusement, plus les modèles d'inférence sont performants, et plus ils sont obscurs [Lam18]. Par exemple, les réseaux de neurones profonds condensent leur description du concept en centaines de millions de paramètres, ce qui rend délicat leur interprétation par le raisonnement.

Nous allons maintenant présenter l'application concrète de l'évaluation empirique pour des problèmes de classification en particulier. Après l'introduction d'un exemple pratique simple, qui nous servira à poser les termes importants, nous parlerons en premier lieu de différentes procédures d'évaluation, e.g. le *holdout* ou la validation croisée, puis nous présenterons certaines

mesures de performance souvent utilisées en pratique, et enfin nous aborderons deux outils statistiques importants en évaluation : les intervalles de confiance et les tests d'hypothèse statistiques. Pour l'ensemble du propos, nous nous inspirons énormément du chapitre 5 du livre de Witten et. al. [Wit+16b].

2.3.1 Le déroulement d'une procédure d'évaluation en apprentissage automatique

Considérons un classifieur que l'on veut entraîner à reconnaître un concept. Pour illustrer notre propos, nous allons prendre le thème de la musique orientale. La première étape est la phase de récolte des instances décrivant le concept, ainsi que leurs annotations. On note l'ensemble des exemples recueillis pour le problème d'apprentissage $\mathcal{E}$ (cf. section 1.4.2). Une fois l'annotation terminée, et la phase d'apprentissage effectuée, le classifieur est évalué pour vérifier si sa description du concept lui permet efficacement de faire la différence entre une musique orientale et une musique d'un autre genre. Pour cela, des instances lui sont présentés, sans leurs annotations, et ses prédictions sont recueillies pour être comparées aux réponses attendues. Cette comparaison prend en pratique la forme d'une mesure de performance, que nous allons dorénavant noter m dans le cas général. De nombreuses mesures différentes existent, et nous en présenterons d'autres par la suite, mais la mesure la plus simple est le taux de réussite du classifieur à prédire si les musiques sont de genre orientales ou non. Cette mesure est appelé la *justesse*. De manière équivalente, on peut plutôt mesurer le *taux d'erreurs*, sachant que la somme des deux mesures est égale à 1. Les instances présentés lors de la phase de test forment ce qu'on appelle l'*ensemble de référence*, ensemble supposé représenter la véritable description du concept.

Un point important est le choix des exemples formant l'ensemble de référence, qui doit respecter deux conditions majeures. L'ensemble de référence doit être [Wit+16b] :

1. disjoint de l'ensemble des exemples utilisés pour l'entraînement ;
2. suffisamment large pour fournir une évaluation précise de la performance du classifieur.

Ces deux conditions forment le principal dilemme de l'évaluation empirique en apprentissage automatique : il est déjà suffisamment difficile d'obtenir des exemples annotés (cf. chapitre 2), mais la phase de test comme la phase d'apprentissage ont chacune besoin d'un ensemble d'exemples aussi grand que possible, ces deux ensembles devant qui plus est être disjoints. Nous allons détailler dans la suite ces deux conditions, en commençant par la première.

Il est bien connu qu'employer dans l'ensemble de référence des exemples ayant contribué à l'apprentissage du classifieur lors de la phase d'entraînement doit être évité [Wit+16b]. En effet, prenons un classifieur qui aurait machinalement appris à associer au titre d'une musique son genre. Il aurait ainsi une justesse maximale sur des exemples qu'il a déjà vu, et pourtant, il resterait incapable de comprendre ce qu'est le genre musical oriental. Le cas échéant, l'évaluation favorise injustement le classifieur : elle est *biaisée*. Pour éviter ce biais, et donc mesurer correctement la qualité de la description du concept du classifieur, il est plus informatif de questionner son aptitude à *généraliser* à de nouveaux exemples, n'ayant pas servi pendant la phase d'entraînement. Par conséquent, l'ensemble des exemples du problème d'apprentissage $\mathcal{E}$ est divisé en deux :

— un ensemble dit d'entraînement ou d'apprentissage $\mathcal{E}_A$;
— un ensemble dit d'évaluation, de test ou de référence $\mathcal{E}_T$; on l'appelle également vérité-terrain.

Les mesures de performance calculées sur l'ensemble $\mathcal{E}_A$ sont dites de *re-substitution*, bien que le terme concerne principalement la mesure du taux d'erreur. L'application de mesures de

re-substitution, c'est-à-dire lorsque $\mathcal{E}_T \subset \mathcal{E}_A$, mène à une évaluation qui surestime le classifieur, à l'image de celui associant machinalement les titres de musiques à leur genre. Ces mesures ne doivent donc pas être utilisées, du moins pas toutes seules, pour évaluer sa performance. C'est pourquoi un ensemble de test $\mathcal{E}_T$ différent est nécessaire. Néanmoins, leur utilisation est utile pour déterminer quand un classifieur a, ou commence, à sur-apprendre les exemples d'apprentissage. En effet, le phénomène de sur-apprentissage est visible par le fait que la mesure de l'erreur de re-substitution d'un classifieur est faible tandis que l'erreur sur un ensemble de test disjoint de l'ensemble d'apprentissage est haute. Plus l'écart entre les deux mesures croît, plus le sur-apprentissage est important.

L'ensemble de référence doit aussi être suffisamment large. En effet, imaginons que l'on choisisse une seule musique A comme unique exemple de test. Les seules mesures de justesse envisageables avec un tel ensemble de référence sont soit 0% soit 100%. Un classifieur sachant prédire correctement si cette musique est orientale ou non obtiendrais donc une justesse maximale, indépendamment de sa capacité à classifier correctement d'autres musiques. En pratique, bien sûr, l'ensemble de référence contient de nombreux exemples de test, mais plus sa taille est réduite, plus la mesure effectuée est imprécise : on parle de variance de l'estimation de la mesure. Lors de l'évaluation, on cherche à réduire au maximum cette variance, ce qui est possible en augmentant la taille de l'ensemble de référence.

Enfin, il est nécessaire de se demander si les exemples du problème d'apprentissage, particulièrement ceux de l'ensemble de test, représentent adéquatement le concept à apprendre. Il serait certainement problématique que lors de l'évaluation, on n'utilisait que des musiques possédant des caractéristiques orientales évidentes, ou alors provenant d'une seule des multiples régions d'orient connues pour leur art musical. Dans ce cas, un classifieur pourrait n'avoir de bons résultats que sur ces musiques, et aurait donc une justesse relativement haute et non représentative de sa performance en situation réelle. Pour être en mesure de fournir une évaluation fiable, il faut en premier lieu essayer de s'assurer au mieux de la variété des exemples collectés, de sorte à ce qu'ils représentent le concept à apprendre dans son entièreté. Si cela semble facile à dire, c'est bien plus compliqué à réaliser en pratique.

Nous nous référons à cette dernière caractéristique de l'ensemble des exemples par le terme de « représentativité ». La notion de représentativité concerne initialement le champ des statistiques, dans lequel elle caractérise un échantillon de données. La variété de définitions qui lui sont attribuées témoignent néanmoins de sa complexité. Ramsey et. al. tentent de proposer une définition commune, scientifiquement cohérente, de la représentativité d'un échantillon statistique [RH05]. Leur propos, orienté pour des études environnementales, fait remarquer qu'une bonne définition du terme est dépendante de la tâche associée, i.e. la question pour laquelle l'échantillon doit servir à fournir une réponse. Les auteurs listent ainsi plusieurs étapes de questionnement sur différents aspects :

- *Quelle est la tâche ?*
- *Quelle est la population cible ?*
- *Quelle niveau de confiance veut-on obtenir ?*
- *Le plan d'échantillonnage respecte-t-il les principes d'hétérogénéité et de choix uniforme des données ?*
- *Un contrôle de qualité, e.g. détermination des erreurs et des biais dans les données récoltées, a-t-il été effectué ?*

Si la collecte des données ne respecte pas l'un de ces points, ces dernières ne peuvent être qualifiées de représentatives, et ne peuvent donc mener à des conclusions statistiques valides. La méthodologie proposée n'aborde malheureusement aucun aspect quantitatif, en particulier pour estimer si l'échantillon est suffisamment hétérogène. La notion de représentativité d'un ensemble

d'exemples est donc difficilement mesurable en pratique, alors qu'elle est essentielle à la qualité et la validité des résultats obtenus, aussi bien pour l'apprentissage d'un classifieur que pour son évaluation.

En pratique, l'ensemble des exemples d'un problème d'apprentissage est donc implicitement supposé suffisamment représentatif du concept. Après tout, dans le cas contraire, il n'y aurait aucun sens à mener à terme la construction d'un modèle. Cette supposition n'est néanmoins pas anodine, et conduit parfois à des classifieurs au comportement inattendu (cf. chapitre 4). En résumé, la fiabilité de l'évaluation dépend de trois facteurs interdépendants :

— **condition (i)** : elle doit porter sur des exemples nouveaux, non présentés au classifieur lors de son apprentissage ;
— **condition (ii)** : l'ensemble de référence doit être suffisamment large, pour une bonne précision de la mesure de performance ;
— **condition (iii)** : le taux de représentativité de cet ensemble pour le concept d'intérêt doit être suffisamment élevé.

2.3.2 Les procédures d'évaluation usuelles

En plus du fait de déployer le maximum d'effort dans la collecte et l'annotation des instances du problème d'apprentissage, des procédures d'évaluation spécifiques ont été développées pour aider au respect de ces trois aspects.

Une procédure fréquemment utilisée répond simplement à la **condition (i)**, et consiste en la séparation des exemples d'entraînement et de test en deux ensembles disjoints, comme nous l'avons exprimé plus tôt. Cette procédure se nomme le *holdout*, ce qui veut littéralement dire *tenu à l'écart*, au sujet de l'ensemble de référence. En général, il est conseillé d'avoir l'ensemble d'entraînement plus grand que l'ensemble de test, même si cela affecte la qualité de l'évaluation. Des taux de séparation à 80%/20%, ou 75%/25%, sont en pratique fréquents.

Une procédure plus coûteuse que le *holdout*, mais permettant une estimation plus précise de la performance d'un classifieur, est la procédure de *validation croisée à n plis*. Le principe est de scinder l'ensemble $\mathcal{E}$ en n partie de tailles égales, puis d'utiliser $n-1$ plis dans la phase d'entraînement, suivi d'une phase de test sur le dernier pli, et de répéter le processus en changeant le pli de test. La mesure d'évaluation, e.g. la justesse, est alors moyennée sur toutes les phases de test. Le coût de cette procédure vient du nombre n de modèles à entraîner, mais son avantage est que contrairement au *holdout*, tous les exemples sont utilisés aussi bien pour l'entraînement que pour l'évaluation. Cela permet donc de mieux satisfaire la **condition (ii)**.

La validation croisée est en général préférable au *holdout* pour des ensembles de données n'étant pas trop large [YS16]. Plusieurs études empiriques montrent que choisir n aussi grand que possible donne de meilleurs résultats, et que $n = 10$ semblent être le choix commun pour avoir un équilibre entre ressources de calcul et précision des estimations [YS16 ; WY19 ; Kim09]. Il est également possible de pousser à l'extrême la procédure, et de ne garder qu'un seul exemple dans l'ensemble de test à chaque fois. Cette procédure prend l'appellation particulière de *leave-one-out* : une validation croisée à n plis où n est le nombre d'exemples dans $\mathcal{E}$. Elle possède l'avantage que les classifieurs sont entraînés sur le plus grand nombre d'exemples possible. Cependant, étant très coûteuse, elle est conseillée seulement lorsque l'ensemble d'exemples est suffisamment petit, et que le modèle utilisé n'implique pas un trop long temps d'entraînement.

Lors d'une procédure de *holdout* ou de validation croisée, il se peut que la division particulière effectuée entre exemples d'entraînement et exemples de test soit malchanceuse. Imaginons en effet que la quasi-totalité des exemples d'une certaine classe aient été omise de l'ensemble d'entraînement. Non seulement la qualité de la phase d'apprentissage est réduite, mais cela im-

plique également un biais lors de la phase d'évaluation, car le classifieur construit est évalué sur des exemples d'une classe surreprésentée par rapport à ce qui lui a été montré lors de son apprentissage. Éviter cette situation va donc dans le sens de la **condition (iii)**. Pour cela, à chaque fois que l'ensemble d'exemples est divisé, il faut s'assurer que chaque classe du problème est correctement représentée dans chaque sous-ensemble. On dit que les procédures de *holdout* ou de validation croisée sont *stratifiées*. De manière plus générale, pour réduire l'impact causé par le choix d'une séparation particulière en sous-ensemble d'entraînement et de test, la procédure peut être effectuée plusieurs fois en divisant différemment l'ensemble d'exemples, et en moyennant les différentes performances obtenues. On parle alors de *holdout* (ou validation croisée) *répété*. Bien sûr, rien n'empêche de combiner les deux techniques, i.e. d'appliquer par exemple une validation croisée stratifiée et répétée plusieurs fois.

En plus de ces procédures d'estimation, i.e. le *holdout*, la validation croisée et le cas particulier du *leave-one-out*, il en existe une autre bien connue, appelée le *bootstrap* [Koh+95 ; Efr82]. Le principe est encore une fois de diviser l'ensemble d'exemples en un ensemble pour l'entraînement et un autre pour l'évaluation, mais cette fois un exemple peut être choisi plusieurs fois pour faire partie de l'ensemble d'entraînement [5]. Lors d'une procédure de *bootstrap*, l'ensemble d'entraînement est donc construit en choisissant N exemples dans l'ensemble initial, lui-même de taille N, où le tirage des exemples est effectué avec remise. De cette manière, il est quasi-certain que des exemples sont choisis plus d'une fois. Les exemples qui n'ont pas été choisis forment l'ensemble de test. La probabilité qu'un exemple ne soit jamais choisi vaut $(1-\frac{1}{n})^n$, ce qui vaut $e^{-1} = 0.368$. Si la taille de l'ensemble N est assez grand, l'ensemble d'entraînement obtenu contiendrait donc à peu près 63.2% des exemples, et l'ensemble de test serait formé des 36.8% restant.

Dans le cadre d'une procédure de *bootstrap*, l'estimation de la performance du classifieur obtenu après entraînement est considérée pessimiste, relativement à une estimation qui aurait été obtenue e.g. par validation croisée, car dans un cas l'entraînement n'est fait qu'avec 63% des exemples, alors que dans l'autre, en général, 90% des exemples sont utilisés. Pour compenser, le *bootstrap* combine la mesure effectuée sur l'ensemble de test avec la mesure de re-substitution, obtenue sur l'ensemble d'entraînement. En général, la mesure utilisée est le taux d'erreur *err*. L'erreur obtenue sur l'ensemble de test est donc combinée avec l'erreur de re-substitution, en pondérant par la proportion du nombre d'exemples contenus dans chaque ensemble :

$$err_{bootstrap} = 0.632 \cdot err_{test} + 0.368 \cdot err_{resub} \tag{2.1}$$

La procédure est ensuite répétée entièrement plusieurs fois, de sorte à moyenner les résultats pour différents tirages des exemples d'entraînement. Le *bootstrap* présente des avantages similaires au *leave-one-out* lorsque le nombre d'exemples disponibles est faible, et est en plus préférable en terme de temps de calcul. Bien sûr, il y a également des désavantages à la pratique de ces procédures. Nous référons le lecteur au chapitre 5 du livre de Witten et. al. pour une présentation de situations artificielles illustrant leurs inconvénients [Wit+16b].

Nous avons présenté dans cette partie les procédures d'évaluation usuelles, ainsi que leurs intérêts et inconvénients. La figure 2.1 en reprend les grandes lignes. Mais en ce qui concerne la mesure estimée par ces procédures, nous avons uniquement parlé de la justesse ou du taux d'erreur. La prochaine partie s'attachera donc à présenter d'autres mesures de performance utiles en classification.

5. Pour l'apprentissage, cela n'est pas sans conséquence : entraîner un modèle sur un ensemble d'exemples, et l'entraîner sur le même ensemble mais contenant maintenant des exemples qui ont été répétés, ne produit pas forcément le même résultat.

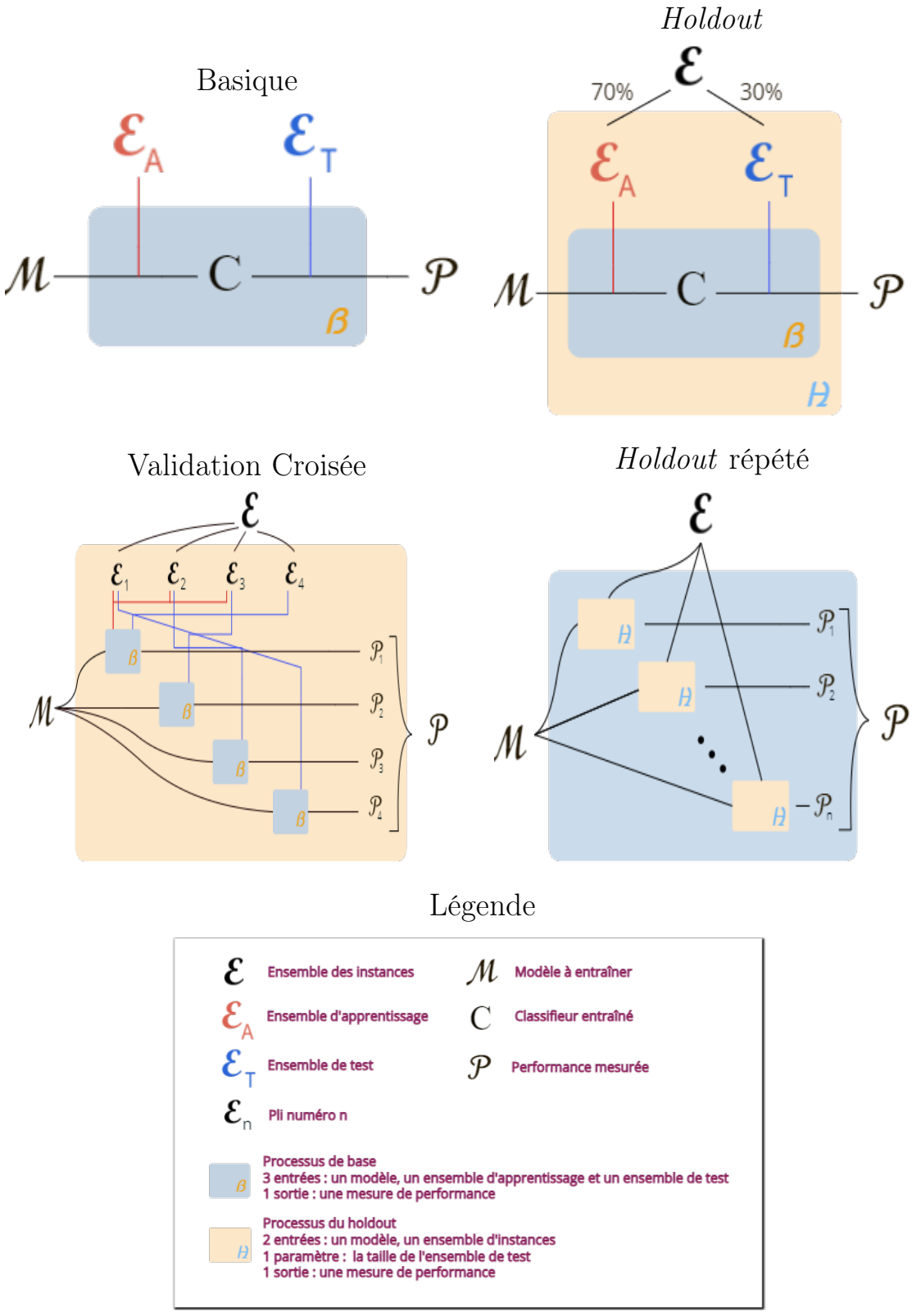


FIGURE 2.1 – Schéma résumant le déroulement des procédures d'évaluations principales. Pour la validation croisée et le *holdout* répété, la performance finale est une moyenne des performances mesurées à chaque étape.

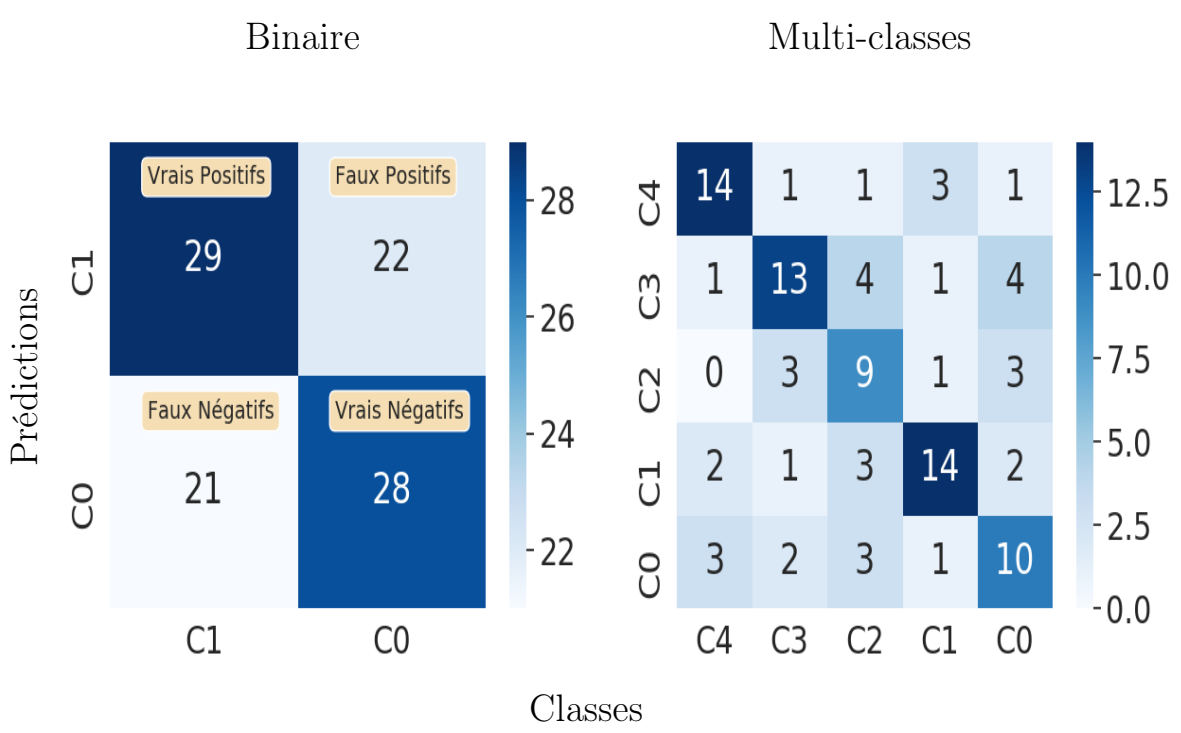


FIGURE 2.2 – Exemples de matrices de confusion pour des cas de classification binaire (à gauche) et multi-classes (à droite).

2.3.3 Les mesures de performance et la matrice de confusion

En ce qui concerne les mesures d'évaluation, mesurer la justesse ou le taux d'erreurs est très habituel, mais pas forcément adapté à toutes les situations. Ces mesures, et plus généralement les indicateurs de performances utilisés pour l'évaluation, peuvent facilement induire en erreur lorsqu'ils sont mal interprétés. Nous avons déjà donné à ce sujet l'exemple des mesures de re-subsitution, pouvant mener à la surestimation d'un classifieur. Voici un exemple différent. Supposons que l'on construise un classifieur pour faire de la détection d'anomalie, disons dans un système informatique complexe où il arrive parfois que certains modules soient sujet à des dysfonctionnements. Les exemples positifs, i.e. les traces d'exécutions qui présentent une anomalie, sont naturellement peu nombreux comparés aux exemples négatifs ne montrant aucune anomalie. Dans ce cas, la justesse du classifieur ne représente pas convenablement sa performance : un classifieur qui ne détecterait aucune anomalie aurait une justesse relativement élevée (selon le nombre d'exemples négatifs dans l'ensemble de test) alors que sa description du concept est totalement inutile. Dans ce genre de situation, d'autres mesures sont préférables. En particulier, la *précision* et le *rappel* sont les plus populaires en classification. Nous allons dans la suite nous concentrer principalement sur le calcul de ces mesures usuelles, i.e. la justesse, le taux d'erreur, ainsi que la précision, le rappel et la f-mesure, et nous indiquerons en fin de partie des références supplémentaires concernant la grande variété de mesures différentes existant dans le domaine.

Avant de donner leurs définitions, nous allons tout d'abord introduire le concept de matrice de confusion. Fréquemment utilisée lors de la phase d'évaluation, cette matrice fournit une représentation compacte de la comparaison entre prédictions d'un classifieur et annotations dans l'ensemble de référence. Pour un problème de classification binaire, par exemple la détection d'anomalie dont il était question précédemment, parmi les deux classes du problème, l'une est

nommée *positive* et codée par 1, l'autre *négative* et codée par 0. La matrice de confusion $\mathcal{M}$ est une matrice 2x2, telle que la cellule (i, j) contienne le nombre d'exemples annotés dans la catégorie j et prédits dans la catégorie i (figure 2.2). Les exemples comptés dans la cellule m_{11} sont dits *vrais positifs*, tandis que ceux dans m_{00} sont nommés *vrais négatifs*. De même, les exemples dans m_{01} et m_{10} sont respectivement appelés *faux négatifs* et *faux positifs*.

Cette matrice se généralise pour de la classification multi-classes. La plupart des mesures utilisées en classification peuvent être calculées à partir de cette matrice. Par exemple, pour la précision p_r et le rappel r_p dans le cas de la classification binaire :

$$p_r = \frac{TP}{TP + FP} \tag{2.2}$$

$$r_p = \frac{TP}{TP + FN} \tag{2.3}$$

Ces deux mesures sont souvent complémentaires, car un classifieur peut avoir une mauvaise description du concept, et malgré tout obtenir un score parfait pour l'une des deux. Un bon classifieur présente en général des scores acceptables pour les deux mesures. C'est pourquoi on utilise souvent une mesure combinant la précision et le rappel : la F-mesure. Son expression est le ratio entre les moyennes harmonique et arithmétique. Lorsque la précision ou le rappel sont trop bas, la F-mesure l'est également :

$$f_m = \frac{2 p_r r_p}{p_r + r_p} \tag{2.4}$$

Dans la suite, nous parlons plus généralement d'*indicateurs de performance* pour désigner ce qui permet d'évaluer la performance d'un classifieur. Un indicateur de performance peut prendre la forme d'une mesure numérique simple. C'est le cas de la justesse et autres mesures que nous venons de présenter, et il en existe bien d'autres [SJS06 ; SL09 ; LC12 ; GG05]. Les indicateurs de performance peuvent aussi être plus complexe, ou de formes différentes :

- combinaison pondérée de plusieurs mesures numériques simples, auquel cas leur interprétation est plus difficile (e.g. précision moyennée sur plusieurs classes, dans le cas multi-classe, ou fusion ad hoc de plusieurs mesures [Pen+11]) ;
- graphes en deux dimensions [HM82 ; SR15] ;
- résultat d'un test statistique permettant de faire la comparaison entre deux modèles d'inférence donnés (cf. section 2.3.5).

Nous devons maintenant aborder une notion essentielle à la question posée dans le cadre de notre thèse : la notion de confiance. Mesurer empiriquement la performance d'un classifieur par un indicateur de performance doit en effet s'accompagner par une quantification du niveau de confiance que l'on peut avoir sur le résultat. En apprentissage automatique, cette confiance est mesurée dans un cadre statistique par un *intervalle de confiance*.

2.3.4 Les intervalles de confiance

Lorsque l'évaluation prend la forme d'une simple mesure, une question importante se pose : quel est le niveau de confiance que l'on peut accorder à la valeur mesurée ? Imaginons qu'un professeur évalue son élève à l'aide d'un questionnaire à choix multiples. Si l'élève obtient 80 réponses justes sur un total de 100, ou bien 8 réponses justes sur un total de 10, sa justesse vaut 80% dans les deux cas. Cependant, le cas de figure mettant le plus en confiance le professeur est naturellement celui où le questionnaire contient 100 questions. L'évaluation lui semble en effet moins incertaine. Cette incertitude provient du fait que la détermination de la mesure est une

estimation empirique approximative de sa valeur réelle, obtenue sur un nombre réduit d'exemples. En effet, dans l'idéal, si l'on voulait connaître la performance d'un classifieur avec une précision parfaite, il faudrait pouvoir l'évaluer sur un ensemble infini d'exemples, et non utilisés pour son apprentissage.

L'incertitude entourant cette estimation est en général mesurée par un intervalle de confiance, étant donné un risque $\alpha \in [0, 1]$. Cela signifie en pratique qu'il y a une probabilité $1 - \alpha$ que l'on obtienne un intervalle contenant la véritable valeur de la performance du classifieur. Pour un risque de 0.05, on obtiendrait par exemple un intervalle de $[0.49, 0.94]$ pour une estimation de 80% de justesse mesurée sur seulement 10 exemples, et $[0.71, 0.87]$ lorsqu'elle est obtenue sur 100 exemples [6]. Dans ce qui suit, nous détaillons la manière dont ces intervalles sont calculés. Cela nous servira par la suite lorsque nous parlerons de la confiance que l'on peut avoir en une mesure de performance estimée à partir d'un ensemble de test contenant des erreurs d'annotation. Les notations $\mathbb{E}$ et $\mathbb{V}$ se réfèrent respectivement aux opérateurs d'espérance et de variance.

Voyons comment déterminer un intervalle de confiance à partir d'une estimation donnée. Cette estimation est obtenue par l'*estimateur* de la mesure m, que l'on notera $\hat{m}$. Un estimateur est une variable aléatoire utilisée pour estimer une quantité précise. Par exemple, pour estimer la quantité m, nous pouvons utiliser une variable aléatoire dont les réalisations sont proches de m. Ces réalisations sont appelées « estimations ». La qualité de l'estimateur dépend de son espérance et de sa variance : dans l'idéal, l'espérance doit valoir m, pour qu'il n'y ait aucun biais dans les estimations (l'estimateur est dit non biaisé), et la variance doit être aussi faible que possible, pour que chaque estimation faite soit suffisamment proche de l'espérance, et donc de m. Nous noterons généralement e une estimation obtenue par un estimateur, indépendamment de la quantité estimée et donc de l'estimateur utilisé. L'estimation, l'estimateur et la mesure sont donc trois objets différents.

En ce qui concerne la mesure de la justesse q, l'estimateur usuel prend la forme d'une somme moyennée de variables de Bernoulli $\mathcal{B}_i$, toutes indépendantes et identiquement distribuées :

$$\hat{q} = \frac{1}{N} \sum_{i}^{N} \mathcal{B}_i \tag{2.5}$$

Chaque variable $\mathcal{B}_i$ vaut 1 lorsque le classifieur prédit la bonne catégorie pour l'exemple i de l'ensemble de test, 0 sinon. Si le nombre d'exemples N est suffisamment grand, on peut considérer que $\hat{q}$ suit une loi normale, d'espérance et de variance :

$$\mathbb{E}(\hat{q}) = \mathbb{E}(\mathcal{B}_i) \tag{2.6}$$

$$\mathbb{V}(\hat{q}) = \frac{\mathbb{E}(\mathcal{B}_i)(1 - \mathbb{E}(\mathcal{B}_i))}{N} \tag{2.7}$$

On note e une réalisation de $\hat{q}$, ce que l'on nomme une estimation. On désire trouver a et b tels que la probabilité que e soit contenue dans l'intervalle $[a, b]$ vaille $1 - \alpha$. Pour cela, on peut utiliser la relation suivante, valable pour une variable aléatoire X d'espérance nulle et de variance unitaire :

$$\mathbb{P}(-z < X < z) = c \tag{2.8}$$

Pour chaque valeur de z, la valeur de c correspondante se trouve facilement dans des tables statistiques couramment utilisées. Comme on suppose l'estimateur $\hat{q}$ normalement distribué, il

6. Ces intervalles de confiance ont été calculés à partir de l'expression 6.30

suffit de le centrer et réduire pour pouvoir l'utiliser dans cette relation :

$$\mathbb{P}(-z < \frac{\hat{q} - \mathbb{E}(\hat{q})}{\sqrt{\mathbb{V}(\hat{q})}} < z) = \mathbb{P}(-z < \frac{e - q}{\sqrt{\frac{q(1-q)}{N}}} < z) = 1 - \alpha \tag{2.9}$$

Obtenir a et b revient alors à inverser l'inégalité dans la relation précédente, et résoudre l'équation du second degré en q qui en résulte. Les bornes de l'intervalle sont les solutions de l'équation, et sont données par :

$$\frac{e + \frac{z^2}{2N} \pm z\sqrt{\frac{e}{N} - \frac{e^2}{N} + \frac{z^2}{4N^2}}}{1 + \frac{z^2}{N}} \tag{2.10}$$

Bien que le calcul ait été effectué pour l'estimateur de la justesse $\hat{q}$, il reste valable pour tout autre estimateur pouvant s'exprimer comme la somme moyennée de variable de bernoulli. Par exemple, cela fonctionne également pour les estimateurs de la précision et du rappel, en modifiant le nombre N d'exemples considérés, comme nous le verrons dans un prochain chapitre.

Il arrive de se méprendre, en ce qui concerne la signification de ces intervalles. Obtenir un intervalle de confiance $[a, b]$ à un risque α étant donnée une estimation e pour une quantité q que l'on cherche à estimer ne veut pas dire qu'il y a une probabilité $1 - \alpha$ pour que q se trouve dans l'intervalle. En effet, cet énoncé n'a pas de sens mathématique car q n'est pas une variable aléatoire, mais seulement un nombre réel fixe, bien qu'inconnu. Ce que l'intervalle signifie, c'est que *si* la quantité q ne s'y trouve pas, la probabilité *a priori* d'observer la valeur e estimée $\mathbb{P}_q(\hat{q} = e)$ est inférieur au risque considéré [7].

Nous terminons ce chapitre par expliciter la notion de test d'hypothèse statistique. Celle-ci est en effet étroitement liée à la notion d'intervalle de confiance, lien que nous aborderons dans le chapitre 6. C'est pourquoi nous présentons dès maintenant le principe d'un tel test.

2.3.5 Les tests d'hypothèse statistiques

Lors d'une procédure d'évaluation en apprentissage automatique, il est fréquent de vouloir comparer deux modèles d'inférence en particulier. Pour cela, on pourrait simplement estimer les mesures de performance de ces deux modèles par *holdout* ou validation croisée, éventuellement répéter plusieurs fois la procédure, et comparer les résultats. Il est vrai que cela fonctionne dans de nombreux cas. Cependant il est également possible que la différence observée entre les deux modèles ne soit due qu'à des fluctuations du processus d'estimation, ce qui ne devrait pas nous permettre de nous décider sur la supériorité d'un des deux modèles. Il existe un outil adapté en statistique permettant de comparer deux éléments aux comportements modélisés par un processus aléatoire : le test d'hypothèse nommé *t-test*. Ce genre de test se fonde sur les notions de confiance et de risque que l'on a vu en ce qui concerne les intervalles de confiance, et c'est la raison pour laquelle il est intéressant d'en présenter le principe, par rapport à notre problématique, où nous nous intéressons à la confiance en le résultat d'une évaluation menée avec des exemples de test de mauvaise qualité.

Le principe des tests d'hypothèse statistiques est le suivant. Étant donnés deux modèles d'inférence A et B et un ensemble d'exemples $\mathcal{E}$, nous nous posons la question de déterminer si l'un des modèles est véritablement meilleur que l'autre. Pour cela, nous allons obtenir k différentes estimations de la mesure m pour A, que l'on note $(m_1^A, \ldots, m_k^A)$, et faire de même pour B. Les

7. La quantité q se trouve en indice car c'est un paramètre de la distribution de probabilité de l'estimateur $\hat{q}$.

pairs (m_i^A, m_i^B) sont obtenues en appliquant une validation croisée à k plis, ce qui donne effectivement k estimations de la mesure m, chacune sur des ensembles différents. De cette manière, nous disposons de plusieurs pairs (m_i^A, m_i^B) indépendantes les unes des autres. C'est pourquoi nous parlons de t-test par pairs. En considérant un nombre d'échantillons k suffisamment grand, la moyenne $\overline{m^A}$, et respectivement celle pour B, suivent des lois normales d'espérances réelles respectives μ^A et μ^B. En connaissant les variances de ces lois, il est alors possible d'estimer des intervalles de confiance pour ces deux espérances. Les variances sont malheureusement inconnues, mais on peut tout de même les estimer à partir des échantillons à disposition.

La variance de $\overline{m^A}$ peut donc être estimée en considérant la variance des échantillons $(m_1^A, \ldots, m_k^A)$, notons la ν^A, et en la divisant par k. On peut alors centrer et réduire $\overline{m^A}$:

$$\frac{\overline{m^A} - \mu^A}{\sqrt{\frac{\nu^A}{k}}}. \tag{2.11}$$

Cette variable ne suit cependant plus une loi normale, car nous avons dû estimer la variance pour la réduire [Wit+16b]. Elle suit ce qu'on appelle une loi de Student à $k-1$ degrés de liberté. Pour obtenir les intervalles de confiance, nous devons donc prendre les valeurs de risque et leur z associés dans la table spécifique à cette distribution et à ce nombre de degrés de liberté. Cela donne en pratique des intervalles de confiance plus large, représentant l'incertitude liée à n'avoir pu obtenir qu'une approximation des variances précédentes.

L'intervalle de confiance que nous voulons calculer est en fait celui pour la différence $\overline{d} = \overline{m^A} - \overline{m^B}$. Cela ne change en rien le problème, les estimations étant effectuées par pairs, il n'y a qu'à considérer que les échantillons correspondent aux différence des éléments de chaque pair. La variable aléatoire $\overline{d}$ peut être centrée et réduite de la même manière, et elle suit la même loi de Student.

Si les moyennes $\overline{m^A}$ et $\overline{m^B}$ sont en réalité les même, leur différence est nulle : c'est ce qu'on appelle *l'hypothèse nulle*. L'objectif du test est de supposer cette hypothèse, i.e. l'espérance de $\overline{d}$ est nulle, d'obtenir pour la loi de Student adéquate les valeurs de z et $-z$ associées au risque α considéré, et de voir si la réalisation de $\overline{d}$ dont on dispose est plus grande que z ou plus petite que $-z$. Le cas échéant, nous pouvons conclure que l'espérance de $\overline{d}$ est significativement différente de 0 et que A est meilleur que B, ou inversement selon le signe obtenue, au risque α de se tromper et d'invalider à tort l'hypothèse nulle.

Ce test possède des limites, dues en particulier au fait que, l'ensemble d'exemples étant fini, il est difficile d'obtenir des échantillons $(m_1^A, \ldots, m_k^A)$ de bonne qualité et surtout réellement indépendants, étant donné que ces mesures proviennent de modèles ayant été entraînés sur des ensembles techniquement différents mais contenant une grande proportion d'exemples similaires. Plusieurs modifications de ce test ont été proposées pour tenir compte de ces problèmes [Wit+16b], mais il n'est pas nécessaire de les aborder dans le cadre de cette thèse.

2.4 Conclusion

Nous avons défini un certain nombre de notations, et présenter les fondamentaux de l'évaluation en classification :

- les procédures de test usuelles : *holdout*, validation croisée, *bootstrap*, ainsi que les principes de stratification et de répétition ;
- les mesures de performance et leur utilité selon la situation : justesse, taux d'erreurs, précision, rappel, f-mesure, ainsi que la définition d'une mesure de re-substitution ;

— l'outil principal pour les calculer : la matrice de confusion ;
— comment la notion de confiance peut être intégrée à la procédure d'évaluation : l'utilisation d'intervalles de confiance ou de tests d'hypothèse statistiques.

Nous avons également parlé, de manière plus générale, du principe de l'évaluation empirique, et de son opposition à un point de vue plus rationnel. Dans le prochain chapitre, nous aborderons le dernier élément d'intérêt de notre problématique : le bruit d'annotation présent dans les ensembles de données en apprentissage automatique.

Chapitre 3

Le bruit d'annotation

Sommaire

La collecte des instances d'un problème d'apprentissage est une tâche essentielle en apprentissage supervisé, en particulier en ce qui concerne leur annotation. Nous allons par conséquent aborder un problème majeur à ce sujet : le bruit d'annotation. Après avoir donné une définition précise de ce terme, nous présentons les conséquences négatives qu'il implique, en nous concentrant pour l'instant sur la phase d'entraînement des classifieurs. Nous parlons enfin des techniques proposées dans la littérature pour pallier ce problème.

3.1 La tâche d'annotation

La tâche d'annotation est une étape importante dans tout problème d'apprentissage. Cependant, l'obtention d'instances annotées n'est pas simple. La première difficulté se situe dès lors que l'on essaie de définir précisément le concept à apprendre, et de récolter les instances de sorte à être en mesure de le décrire au mieux. Imaginons par exemple que nous voulons apprendre à un classifieur les concepts de chiens et de loup. Sans bien réfléchir aux concepts, et à comment les images que l'on récolte les représentent, on peut se retrouver avec un classifieur qui répond « loup » lorsque l'image montre un paysage enneigé, et « chien » sinon.

La tâche d'annotation doit également être spécifiée. Il faut donc penser à une façon cohérente de séparer les instances récoltées en différentes catégories, en tenant compte des caractéristiques qui les définissent, et plus particulièrement sur quels axes ces caractéristiques permettent de

les différencier. Un ensemble de consignes précises doit être fourni aux annotateurs, pour éviter qu'ils ne se retrouvent dans des situations ambiguës, où il est difficile d'interpréter une instance spécifique, bien qu'il soit impossible d'éviter complètement ce genre de situations.

Enfin, la tâche d'annotation en elle-même est un processus long et fastidieux pour les annotateurs. Les difficultés de l'annotation manuelle de données touchent tous les domaines, bien que pour certains, la tâche soit encore plus complexe, e.g. le traitement automatique du langage [FNR12]. Dans ce qui suit, nous allons énoncer les difficultés survenant durant la tâche d'annotation dans un cadre général, de sorte à introduire ensuite le problème qui nous intéresse : le bruit d'annotation.

En ce qui concerne les annotateurs, l'idéal est en général de choisir un ou plusieurs experts, i.e. ayant d'une part l'habitude du processus d'annotation, et connaissant le mieux le concept dont on veut apprendre la description, surtout si celui-ci n'est pas trivial. Cela n'est cependant pas toujours faisable par manque de moyens, et en pratique beaucoup d'ensemble de données sont annotés par de la main d'œuvre peu cher, fournissant un travail de moindre qualité. Selon le niveau d'expertise des annotateurs, ainsi que la particularité du problème d'apprentissage, les difficultés rencontrées lors de la tâche d'annotation sont multiples, et d'ampleur variables :

— (i) la fatigue qui s'accumule après avoir annoté des dizaines, voire des centaines d'instances ;
— (ii) l'existence d'instances *marginales* difficiles à annoter correctement ;
— (iii) l'ambiguïté d'interprétation, pouvant soit provenir d'une subjectivité inhérente au concept, i.e. certaines instances peuvent légitimement être interprétées de plusieurs façons, soit être simplement causée par une mauvaise spécification du problème d'apprentissage et de la tâche d'annotation ;
— (iv) plus simplement, le manque d'expertise des annotateurs pour le concept d'intérêt.

En pratique, ces points sont souvent corrélés les uns aux autres. En guise d'exemple, considérons les deux situations suivantes :

— imaginons que nous voulons prédire la tranche d'âge d'une personne, i.e. 0-10 ans, 10-20 ans et ainsi de suite, uniquement à partir de ses caractéristiques physiques, principalement son visage et sa taille. Posé de cette manière, le problème contient naturellement des instances marginales : des instances appartenant en réalité à une classe i, e.g. des personnes ayant la trentaine, mais dont les caractéristiques sont très semblables aux instances d'une autre classe j, e.g. de la tranche d'âge 50-60 ans.
— Supposons maintenant que nous voulons plutôt prédire si une phrase concerne un sentiment positif ou négatif. Il se peut alors qu'il y ait des phrases pouvant aussi bien être interprétées positivement que négativement, e.g. « j'ai bien aimé ... mais je déplore le manque de ... », ou alors que certaines phrases puissent être à caractère ironique selon le point de vue du lecteur. L'annotation devient alors subjective à l'interprétation de l'annotateur.

Ces problèmes peuvent même affecter les annotateurs experts. Ainsi, deux experts peuvent avoir des interprétations différentes pour la même instance. Par conséquent, pour obtenir une annotation de meilleure qualité, il est en général nécessaire de demander l'avis de plusieurs experts pour chaque instance, de sorte à pouvoir calculer une mesure de *l'accord inter-annotateurs*, i.e. un pourcentage représentant le nombre de fois où les annotateurs ont eu la même interprétation. La difficulté d'obtention de cette statistique varie selon le domaine [NLF99 ; Vér98 ; Bra00 ; NR10]. Si cet indicateur a des points faibles [Art17], il permet tout de même de donner une idée quantitative de la subjectivité de la tâche d'annotation d'un problème d'apprentissage donné, telle qu'elle a été spécifiée, et par conséquent d'améliorer cette spécification de manière itérative [Ter+18]. Par exemple, obtenir un accord inter-annotateur de 60% indique que pour 40% des

instances, l'annotation (telle qu'elle a été spécifiée) est trop subjective car les annotateurs ont des avis divergents, ce qui permet de revoir sa spécification, de sorte à rendre moins ambiguës les instances ayant précédemment posé problème.

La complexité de la tâche d'annotation a une conséquence majeure : la fréquence d'erreurs d'annotation. Mais que peut-on désigner par « erreurs », lorsque la subjectivité de la tâche est une réalité, et que les différences d'interprétation sont nombreuses même entre experts ? Pour les points (i), (ii) et (iv), on peut admettre qu'un annotateur commette une réelle erreur sur un exemple à interprétation unique, soit due à la fatigue, soit parce que l'exemple est véritablement difficile car il présente des caractéristiques marginales et/ou nécessite une connaissance élargie du domaine. Mais en ce qui concerne le point (iii), il est sans doute moins évident de parler d'erreurs. Dans la partie suivante, nous nous intéressons à cette problématique, qu'on appelle communément le *bruit d'annotation*.

3.2 Le bruit d'annotation

Dans le contexte de classification dans lequel nous nous plaçons, le bruit d'annotation se réfère à des instances munies d'une annotation qui diffère de leur classe réelle. Nous noterons $\mathbf{x}$ l'instance, $\mathbf{y}$ son annotation, et $\mathbf{c}$ sa classe réelle. Nous avons vu dans la section précédente pour quelles raisons cela peut arriver : la fatigue, les instance marginales, l'ambiguïté du concept ou de la spécification de la tâche d'annotation, ou alors le manque d'expertise des annotateurs. En particulier, la subjectivité des annotateurs, source importante de bruit d'annotation, est un problème connue dans certains domaines, comme la recherche médical [MBN06 ; HRT04], ou l'analyse d'images [Smy+95 ; Smy96]. Dans d'autres domaines, le bruit d'annotation et son impact sont peu souvent abordés. Nous verrons dans la section 3.3 que ce phénomène implique en général des conséquences négatives. Avant cela, nous commençons par définir le bruit d'annotation, et présentons les manières de le modéliser.

3.2.1 Contexte

Pour parler de bruit d'annotation dans un problème d'apprentissage, il faut supposer que, pour toute instance $\mathbf{x}$, il existe une catégorie $\mathbf{c}$, la véritable classe de l'instance $\mathbf{x}$, de sorte à ce qu'on puisse comparer $\mathbf{c}$ avec l'annotation $\mathbf{y}$ de $\mathbf{x}$. Si $\mathbf{c} = \mathbf{y}$, $\mathbf{x}$ est correctement annotée, sinon, l'annotation est une erreur. En pratique, bien sûr, les véritables classes des instances ne sont pas connues, seules les étiquettes attribuées à ces instances sont accessibles. Dans la suite, lorsque nous parlons de « vérité-terrain absolue », nous nous référons donc à une annotation des instances pour laquelle les étiquettes coïncident avec les véritables classes. Nous considérons qu'elle est en général inaccessible, et nous désignons en opposition l'annotation fournie par les annotateurs par le terme « vérité-terrain apparente ».

Cela peut être, au moins sur le principe, gênant de considérer que toute instance possède bien une unique interprétation correcte. En effet, si l'interprétation de certaines instances est subjective à l'annotateur, pourquoi considérer qu'il y a forcément eu une erreur lorsque deux annotateurs sont en désaccord ? Pour pouvoir légitimer la suite du raisonnement, nous avons donc besoin de faire une hypothèse supplémentaire. Nous allons considérer, peu importe l'application, qu'il existe une entité endossant la *responsabilité de juger* le comportement futur du classifieur, i.e. ses prédictions après qu'il ait été entraîné. En pratique, cette responsabilité peut par exemple être d'ordre légal. Cette entité représente ainsi une autorité supérieure aux constructeurs du classifieur, qui doit respecter les règles qu'elle a définies. La multiplicité d'interprétations pouvant exister pour les instances du problème d'apprentissage devient alors caduque, car l'interprétation

de l'autorité supérieure possède une valeur de vérité supérieure au reste. Nous pensons que, par cette considération, la notion de bruit d'annotation gagne en tangibilité, ce qui appuie les réflexions autour de cette problématique. Nous appuyons ce point de vue par le développement de notre problématique dans un cadre légal et éthique au chapitre 4.

Pour l'heure nous considérons que, pour tout problème d'apprentissage, la vérité-terrain absolue correspond à l'interprétation des instances par une autorité supérieure, bien qu'elle reste inaccessible en pratique. Le bruit d'annotation correspond ainsi à chaque instance dont l'étiquette dans la vérité-terrain absolue diffère de celle qui lui a effectivement été attribuée durant la phase d'annotation.

3.2.2 Définition et modélisation

A partir de maintenant, nous nous inspirons largement des travaux de B. Frénay dans son étude bibliographique [FV14] sur le bruit d'annotation et ses conséquences sur le processus d'apprentissage d'un classifieur. Pour modéliser ce bruit d'annotation, B. Frénay suppose qu'il est le résultat d'un processus stochastique venant attribuer l'annotation $\mathbf{y}$ à un exemple $\mathbf{x}$ de la classe $\mathbf{c}$ avec une probabilité $p_{\mathbf{x},\mathbf{y},\mathbf{c}}$. Dans ce cadre, on peut envisager trois types de bruit différents :

- bruit uniforme : le modèle le plus simpliste, consistant à supposer que la probabilité d'attribution d'une mauvaise annotation $\mathbf{y}\neg\mathbf{c}$ est uniforme et complètement indépendante de l'instance : $p_{\mathbf{x},\mathbf{y},\mathbf{c}} = p$;
- bruit par classe : modèle de complexité modérée, où la probabilité d'une annotation $\mathbf{y}$ quelconque ne dépend que de la classe réelle $\mathbf{c}$ de l'instance : $p_{\mathbf{x},\mathbf{y},\mathbf{c}} = p_{\mathbf{y},\mathbf{c}}$;
- bruit par caractéristiques : le modèle le plus complexe, où la probabilité d'une annotation $\mathbf{y}$ quelconque dépend aussi bien de la classe réelle $\mathbf{c}$ que des caractéristiques de l'instance $\mathbf{x}$.

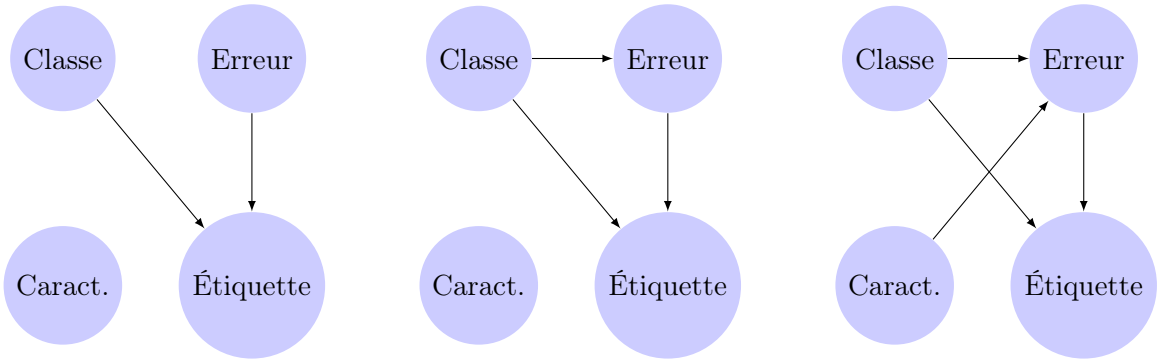


FIGURE 3.1 – Graphes illustrant la dépendance entre les différentes variables dans chaque modèle de bruit. Le nœud Erreur représente une variable binaire valant 0 s'il n'y a pas d'erreur d'annotation sur l'instance concernée, 1 sinon. L'étiquette attribuée à une instance dépend toujours de la véritable classe de cette instance, ainsi que de l'éventualité d'une erreur d'annotation commise. Ensuite, pour un bruit uniforme, l'erreur ne dépend d'aucune autre variable, tandis que pour un bruit par classe, elle dépend de la classe de l'instance. Enfin, pour un bruit par caractéristiques, l'erreur dépend en plus des caractéristiques de l'instance.

Cette taxonomie des différents bruits d'annotation avait déjà été considérée dans [NOF10].

La figure 3.1 illustre les relations de dépendances pour ces trois cas.

Le modèle de bruit uniforme est assez peu réaliste en ce qui concerne la tâche d'annotation. D'une certaine manière, c'est comme si l'on considérait que, à chaque annotation, un perturbateur choisissait de façon aléatoire entre bander les yeux de l'annotateur de sorte à ce que son annotation soit véritablement décidée indépendamment de l'instance qu'on lui présente, ou alors de le laisser annoter librement. On pourrait argumenter que la fatigue accumulée peut avoir un effet assez similaire, mais nous pensons malgré tout que son choix reste un minimum influencé par la nature de l'instance à annoter.

Le modèle de bruit par classe est quant à lui populaire étant donné sa complexité relativement faible, tout en étant bien plus réaliste que la supposition d'un bruit uniforme. Nous sommes d'avis qu'il reste cependant plus ou moins éloigné du cas réel, car la difficulté d'interprétation d'une instance, et par conséquent la probabilité qu'elle soit mal annotée, peut difficilement être réduite uniquement à sa classe. Au sein d'une même classe $\mathbf{c}$, on peut en effet trouver certaines instances dont l'appartenance à $\mathbf{c}$ ne fait aucun doute, tandis que pour d'autres, cela peut être plus ambigu. Le modèle de bruit par caractéristiques est le plus proche de la réalité, mais il est rarement considéré dans les études sur le bruit d'annotation étant donné sa complexité.

3.2.3 La matrice de transition de bruit

De manière générale, quel que soit le modèle choisi, le bruit d'annotation peut être représenté par une matrice $\mathcal{T}$, appelée *matrice de transition de bruit*, de taille $n \times n$ où n est le nombre de classes du problème. Pour tout couple (i, j), la cellule $\mathcal{T}_{ij}$ contient le nombre d'instances de la classe i munies de l'annotation j. Par exemple, pour un bruit uniforme, si l'on note η la probabilité qu'une instance soit mal annotée, l'espérance de la valeur de chaque cellule non diagonale est $\frac{\eta}{n-1}$ (Figure 3.2-A). Dans le cas de bruit par classe, la matrice est soit symétrique, si l'on suppose que $p_{\mathbf{y}=j,\mathbf{c}=i} = p_{\mathbf{y}=i,\mathbf{c}=j}$, auquel cas on parle de bruit d'annotation *symétrique*, soit quelconque dans le cas d'un bruit d'annotation *asymétrique* (Figure 3.2-B).

Une matrice de transition de bruit n'est néanmoins pas capable de capturer entièrement un modèle de bruit par caractéristiques. En effet, la dépendance entre l'annotation d'une instance et ses caractéristiques ne peut pas y apparaître naturellement. On ne peut qu'observer la dépendance entre l'appartenance d'une instance à une classe, et son annotation. Par conséquent, un modèle de bruit uniforme ou par classe est tout à fait représentable par une unique matrice de transition, tandis que pour deux modélisations différentes de bruit par caractéristiques, les matrices correspondantes peuvent être identiques. Malgré tout, l'utilisation de cette matrice offre une représentation compacte avantageuse, dans le cas d'un bruit par caractéristiques, car elle permet de calculer aisément certaines probabilités conditionnelles d'intérêt. Nous en aurons d'ailleurs l'usage au chapitre 6.

3.3 Conséquences du bruit d'annotation

Nous détaillons maintenant les conséquences que peut avoir le bruit d'annotation en classification supervisée, selon la littérature. L'analyse présentée dans cette section est largement inspirée de l'étude de B. Frénay [FV14]. Dans la littérature, l'étude de ce sujet concerne principalement la phase d'apprentissage d'un classifieur. Cependant, notre problématique nous demande d'aborder le problème posé par le bruit d'annotation lors de la phase d'évaluation, question rarement abordée. Nous semblons en effet moins sensibilisés au problème posé par un éventuel défaut dans la procédure d'évaluation d'un classifieur, tandis que nous sommes beaucoup plus enclins à accorder de l'importance aux défauts dans son apprentissage. Dans cette section, nous choisissons

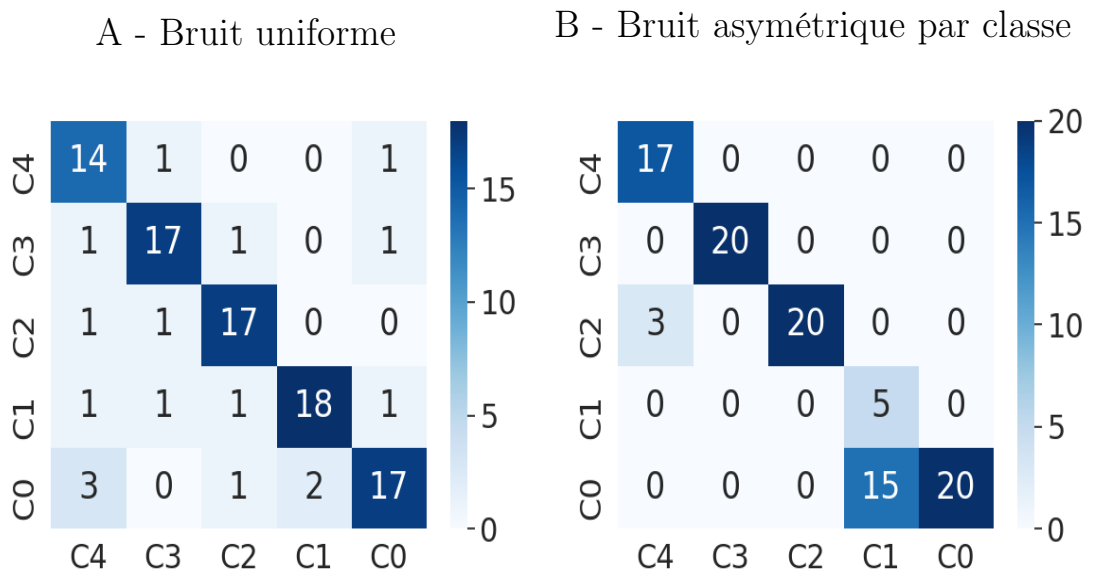


FIGURE 3.2 – Exemples de matrices de transition de bruit pour les cas uniforme (à gauche) et par classe (à droite). Le bruit par classe est ici un bruit asymétrique, avec 20% de chance que les instances de la classe 4 soient étiquetées en tant que 2, et 40% de chance que celles de la classe 1 soient étiquetées en tant que 0. Le niveau de bruit dans le cas uniforme est de 20%.

de parler uniquement de l'impact du bruit d'annotation lors de la phase d'apprentissage. Nous nous concentrons sur la problématique de l'évaluation en présence d'un bruit d'annotation dans les chapitres 4, 5 et 6. Plus précisément, le chapitre 4 a principalement pour but de sensibiliser le lecteur à l'impact du bruit d'annotation sur l'évaluation, et les chapitres 5 et 6 étudient la forme du problème et les solutions qui peuvent y être apportées.

3.3.1 Dégradation des performances des classifieurs

Lorsqu'un classifieur est entraîné avec des exemples d'apprentissage contenant du bruit d'annotation, il a plus de difficultés à acquérir une bonne description du concept. Cela se remarque par exemple simplement lorsqu'on introduit des erreurs d'annotation de manière artificielle dans l'ensemble d'entraînement, et qu'on évalue ensuite le classifieur sur les exemples de test, ces derniers n'ayant pas été bruités. Le cas échéant, la performance mesurée est sensiblement plus faible que quand le classifieur est entraîné sur les exemples d'apprentissages sans bruit d'annotation.

Plusieurs études théoriques corroborent cet impact, en l'analysant pour des modèles d'apprentissage simples, tels que l'analyse discriminante linéaire (ADL) [McL72 ; Lac79], la régression logistique [BJ10], le perceptron ou encore le modèle des k plus proches voisins [SPF97 ; WM00]. Pour le perceptron, par exemple, [Hes94] montre que dans le cas d'un bruit uniforme, représenté par la métaphore d'un professeur « distrait » fournissant les exemples à son élève avec une probabilité fixe de produire une mauvaise annotation, la performance finale d'un élève (i.e. le classifieur) n'ayant aucun regard critique sur les annotations diminue significativement jusqu'à devenir plus faible que celle de son professeur. En général, ces études concluent que la réduction des performances survient toujours, et que son importance varie de minime à critique :

— minime en particulier pour la régression logistique ou les k plus proches voisins avec un bruit d'annotation simple [Lac66 ; OY97], i.e. uniforme ou symétrique et suffisamment

faible ;
— critique pour les autres.

Les travaux empiriques mettent également en évidence, pour une plus grande variété de modèles d'apprentissage, l'impact négatif du bruit d'annotation sur la performance des prédictions ainsi que l'existence de modèle moins influencés que d'autres. Par exemple, les machines à vecteur de support (SVM) semblent largement plus impactées que les classifieurs naïfs bayésiens, où le C4.5 [NOF10 ; Pec+06]. Ces différences sont attribuées aux hypothèses faites par chaque modèle : là où les SVM supposent l'interdépendance des caractéristiques des instances, les classifieurs naïfs bayésiens utilisent des probabilités conditionnelles et supposent des relations d'indépendance conditionnelle, ce qui leur permet d'être moins affectés par les instances mal annotées. Des modèles d'apprentissage comme le *boosting* où les arbres de décision sont également influencés à cause d'un sur-apprentissage des instances mal annotées [Die00 ; Qui86]. La plupart des travaux mentionnés ici concernent un bruit d'annotation par classe, mais certaines études ont également été effectuées par rapport à l'impact du bruit d'annotation par caractéristiques sur la performance des prédictions de classifieurs [CM84]. Cependant, elles abordent uniquement le contexte de la classification binaire avec des modèles d'apprentissage simples comme l'ADL.

3.3.2 Conséquences sur l'environnement d'apprentissage

En plus de réduire la capacité d'un classifieur à construire une bonne description du concept, d'autres travaux présentent des conséquences diverses, aussi bien en théorie qu'en pratique. La présence de bruit d'annotation peut par exemple modifier les exigences nécessaires à l'algorithme d'apprentissage, en particulier la quantité d'exemples nécessaires [AL88]. De plus, la complexité des modèles construits augmente étant donné la structure artificielle créée dans l'ensemble d'apprentissage par les instances mal annotées [AM10]. Les conséquences vont encore pus loin, et affectent également des tâches annexes, e.g. qui concernent les caractéristiques des instances. En effet, [ZRB06] montre que la présence d'une seule instance mal annotée peut déjà empêcher la détermination d'environ un cinquième des caractéristiques discriminantes. Le bruit d'annotation modifie de manière complexe la stabilité de l'importance relative des caractéristiques dans un problème donné. Enfin, et de façon assez évidente, la présence d'un bruit d'annotation modifie la fréquence à laquelle chaque classe est observée, car seules les annotations sont observables, contrairement aux classes réelles, et cela est problématique pour les cas pratiques qui exigent de connaître avec précision lesdites fréquences [8].

Paradoxalement, le bruit d'annotation est parfois utilisé de manière contrôlée à des fins bénéfiques. Par exemple, il peut permettre d'empêcher ou rendre plus difficile la récupération d'informations personnelles à partir des ensembles de données [HH02]. Il permet aussi d'accroître la variabilité au sein de l'ensemble dans lequel il est introduit, ce qui permet d'augmenter l'efficacité de méthodes d'apprentissage comme le *bagging* [MS05], qui bénéficient grandement de cet aspect.

3.4 Méthodes de réduction de l'impact du bruit d'annotation lors de la phase d'apprentissage

De nombreuses méthodes ont été proposées pour combattre le bruit d'annotation en classification supervisée. On peut les séparer en trois principales catégories : la robustesse, la tolérance,

8. En médecine, par exemple, connaître la prévalence d'une maladie, i.e. sa fréquence au sein de la population, est une donnée importante pour différents types d'études.

et le nettoyage. Cette séparation n'est pas parfaite, et certaines méthodes pourraient être classées dans deux catégories à la fois. Pour une présentation plus détaillée, nous dirigeons le lecteur vers [FV14], l'article contenant notamment un tableau récapitulatif des différentes options pour gérer la présence de bruit dans l'ensemble des exemples d'apprentissage. De la même manière que pour la section précédente, nous orientons le propos de cette section sur la résolution du problème du bruit d'annotation dans le cadre de la phase d'apprentissage d'un classifieur. En ce qui concerne les méthodes pour limiter ses conséquences négatives sur la phase d'évaluation, nous invitons le lecteur à se rapporter aux chapitres 5 et 6.

3.4.1 Les méthodes de robustesse et de tolérance au bruit d'annotation

Le principe des méthodes de robustesse est d'utiliser des modèles d'apprentissage naturellement moins affectés par le bruit d'annotation. Cette robustesse peut par exemple provenir de la fonction d'optimisation utilisée : certaines sont en effet plus impactées par le bruit d'annotation que d'autres [BK09 ; MS13]. Par ailleurs, les techniques ensemblistes peuvent présenter une robustesse naturelle à la présence de bruit d'annotation. C'est le cas du *bagging*, lorsqu'il combine des modèles à l'origine très sensibles au bruit d'annotation, comme les arbres de décision. En effet, la présence de bruit d'annotation a pour effet d'augmenter la variabilité de ces classifieurs, ce qui est profitable pour la méthode ensembliste [Die00].

De manière assez similaire, les méthodes dites tolérantes au bruit d'annotation consistent à apprendre la structure de ce bruit en même temps que la description du concept. Leur intérêt provient de cette séparation entre la modélisation des données et la modélisation du bruit, ce qui permet d'utiliser d'éventuelles connaissances sur la nature particulière du bruit d'annotation, selon l'application. Ces méthodes peuvent tout d'abord être probabilistes. Elles adoptent alors un point de vue bayésien ou fréquentiste [JGC95 ; GW92 ; Hai96 ; Swa+04 ; Esk00 ; LS01 ; Lar+98 ; Sig+02], ou font usage de *clustering* [BGO09 ; RB07 ; ESP06] ou de fonctions de croyance [TGE11 ; Den00 ; Den08]. Le point de vue bayésien est appuyé par l'avis de certains scientifiques selon lesquels, en règle général, il est nécessaire d'avoir une connaissance a priori pour aborder le problème du bruit d'annotation, connaissance pouvant justement être modélisée de manière bayésienne. De manière analogue, des méthodes de *clustering*, non supervisées, peuvent être utilisées pour tenter d'estimer la forme de la distribution de probabilité a priori des véritables classes des instances. Il en est de même pour les méthodes utilisant les fonctions de croyances ou le point de vue fréquentiste : l'objectif est toujours de déterminer une distribution de probabilité antérieure des classes réelles, et de l'utiliser lors de l'apprentissage pour faire en sorte que le modèles d'inférence apprennent sur ces classes, bien qu'elles restent inconnues en pratique. Enfin, d'autres méthodes tolérantes choisissent plutôt de modifier la fonctionnement d'un modèle d'inférence particulier, de sorte à prendre en compte la présence de bruit d'annotation. Par exemple, [Lin+04 ; AL13] intègrent de la logique floue à une SVM, modèle initialement vulnérable au bruit d'annotation, ce qui permet de le rendre plus tolérant aux instances mal annotées.

3.4.2 Les méthodes de nettoyage

Les méthodes de nettoyage servent à détecter quelles sont les instances mal annotées, de sorte à les retirer de l'ensemble d'apprentissage, ou encore à réduire leur importance par l'utilisation de pondérations, ou même à les corriger en modifiant leur annotation. Bien que ces méthodes puissent sembler attrayantes sur le principe, il est en pratique difficile de correctement détecter les exemples bruités, et en particulier de les différencier des véritables exceptions existantes au sein de l'ensemble des exemples du problème.

Une partie de ces techniques se fonde sur l'utilisation de mesures particulières [Sun+07 ; GLD96 ; GLD00]. Par exemple, des mesures d'entropie peuvent être employées pour savoir si le classifieur effectue sa prédiction avec une grande confiance, auquel cas l'annotation de l'instance concernée peut être remplacée par la dite prédiction. D'autres méthodes mettent à profit le fait que certains modèles d'apprentissage sont hautement influencés par la présence d'erreurs d'annotation [MBN06 ; Hes00 ; Zha+09]. Par exemple, sachant que le bruit d'annotation augmente la complexité de la description du concept acquise par des modèles tels que les arbres de décision, on peut considérer qu'un exemple est bruité si son ajout à l'ensemble d'apprentissage augmente anormalement la complexité de ce type de modèle. De manière générale, tout modèle d'apprentissage influencé significativement par le bruit d'annotation peut permettre de déterminer les instances bruitées en les incluant dans l'ensemble d'apprentissage et en procédant ensuite par introspection du modèle construit.

Certaines techniques font plutôt usage d'une notion de proximité entre les caractéristiques des instances. C'est le cas de la méthode des k plus proches voisins, qui permet d'utiliser les *clusters* obtenus pour détecter les instances mal annotées [Har68 ; Gat72 ; DC04 ; FMN10]. En effet, si une instance se trouve dans un *cluster* principalement formé d'instances d'une autre classe, elle peut être identifiée comme bruitée. De manière analogue, des méthodes par graphe de voisinage, où les exemples, représentés par des nœuds, sont liés entre eux lorsqu'ils sont suffisamment proches les uns des autres, peuvent aussi être appliquées [SPF97 ; DU09 ; LMZ02].

Une autre méthode consiste à utiliser directement les prédictions d'un modèle entraîné sur la totalité des exemples, de manière à ce que les instances mal classées soient simplement considérées comme du bruit : c'est ce qu'on appelle un filtrage par classification [GLG99 ; Tho+08]. Cette technique est généralisée par un point de vue ensembliste, consistant à entraîner différents classifieurs de sorte à ce que, pour une instance donnée, chacun puisse *voter* pour l'annotation la plus probable. Les votes, i.e. les prédictions de chaque modèle, sont combinés pour décider quels exemples peuvent être jugés comme étant bruités. On parle de filtrage par votes [BF+96 ; BF96 ; BF99]. Des filtres plus sophistiqués, comme le filtre par partitionnement [ZWC03 ; ZWC06], ont ensuite été développés.

Les méthodes de nettoyage présentent l'avantage certain que les instances mal annotées et correctement détectées n'ont ensuite aucune influence sur le modèle entraîné si elles sont retirées de l'ensemble d'apprentissage. Certains travaux ont empiriquement montrés qu'il était souvent préférable de retirer les instances détectées que de tenter de les corriger [Mir+09 ; CHS07]. Néanmoins, d'autres travaux montrent que l'application de ces méthodes peut facilement résulter en la sur-détection d'exemples non bruités et utiles pour l'apprentissage du concept [Ten00 ; Ten01]. En particulier, lorsque le problème d'apprentissage est déséquilibré, i.e. certaines classes sont largement sous-représentées par rapport à d'autres, les instances bien annotées des classes minoritaires sont très facilement détectées par ce genre de méthodes [Sei+14], ce qui rend l'apprentissage du concept encore plus difficile. Au lieu de complètement retirer les exemples détectés, il est également possible de pondérer leur importance selon le doute à leur sujet, comme le font les méthodes utilisant des fonctions de croyance. Il est enfin envisageable de retirer uniquement leur annotation, et d'appliquer une procédure d'apprentissage semi-supervisée, bien que nous n'ayons pas connaissances de travaux allant dans ce sens.

3.5 Conclusion

Nous avons défini la notion de bruit d'annotation, de matrice de transition de bruit, et avons présenté trois modélisations de bruit différentes : bruit uniforme, bruit par classes et bruit par

caractéristiques. Nous avons également abordé :

— le danger posé par le bruit d'annotation pour la phase d'apprentissage d'un classifieur ;
— les méthodes développées pour le pallier : robuste, tolérante ou de nettoyage.

Notre problématique concerne l'impact du bruit d'annotation sur la phase d'évaluation, ce dont nous n'avons pas parlé dans ce chapitre. En effet, l'objectif de ce dernier est bibliographique : nous y avons présenté un portrait de ce qui a été majoritairement étudié dans la littérature. À l'inverse, la question de notre problématique y est rarement posée. C'est pourquoi nous consacrons les trois prochains chapitres à cette étude, en commençant par introduire le sujet, dans le chapitre 4, selon un point de vue particulier : celui de l'éthique et de la législation. Nous voulons de cette manière sensibiliser le lecteur à notre problématique, en établissant un lien entre questionnements éthiques et légaux d'un côté, et impact du bruit d'annotation sur une procédure d'évaluation de l'autre.

Chapitre 4

Aspects légaux et éthiques dans le secteur de l'IA

Sommaire

Si le chapitre précédent a introduit la notion de bruit d'annotation, et présenté les conséquences sur la phase d'apprentissage d'un classifieur, l'impact sur la phase d'évaluation reste à clarifier. Étant donné que cette question est peu souvent abordée dans l'état de l'art, nous sentons le besoin de lui donner plus de légitimité. L'objectif de ce chapitre est précisément là : légitimer notre questionnement au-delà d'un contexte purement scientifique, en l'encrant dans un cadre sociétal.

Intuitivement, on peut penser que des instances mal annotées dans l'ensemble de test risque de brouiller le jugement que l'on porte au sujet d'un classifieur évalué sur cet ensemble. Le cas échéant, il est important de se questionner sur le danger que représente ce classifieur lors de son application en société, e.g. sur les plans éthique et légal. Nous développons ces aspects dans ce qui suit, en utilisant la plupart du temps le terme « système autonome » ou « système prédictif » pour parler d'un classifieur, car le propos s'applique plus généralement à tout système pouvant avoir un impact sur les citoyens ou la société à travers des décisions ou des actions prises de manière automatiques. Nous terminerons par introduire la notion de biais, de sorte à réunir sous un même terme les conséquences négatives qu'entraîne le bruit d'annotation.

4.1 Systèmes autonomes : de la littérature populaire aux législations contemporaine

Les réflexions sur le danger des systèmes autonomes pour l'homme ne sont pas récentes. Parmi les premières occurrences dans la littérature, on trouve par exemple le golem, un être capable de se mouvoir de manière autonome mais sans libre-arbitre, et dont le seul but est de protéger son créateur. On le retrouve particulièrement dans la mythologie juive. Il est raconté que sur le front de la créature était écrit *emet* en hébreux, signifiant « vérité », qui en effaçant la première lettre, donnait *met*, la « mort », permettant de le renvoyer à l'état de poussière si cela devait se révéler nécessaire, e.g. pour protéger les hommes au cas où le monstre deviendrait incontrôlable.

Les auteurs de science-fiction des années 30 abordaient d'ailleurs fréquemment le thème des machines et autres créations se retournant contre les hommes. Pour n'en citer qu'une, l'œuvre de Mary Shelley, Frankenstein [She18], publiée en 1818, a été adaptée en long-métrage en 1931. C'est à cette même époque qu'un célèbre écrivain du nom d'Isaac Asimov, contrarié par cette image négative attribuée aux machines, formula en 1942 trois lois [Asi04] qui constituent probablement la première tentative contemporaine sérieuse de développer une réflexion sur les mécanismes de défense et garde-fous que les hommes peuvent mettre en place au sein des systèmes autonomes qu'ils conçoivent. Asimov écrivit plusieurs nouvelles de fiction sur les robots dans lesquels ses lois et leurs ambiguïtés purent être éprouvées dans de multiples situations. Cela lui permit de les modifier pour arriver à un total de quatre lois :

— Loi Zéro : un robot ne peut pas porter atteinte à l'humanité, ni, par son inaction, permettre que l'humanité soit exposée au danger ;
— Première Loi : Un robot ne peut porter atteinte à un être humain, ni, restant passif, permettre qu'un être humain soit exposé au danger, sauf contradiction avec la Loi Zéro ;
— Deuxième Loi : Un robot doit obéir aux ordres que lui donne un être humain, sauf si de tels ordres entrent en conflit avec la Première Loi ou la Loi Zéro ;
— Troisième Loi : Un robot doit protéger son existence tant que cette protection n'entre pas en conflit avec la Première Loi, la Deuxième Loi ou la Loi Zéro.

Asimov ne restreint pas la portée de ces lois à l'unique domaine de la robotique. Il précise lui-même [Asi04] qu'elles sont censées être applicables à n'importe quel outil, et a fortiori, les systèmes auxquels nous nous intéressons sont bien évidemment concernés. Ces lois ont par ailleurs constitué la base de certaines chartes sur l'éthique en robotique comme en intelligence artificielle. Ce fut le cas en particulier pour la Corée du Sud en mars 2007 [GZ09], où le gouvernement prévu de concevoir une charte sur les robots, annonçant qu'elle reflétera les trois lois d'Asimov. De la même manière, en France, en janvier 2020, le député Pierre-Alain Raphan déposa une proposition de loi relative à une charte de l'intelligence artificielle et des algorithmes [20b]. L'article 2 de cette dernière est composé des lois d'Asimov. Bien sûr, ces lois dans leur forme d'origine restent uniquement un outil de réflexion philosophique dont Asimov se servit lors de la création de son œuvre de fiction. Elles sont difficilement applicables pour spécifier la conception technique de systèmes autonomes. Elles permettent néanmoins de servir de fondation dans le domaine de l'utilisation éthique des machines, et des travaux scientifiques poursuivent dans cette voie en proposant des versions de ces lois orientées de manière à être utilisable concrètement lors du développement de tels systèmes [MW09].

4.2 Cinq axes de développement pour une pratique éthique de l'IA

Loin des peurs spéculatives de certains contemporains d'Asimov concernant des scénarios catastrophes où la machine prend le pouvoir aux hommes, une partie des acteurs du secteur de l'intelligence artificielle s'interrogent donc sur l'aspect éthique de l'utilisation au quotidien de systèmes autonomes. Dans des sociétés où de tels systèmes nous guident dans notre façon de vivre, de consommer et de prendre des décisions, à l'instar des moteurs de recommandations de contenu personnalisé, ou des systèmes de décision automatique au sujet de l'obtention de crédits ou d'emplois, la partialité illégitime de l'IA est au cœur du débat. Dans son rapport « Donner un sens à l'intelligence artificielle : pour une stratégie nationale et européenne », C. Villani consacre un chapitre entier à l'étude de cette question [Vil+18], proposant ainsi un cadre éthique de réflexion sur une meilleure approche de l'IA. Selon lui, celle-ci devrait s'articuler autour de cinq axes :

1. la transparence et l'auditabilité des systèmes autonomes ;
2. la création de droits collectifs sur les données ;
3. la responsabilité légale des personnes ou organisations déployant de tels systèmes devant les potentiels dommages causés ;
4. la formation éthique des futurs concepteurs de ces systèmes ;
5. la mise en place d'une instance de débat, plurielle et ouverte sur la société, permettant de décider démocratiquement la direction que nous souhaitons pour l'IA au sein du pays.

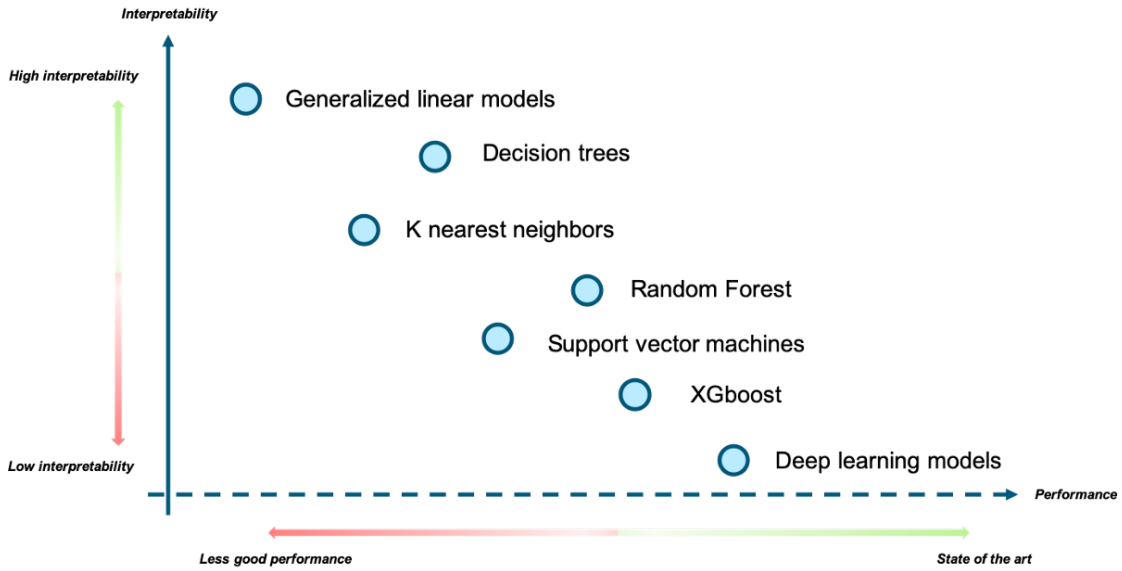


FIGURE 4.1 – Graphe comparant l'explicabilité de différents modèles d'inférence aux performances qu'ils permettent d'atteindre (crédits à [Kow+19]).

4.2.1 Transparence et auditabilité des systèmes autonomes

Le premier point fait fortement écho à un des plus grands défis scientifiques de notre domaine : *l'explicabilité* des décisions prises par les systèmes autonomes. Lorsqu'on parle d'un système

prenant des décisions en fusionnant un nombre d'informations variant d'une poignée, pour des modèles d'inférence très simples, à des centaines de millions, pour les modèles les plus complexes, i.e. ceux développés dans le cadre de l'apprentissage profond, qui sont les plus performants en général, notre capacité à expliquer les choix du système chutent rapidement (figure 4.1). Cela n'est pas si surprenant : en l'espace de quelques années, le domaine de l'IA est passé d'une époque où des règles d'inférence étaient écrites une à une pour spécifier un modèle déductif précis, à celle que nous connaissons actuellement, où ces règles ne sont plus formulées par l'homme, mais plutôt apprises automatiquement à partir d'un trop grand nombre de caractéristiques pour être assimilable par la seule compréhension humaine. Néanmoins, il est difficile d'accepter que des décisions inexplicables soient prisent au sein d'une société, lorsqu'elles peuvent impacter la vie des citoyens. La question de l'explicabilité de l'IA est donc un enjeu majeur, que nous détaillons dans le chapitre 5.

4.2.2 Création de droits collectifs sur les données

Cet axe s'établit en réaction au déphasage existant entre la législation actuelle sur les données et la nature des systèmes autonomes. En effet, la logique mise en place par le droit français est orientée sur une lourde protection des données privées des citoyens pour éviter tout abus potentiel[78 ; VV17]. Face à cela, le développement d'un système autonome nécessite un grand nombre de données, non seulement pour construire une description du concept en adéquation avec le plus de situations possibles, mais également pour être capable d'évaluer ce système sur de nombreux nouveaux exemples. En particulier, pour identifier correctement qu'un système réagit mal à certaines situations réelles, il est nécessaire de le tester autant que possible sur des données de toutes sortes. La difficulté d'obtention de celles-ci dans certains secteurs peut compliquer cette tâche pourtant nécessaire à la sécurité des citoyens.

4.2.3 Responsabilité légale au sujet des dérives des systèmes autonomes

Mieux définir le responsable en cas de dérive d'un système autonome est une urgence du domaine du droit en IA. En effet, la clarification de cet aspect reste pour l'instant insuffisante. Bien que la loi informatique et libertés [78] et le RGPD [VV17] orientent la réflexion dans ce sens, ces textes se concentrent essentiellement sur la spécification d'un cadre d'utilisation des données personnelles des citoyens, ce qui, comme nous l'avons mentionné ci-avant, nécessiterait d'être adapté pour mieux refléter la nature des systèmes autonomes.

De nombreux travaux sont récemment apparus pour proposer des solutions de maintien du caractère personnel des données, quand bien même on les utilise à des fins d'apprentissage [Pap+16 ; BH02 ; Rin97 ; FJR15 ; Sho+17]. Cependant, ces techniques sont encore jeunes et échouent pour l'instant à rendre véritablement inaccessibles les informations des individus concernées. Par ailleurs, la législation actuelle sur la protection des données s'applique uniquement aux systèmes faisant explicitement usage de données à caractère personnel et ayant une influence directe sur les personnes. Seulement, les algorithmes d'intelligence artificielle qui manipulent d'autres types de données peuvent également avoir des effets significatifs sur des groupes d'individus particuliers. L'exemple d'Amazon le montre bien : la société a été l'objet d'une controverse du fait que certains quartiers à population noire dominante aient été défavorisés par leur système de gestion des livraisons. Le déséquilibre ne provenait pas d'une utilisation directe de données de type racial, mais était dû à leur approche entièrement conduite par des éléments statistiques tels que la concentration géographique d'utilisateurs premium [Bru18].

De façon générale, faire la distinction entre données privées ou non a peu de sens dans

une ère centrée autour des technologies de l'information, comme l'expliquent les travaux d'H. Nissenbaum [Nis09]. En effet, toute donnée est en réalité un objet contextuel synthétisant de multiples informations, y compris des informations typiquement personnelles. Autrement dit, dès lors qu'un système fait usage de données au sens large, l'agrégat de l'ensemble de ces informations sur lequel ce système fonde sa réponse peut parfaitement refléter des discriminations ou autres biais de jugement en lien avec des caractéristiques personnelles, quand bien même les données utilisées ne sont pas directement de l'ordre de la vie privée. En somme, comme l'annonce C. Villani, « beaucoup de ces enjeux soulevés par les algorithmes constituent aujourd'hui un angle mort du droit ».

Il est donc urgent de proposer un écosystème plus cohérent avec la mise en place de ces systèmes autonomes dans le respect des droits fondamentaux des citoyens. Bien que ces algorithmes ne font que répéter les biais existant d'ores et déjà dans notre société, tarder à instaurer des garde-fous pourrait rapidement résulter en un fort rejet de ces technologies par un grand nombre de groupes d'individus. Cela est d'autant plus pressant que ces systèmes sont déjà utilisés pour des tâches prédictives critiques telles que la surveillance de la criminalité, la justice, les armes autonomes, i.e. autant de secteurs très vulnérables à de potentielles dérives.

Par rapport à cela, C. Villani mentionne la nécessité de développer l'audit des IA, par la création d'une autorité administrative indépendante d'experts capable, en cas de suspicion ou de litige, de procéder à tous les tests nécessaires concernant aussi bien les systèmes de prédiction que les bases de données ayant servies à les construire. De manière complémentaire, il pourrait également être bénéfique d'exiger de ceux qui les déploient de mener eux-mêmes au préalable des études de leur système, selon des protocoles communs et précis. Cela a récemment été mis en place par le groupement européen des autorités de protection des données (G29), qui demande à ce qu'un PIA (Privacy Impact Assessment) soit effectué dès que le système concerné est susceptible d'avoir des effets sur la vie privée de personnes. De cette manière, il relève de la responsabilité du constructeur de s'assurer que le maximum a été mis en œuvre pour identifier et corriger ces impacts, et de pouvoir le justifier en cas de contrôle. De manière analogue, il est possible d'envisager une étude d'impact sur les discriminations et autre biais engendrées par l'utilisation du système.

Bruit d'annotation et subjectivité d'interprétation

Nous avons parlé au chapitre 3 de la subjectivité d'interprétation des données. Cela rend la notion de bruit d'annotation compliquée à définir, car elle dépend de l'existence d'une vérité-terrain absolue, qui associe à chaque donnée une unique interprétation possible. Nous avons par conséquent admis que dans le cadre de cette thèse, la vérité-terrain absolue existe toujours, et nous l'avons justifié en considérant qu'il existe une autorité supérieure qui la fournit, quelle que soit l'application. La législation, ou tout organisme s'y rapportant, peut jouer le rôle de cette autorité. Cela ne signifie pas que la vérité-terrain absolue devienne accessible, ni qu'elle soit parfaitement définie, car il arrive que la législation soit floue. Cependant, la notion de bruit d'annotation gagne en tangibilité lorsqu'elle est définie par rapport à une interprétation légale.

4.2.4 Sensibiliser et responsabiliser la société sur le sujet de l'IA

Les deux derniers axes représentent l'importance de sensibiliser la population aux problématiques éthiques et légales sur l'IA. D'une part, il est primordial que les futurs ingénieurs en IA, et autres étudiants dans les domaines connexes, soient formés à identifier, prévenir et corriger

les biais pouvant apparaître à toute étape du processus de construction d'un système autonome, et ce à l'aide d'outils et de méthodes précises et solides. D'autre part, « la capacité d'évaluation et d'audit des IA ne peut être réservée à un organe public, mais doit aussi provenir de la société civile ». Selon C. Villani, pour l'acceptation de l'usage de ces systèmes, le citoyen doit pouvoir se maintenir au courant de l'évolution de cet écosystème, et les organismes de défense d'intérêts civils et autres associations en matière d'IA doivent avoir les moyens financiers et culturels de mener par eux-mêmes l'audit de systèmes autonomes. Cela implique plus de transparence et d'échange de données entre les différents acteurs du secteur. Si ce type d'initiatives peut sembler ardu à mettre en œuvre, il existe déjà aux États-Unis, à l'instar du média d'investigation de référence en matière de libertés numériques, ProPublica [ABN11].

4.2.5 Audit des IA : exemple

Le cadre éthique décrit par C. Villani amène donc à imaginer la conception d'un système de prédiction de manière plus prudente et consciencieuse. Prenons l'exemple concret de la conception d'une voiture autonome, pour imaginer à quoi pourrait ressembler en pratique un tel cadre.

Le fonctionnement d'une voiture autonome demande la collaboration d'un grand-nombre de sous-systèmes prédictifs. Un système de reconnaissance et de suivi d'objets dans un flux vidéo est certainement nécessaire : cela lui permet d'identifier dans son environnement où se trouvent les autres utilisateurs de la route, i.e. véhicules ou piétons, de sorte à pouvoir agir en conséquence. Si ce sous-système présente un défaut de prédiction, qui reste invisible durant la phase d'évaluation, il est très probable qu'un accident survienne dès lors que des centaines de ces voitures sont déployées dans les rues.

C'est précisément ce qui est arrivé pour une voiture autonome de la compagnie Uber, premier exemple de cette technologie à avoir tué un piéton [Wak18]. Le défaut de prédiction venait du fait que la caractéristique sur laquelle le système se fondait pour déterminer la présence d'un piéton à éviter était que l'objet détecté se trouvait ou non sur un passage piéton. Les récalcitrants à respecter le code de la route étaient donc invisibles aux yeux du système, et ce biais a causé la mort de l'un d'entre eux. Bien sûr, le fait que ce problème n'ait pas été détecté durant la phase de test est la preuve de la présence d'un autre biais, bien plus dangereux : celui d'un en semble de test non représentatif de la réalité. En effet, si la procédure de test avait confronté le système à de telles situations, i.e. où les piétons ne se tiennent pas forcément là où on les attend, l'accident ne serait probablement pas arrivé. On en revient donc à une des conditions devant être respectée durant la construction d'un classifieur (cf. chapitre 2) : l'ensemble des exemples, surtout ceux utilisés pour l'évaluation, doit être représentatif de la réalité à laquelle ce classifieur est confronté lors de son déploiement. De façon analogue à l'exemple d'Uber, il est possible d'envisager d'autres défauts de prédiction qui pourraient ne pas être relevés à cause d'un manque dans l'évaluation. On peut par exemple imaginer que le même système de détection et suivi d'objet ne soit pas capable de reconnaître une personne portant un gros parapluie la couvrant de manière trop importante, où alors ne puisse pas identifier des animaux tels que des chiens ou des chats sur la route.

Imaginons maintenant un autre sous-système de prédiction pour voiture autonome, qui consisterait plus généralement à prédire si la situation dans laquelle se trouve la voiture sur la route implique un danger ou non, et le cas échéant, à envoyer l'ordre de ralentir. La situation est donc représentée par quatre flux vidéos, filmant les évènements à l'avant, l'arrière, et sur les deux côtés de la voiture, et nous voulons entraîner un classifieur binaire à différencier les situations où il faut ralentir des autres, à partir des données des flux vidéos et de la vitesse actuelle de la voiture. Exprimée de cette manière, cette tâche est foncièrement ambiguë, et le point de vue

d'un annotateur pourrait facilement différer de celui d'un autre [9].

Supposons que la phase d'annotation, puis les phases d'entraînement et de test aient malgré tout lieu, et que, suite au déploiement du système, un accident se produise. L'autorité d'audit des systèmes autonomes pourraient alors se saisir du problème. Les concepteurs se verraient dans l'obligation de fournir leur code source, la totalité des données utilisées lors de la construction du classifieur, la version du programme ayant commis l'accident, toute les preuves nécessaires pour montrer que la compagnie a mené les procédures d'évaluation et de prévention requises, à l'instar du PIA, et éventuellement un protocole complet précisant en détails comment reconstruire de manière identique le système, à partir des données et du code source, incluant même les informations telles que les graines de génération de pseudo-nombres aléatoires utilisées [10]. Des tests de différentes sortes seraient alors menés. En particulier, on peut envisager que des experts soient mandatés pour vérifier la représentativité des ensembles d'entraînement et de test, ainsi que la qualité de l'annotation des instances.

Si les experts se montrent plus prudents et rigoureux sur la spécification de la tâche d'annotation que l'ont été les concepteurs, il est possible qu'ils identifient plusieurs instances mal annotées. Pour peu qu'une étude soit menée au sujet de l'impact de ce bruit d'annotation sur la qualité de l'apprentissage et la validité de l'évaluation [11], et qu'elle conclue à une corrélation significative entre cet aspect et l'accident, les conséquences pourraient être désastreuses pour la compagnie.

Dans cet exemple, nous parlons de *défauts* de prédictions et d'évaluation. Nous allons à présent changer cette appellation, en introduisant les termes de *biais de prédiction* et *biais d'évaluation*.

4.3 Le biais dans le secteur de l'IA

Les questionnements légaux et éthiques que l'on se pose dans ce chapitre entrent en écho avec un problème majeur identifié dans le domaine de l'apprentissage automatique, dont nous avons parlé à de multiples reprises jusqu'à maintenant : la notion de biais. Il est possible d'argumenter que dans toute procédure incluant une intervention humaine, ou alors ayant pour cible des humains, les biais sont inévitables et par conséquent, ne sont pas nécessairement négatif, faisant partie intégrante de la nature humaine. Cependant, il est indéniable qu'il existe des biais indésirables selon des principes sociaux, éthiques ou encore légaux [12]. C'est précisément la présence de ce genre de biais qui créa récemment une prise de conscience progressive au sein de multiples communautés, allant des scientifiques aux politiciens. La réflexion de C. Villani le montre pour le côté politique, mais les mêmes points sont actuellement soulevés dans la littérature scientifique, par exemple par O.C. Jenkins et D. Lopresti [JLM20].

9. En réalité, cette tâche serait plutôt formulée dans un cadre d'apprentissage semi-supervisé ou par renforcement, mais pour l'exemple, nous l'abordons de façon complètement supervisée.

10. Dans le cas général, il faudrait toute l'attention nécessaire pour éviter d'être trompé par des compagnies mal intentionnées, qui fourniraient une fausse version de leur système, à l'image du scandale au sujet des compagnies Renault et Volkswagen, qui avaient triché pour les tests antipollution de leurs voitures.

11. Concrètement, les experts pourraient par exemple comparer l'évaluation du classifieur avec l'ensemble de test d'origine, ainsi qu'avec le même ensemble où les erreurs d'annotation sont corrigées, et où des instances de test supplémentaires sont rajoutées pour le rendre plus représentatif, ce qui permettrait de mieux identifier les failles du système.

12. Il a par exemple déjà été montré à maintes reprises que les décisions des juges pouvaient être inconsciemment influencées par leurs propres caractéristiques, ou encore que les recruteurs attribuaient des entretiens à des fréquences variables aux candidats possédant des résumés identiques, mais des noms reflétant une appartenance à des ethnies différentes.

Définition général d'un biais

Dans ce qui suit, lorsque nous parlons de biais négatifs, nous désignons en général une *tendance à interpréter* une caractéristique, ou un groupe de caractéristiques, d'une façon qui diffère d'une certaine *attente*. Ce biais est dit indésirable lorsque cette attente correspond à une norme qu'il est nécessaire de respecter, selon le cadre ou l'application. Les biais de pensée connues dans notre société (e.g. ethniques, religieux, de classes financières, etc.) sont par exemple indésirables selon des normes éthiques ou légales.

4.3.1 Biais de prédiction et biais d'évaluation

Nous distinguons deux type de biais par rapport à la construction d'un système prédictif :

— le biais de *prédiction*, qui se rapporte à la manière dont les prédictions sont effectuées, et qui est acquis par le système au cours de son apprentissage ; il désigne toute tendance du *système* à interpréter certaines données de façon différente de la description du concept idéale ; il devient indésirable lorsqu'il entre en conflit avec la description du concept attendue par une autorité supérieure ;

— le biais d'*évaluation*, qui se rapporte à notre aptitude à détecter et mesurer la présence d'un biais de prédiction lors de la phase d'évaluation ; il désigne toute tendance d'une *procédure d'évaluation* à interpréter les statistiques calculées d'une façon non représentative de la véritable performance du système pour la tâche attendue ; il devient indésirable lorsqu'il empêche de se rendre compte des biais de prédiction qui le sont également.

Formellement, le biais de prédiction correspond à la situation où un classifieur $\mathcal{C}$ est confronté à une instance $\mathbf{x}$ de la classe $\mathbf{c}$, éventuellement étiquetée par $\mathbf{y}$ si cette instance fait partie du problème d'apprentissage, et où ce classifieur effectue une prédiction qui diffère de $\mathbf{c}$. La catégorie $\mathbf{c}$ correspond ici à l'interprétation idéale de l'instance $\mathbf{x}$. Il est ici important de remarquer que la prédiction de $\mathcal{C}$ peut tout à fait être équivalente à l'étiquette $\mathbf{y}$. Le cas échéant, l'exemple $(\mathbf{x}, \mathbf{y})$ se trouve être en réalité bruité, et le classifieur $\mathcal{C}$ s'est simplement rangé du côté de la mauvaise interprétation qui a été fournie par les annotateurs pour l'instance $\mathbf{x}$.

Le biais d'évaluation est différent du biais de prédiction, dans la mesure où il ne se rapporte pas directement à ce que le classifieur prédit, mais plutôt à la façon dont la procédure d'évaluation juge de la justesse de ses prédictions et conclut au sujet de sa performance. Ce dernier biais peut donc concerner aussi bien un classifieur en accord parfait avec la description du concept de l'autorité supérieure, i.e. ne présentant aucun biais de prédiction, qu'un classifieur effectuant quant à lui des prédictions inadéquates par rapport aux normes à respecter. Dans les deux cas, il est en effet possible que la phase d'évaluation ne permette pas d'estimer correctement la performance du système, et ce pour deux raisons :

— le manque de variabilité dans les exemples de test, impliquant que le système ne peut être évalué pour certaines situations critiques et nécessaires pour bien appréhender sa véritable performance ;

— le bruit d'annotation, pouvant amener de bonnes prédictions du système à être considérées comme étant mauvaises, et inversement.

Formellement, le biais d'évaluation correspond au sens mathématique du terme, i.e. le biais de l'estimateur de la mesure mise en œuvre pour évaluer la performance du classifieur. Nous détaillons cet aspect dans le chapitre 6.

4.3.2 Sources des biais : algorithmes et données

La présence d'un biais peut provenir de différentes sources, mais elles présentent toutes un point commun : elles concernent en premier lieu les données collectées, et en second lieu la manière de traiter et utiliser ces données.

Dans le premier cas où le biais provient directement des données, les raisons sont multiples :

— les exemples peuvent inclure des caractéristiques représentant directement des biais intolérables, comme le genre ou l'ethnie, et pour peu que le modèle considère que l'utilisation de ces caractéristiques l'aide à obtenir une meilleur performance, ces biais se retrouveront dans ses prédictions ; e.g. un modèle entraîné sur des articles de journaux peut aisément assimiler certains stéréotypes apparents dans les sujets abordés par les médias ;

— même si les exemples ne contiennent pas explicitement ce genre de caractéristiques, ces biais peuvent être encodés de manière complexe par les autres variables du problème, laissant la possibilité à l'algorithme de les retrouver par lui-même ; e.g. le cas des livraisons de la compagnie Amazon, effectuées au détriment de certains quartiers à population raciale particulière sans que le système se fonde explicitement sur les données ethniques des utilisateurs ;

— la manière dont les instances sont récoltées est également importante : dans le secteur de la police prédictive, par exemple, collecter des données dans les quartiers où la concentration de policier est supérieure à la normale implique l'obtention de plus de données concernant des crimes et délits dans les mêmes lieux pour des raisons statistiques, ce qui peut conduire à un cercle vicieux où il serait décidé d'accroître la surveillance de ces zones ;

— enfin, les données générées et/ou annotées par des humains peuvent illustrer certains de leurs biais de jugement ou de raisonnement, à l'image de l'exemple fictif précédent concernant les voitures autonomes, où le modèle a assimilé une description de la notion de danger sur la route correspondant à celle des annotateurs, mais différente de celle exigée par les normes d'une autorité supérieure, ce qui se révèle problématique au moment où il se retrouve soumis à l'arbitrage de cette autorité.

Dans le second cas, on parle de *biais algorithmique*, dans le sens biais *de source* algorithmique [13]. En principe, l'utilisation d'algorithmes aurait le potentiel de réduire le biais présent dans de nombreux processus impliquant des humains [Kle+18]. Mais, en pratique, les faits montrent que ce n'est pas aussi simple, ces algorithmes pouvant également amplifier les biais, si des précautions insuffisantes sont prises durant leur conception et leur déploiement. Des travaux récents tentent d'établir une pratique de l'IA consciente de ces problèmes et respectant la notion *d'équité* [Mad+20 ; Bir+20 ; FR21], pour mitiger au mieux les conséquences du biais en IA, mais le champ n'en est qu'à ses débuts, et le recul acquis est encore insuffisant. En particulier, la définition même d'équité pour un modèle, ainsi que la manière de la mesurer, est encore floue et sujette à débat [SM19].

Nomenclature des différents biais

Dans notre réflexion, nous avons fait le choix de différencier deux types de biais, que nous avons appelé biais de prédiction et biais d'évaluation. Ces termes désignent la *cible* du biais. Nous venons par ailleurs de parler de biais algorithmiques et de biais provenant des données. Ces appellations se réfèrent à la *source* des biais. Par exemple, un biais algorithmique entre dans la catégorie d'un biais de prédiction, car il concerne la manière

13. L'exemple de la voiture Uber utilisant une caractéristique non pertinente pour la détection de piétons en est une illustration.

dont les prédictions du système sont effectuées. Nous pouvons faire l'analogie entre notre choix de nomenclature et la psychologie cognitive, qui étudie la présence de biais cognitifs chez l'homme. En effet, notre biais de prédiction correspondrait plutôt à ce qu'on appelle un biais de raisonnement, tandis que notre biais d'évaluation se rapporterait à la notion de biais de jugement.

4.3.3 Impact du bruit d'annotation dans la formation des biais

Nous venons de présenter une liste non exhaustive de différentes causes de l'apparition d'un biais dû aux données, dans un système prédictif. On peut maintenant se demander quel rapport cela a à voir avec le bruit d'annotation, élément essentiel de notre problématique. À ce sujet, le dernier point de ladite liste mentionne la présence potentielle, dans un ensemble de données, des biais de jugement ou de raisonnements des annotateurs, émanant de leurs annotations. Celles-ci peuvent ainsi être considérées comme bruitées, dans la mesure où l'autorité supérieure responsable du jugement d'un système prédictif ne cautionne certainement pas les biais que ces annotations peuvent contenir. De ce point de vue, le bruit d'annotation devient effectivement une source de biais.

En ce qui concerne le biais de prédiction, l'impact du bruit d'annotation est étudié dans la littérature sous la forme de la dégradation des performances des classifieurs (cf. chapitre 4). Récemment, les travaux de Geva et al. [GGB19] tentent d'aller plus loin dans l'étude de cet impact, en y incluant la dimension subjective de l'annotation des données. Ils posent la question suivante : en apprentissage automatique, modélise-t-on la tâche, i.e. le concept à apprendre, ou bien l'annotateur ? L'étude menée s'attache à mesurer ce qu'elle nomme le biais d'annotation, i.e. le biais de prédiction d'un modèle ayant sur-appris la subjectivité des annotateurs des exemples d'entraînement qui lui ont été présentés. Ces exemples sont tirés d'ensembles de données textuelles, pour des tâches de traitement automatique du langage, i.e. des situations plus complexes qu'un simple problème de classification. Les conclusions sont multiples :

1. la performance d'un modèle augmente lorsque, dans les caractéristiques des instances, on inclut explicitement l'identité de l'annotateur, ce qui indique que cette identité est une information utile pour la tâche de prédiction ;
2. même en gardant l'identité des annotateurs secrète pendant l'apprentissage, le modèle est capable d'apprendre à reconnaître l'annotateur d'un exemple donné, y compris d'un exemple qu'il n'a pas vu lors de son entraînement, pour peu qu'il en ait vu d'autres provenant du même annotateur ;
3. enfin, la performance d'un modèle est sensiblement plus basse lorsque les annotateurs des exemples d'entraînement ne sont pas les mêmes que ceux des exemples de test, ce qui confirme d'autant plus la présence d'un biais d'annotation.

L'étude met en évidence ce problème uniquement pour le champ du traitement automatique du langage, où les annotateurs contribuent à la création complète des exemples, e.g. pour une tâche de question-réponse où leur rôle est d'écrire des questions ainsi que les réponses associées. Des éléments comme le style d'écriture d'un annotateur permettent alors facilement de l'identifier à travers les exemples qu'il a créé. Cela implique également que moins il y a d'annotateurs différents, plus la variabilité des données est faible, et donc plus les points susmentionnés prennent de l'importance.

Cependant, rien ne prouve que l'identité de l'annotateur ne puisse également transparaître à travers l'annotation des instances dans le cadre d'autres tâches de classification plus simples. Si l'on fait les hypothèses suivantes :

— la subjectivité d'un annotateur est encodée dans les annotations qu'il fournit (en accord avec le point 1) ;
— un classifieur est en mesure d'acquérir la même subjectivité, et donc de la reproduire sur des exemples nouveaux (en accord avec le point 3) ;
— cette subjectivité prend la forme d'un bruit d'annotation, car elle est incompatible avec la description du concept de l'autorité supérieure.

Dans ce cas, on peut considérer que le bruit d'annotation implique un biais de prédiction indésirable, car les prédictions du classifieur sont influencées par l'interprétation subjective de l'annotateur émanant des exemples d'apprentissage.

L'étude de Geva et al., ainsi que les hypothèses que l'on peut établir sur des cas de classification plus simples, montrent également que le bruit d'annotation peut créer un biais d'évaluation. En effet, si un modèle ayant appris à reproduire l'interprétation subjective d'un annotateur est évalué sur des exemples présentant la même subjectivité, sa performance peut apparaître plus élevée que ce qu'elle n'est en réalité.

Plus simplement, lorsqu'un bruit d'annotation est présent dans les exemples d'entraînement, et qu'un classifieur apprend à faire les mêmes erreurs sur de nouveaux exemples, un ensemble de test contenant un bruit d'annotation similaire implique la présence d'un biais d'évaluation. Ceci ne relève pas simplement de l'ordre de l'hypothèse. L'étude récente de C. G. Northcutt et al. [NAM21] a en effet révélé que dans 10 ensembles de données parmi les plus utilisés de nos jours dans la recherche, 3.4% des exemples en moyenne étaient bruités. L'étude concerne entre autres les ensembles CIFAR10, CIFAR100 et ImageNet, fréquemment utilisés pour des tâches de classification d'images. Les auteurs ont mis à disposition un site web [20a] recensant toutes les erreurs d'annotation identifiées dans chaque ensemble de donnée. Ils conseillent enfin d'adopter une position plus prudente vis à vis de l'évaluation en apprentissage automatique, en proposant d'évaluer sur des ensembles de test corrigés au préalable, en particulier pour des applications dans le monde réel.

4.4 Conclusion

Nous avons en premier lieu présenté les réflexions contemporaines sur l'éthique et la législation au sujet de l'IA, et nous avons fait le lien entre ces considérations et la notion de bruit d'annotation. En particulier, définir qui assume la responsabilité dans le cas d'une dérive commise par un système prédictif implique de créer une autorité de référence en matière de litige. Cette autorité a donc le pouvoir de trancher entre une bonne ou une mauvaise interprétation au sujet d'une donnée d'un problème d'apprentissage, et peut donc représenter la vérité-terrain absolue, permettant ainsi de définir le bruit d'annotation.

Nous avons ensuite développé le sujet du biais en IA, pour lequel nous avons défini une nomenclature particulière, à savoir les biais de prédiction et d'évaluation. Le premier se rapporte au biais de raisonnement d'un système prédictif, par analogie au biais de raisonnement d'un homme, tandis que le second concerne notre tendance à mal estimer la véritable performance du système. Nous avons enfin mis en évidence le lien pouvant exister entre le bruit d'annotation et l'apparition de ces biais lors de la construction et de l'évaluation d'un classifieur. Notre problématique peut donc se remanier pour se rapporter à l'étude de l'impact du bruit d'annotation sur le biais d'évaluation d'une procédure de test.

Chapitre 5

Méthodes de réduction du biais d'évaluation

Sommaire

Résumons tout d'abord brièvement ce que nous avons vu jusqu'ici. Nous avons commencé par établir l'intérêt scientifique de la phase de test dans un problème d'apprentissage (Cf. chapitre 2), puis nous avons présenté le phénomène de bruit d'annotation, et ses conséquences néfastes dans le cadre de l'apprentissage supervisé (Cf. chapitre 3). Ensuite, dans le but de justifier l'étude de l'impact du bruit d'annotation sur la confiance en le résultat d'une phase de test, nous avons parlé des conséquences de procédures de test biaisées sur la société, et du biais d'évaluation que pouvait justement induire la mauvaise annotation des données lors du test d'un classifieur (Cf. chapitre 4). Cependant, le problème que pose le bruit d'annotation pour l'évaluation de classifieurs étant un sujet de recherche assez marginal, il est difficile de distinguer clairement dans la littérature les mesures pouvant être prises pour le pallier.

Ce chapitre vise par conséquent à établir une synthèse de ce que l'état de l'art offre comme options pour diminuer l'impact du bruit d'annotation lors de la phase de test, i.e. sur le biais d'évaluation. Comme nous l'avons expliqué lors du chapitre 4, ce biais d'évaluation est en pratique dû à un décalage entre la réalité apparente symbolisée par les exemples de test, et la réalité

effective du cadre d'application futur d'un classifieur. Bien que ce décalage s'explique souvent par le manque de variabilité des données de test, le bruit d'annotation peut l'accentuer davantage. Les méthodes que nous présentons dans la suite, bien que parfois conçues dans un but différent, peuvent être mises en œuvre pour réduire la contribution du bruit d'annotation au biais d'évaluation.

5.1 Prévention de l'apparition d'un biais d'évaluation

La question que l'on se pose dans cette partie est la suivante : quels leviers permettent de réduire le problème posé par le biais d'évaluation avant la phase de test ? En effet, la présence d'un biais d'évaluation peut plus fréquemment amener à mettre en service un classifieur qui, en réalité, incorpore des biais de prédiction. Nous nous interrogeons sur les techniques permettant d'empêcher cela, et applicables avant la phase de test.

Les précautions dont nous allons parler concernent donc la phase d'annotation et d'apprentissage, ou surviennent avant celles-ci. Comme nous nous plaçons dans un cadre où les biais d'évaluation et de prédiction sont supposés dus au bruit d'annotation affectant les instances d'entraînement et de test, ces précautions consistent essentiellement à améliorer la qualité de ces instances, de sorte à naturellement réduire la formation de ces deux biais. De plus, sachant que l'objectif de la réduction de l'impact du biais d'évaluation est d'obtenir à terme un classifieur incorporant un minimum de biais de prédiction (car une évaluation non biaisée permet plus facilement de détecter et d'écarter les autres), nous nous intéressons également aux techniques limitant directement la formation des biais de prédiction lors de la phase d'apprentissage. Rappelons que selon notre définition, un biais de prédiction se rapporte à une erreur de prédiction du classifieur lorsqu'on le compare à la vérité-terrain absolue, laquelle est inaccessible en général.

5.1.1 Bonnes pratiques

En premier lieu, il y a un certain nombre de bonnes pratiques, issues des réflexions développées autour du thème du biais en apprentissage automatique, qu'il convient de mentionner au sujet de la phase d'annotation. Nous pensons en particulier aux lignes directrices suivantes, énoncées par Thomas dans [Rac18], par rapport au biais pouvant être introduit par les annotateurs lors de l'annotation des instances :

- la création d'un ensemble d'instances doit être accompagnée d'une description précise du protocole de récolte de ces instances, ainsi que des directives transmises aux annotateurs pour les étiqueter ;
- les questionnements légaux et éthiques que se sont posés les auteurs de l'ensemble, ainsi que les solutions mises en œuvre pour offrir des réponses convenables à ces questions, doivent être documentés ;
- composer une équipe d'annotateurs aux profils variés, de sorte qu'ils soient sensibles, ou enclins, à des biais différents.

L'objectif de ce genre de règles est double. D'une part, l'accès à toute méta-information sur la création d'un ensemble d'instances permet de mieux comprendre ces dernières, de sorte à correctement identifier si leur utilisation lors de la phase d'apprentissage ou de test est réellement pertinente pour le problème à résoudre, ou si elles nécessitent d'être préparées d'une façon particulière. D'autre part, réduire l'introduction de biais indésirables par la main des annotateurs contribue à accroître l'aspect qualitatif de l'ensemble d'instances, i.e. son niveau de représentativité du concept qui lui est associé. Nous n'avons malheureusement pas connaissance de procédés

ayant mis en œuvre de telles directives en établissant des mesures pour quantifier le gain qualitatif de l'ensemble de données.

5.1.2 Quantité *vs.* qualité

Bien qu'il soit difficile de quantifier la qualité d'un ensemble d'instances, nous pouvons cependant mentionner les travaux de Lam et Stork [LS03], qui ont abordé la question suivante : vaut-il mieux privilégier un ensemble de test de bonne qualité, contenant des exemples annotés soigneusement par des experts, au détriment de la quantité de ces exemples, ou à l'inverse préférer utiliser un ensemble de test mal annoté par de la main d'œuvre peu chère mais contenant de nombreux exemples ?

Les auteurs arrivent à la conclusion que l'impact négatif d'une annotation de mauvaise qualité est contrebalancé par la présence en grand nombre des exemples de test. Cependant, leur étude a été menée sous l'hypothèse d'un bruit d'annotation uniforme. Cela ignore le problème de la dépendance entre le biais de prédiction d'un modèle et le biais d'évaluation, et nous en parlons plus en détails au chapitre 6. Par ailleurs, étant donné les réflexions récentes sur les exigences de qualité des systèmes d'IA [Mad+20 ; Bir+20 ; FR21 ; JLM20], l'aspect qualitatif des données d'apprentissage et de test apparaît de plus en plus essentiel.

5.1.3 Favoriser l'indépendance entre ensemble d'entraînement et de test

Enfin, nous aimerions ajouter à la liste de bonnes pratiques précédente le conseil formulé dans [GGB19] : les annotateurs des exemples d'entraînement devraient être distincts des annotateurs des exemples de test. L'objectif est simple : limiter la dépendance entre les deux ensembles. Nous savons qu'une règle essentielle d'une bonne phase de test est de présenter au classifieur des exemples qui n'ont joué aucun rôle lors de l'apprentissage, pour lesquels le seul lien avec les exemples d'entraînement doit être que leurs caractéristiques proviennent d'une même distribution. Le fait qu'une instance destinée à faire partie de l'ensemble d'apprentissage, et qu'une instance gardée pour la phase de test, ait toutes deux été annotées par le même annotateur, peut constituer un lien indésirable lors de l'évaluation du classifieur. Les travaux des auteurs de [GGB19] le montrent expérimentalement au moins pour le domaine particulier du traitement du langage (Cf. chapitre 4). Nous n'avons pas connaissances d'étude menée sur cette question concernant d'autres domaines, cela constitue donc une perspective intéressante pour de futurs travaux.

Les bonnes pratiques de la phase d'annotation

Pour résumer, les points à respecter lors de la phase d'annotation pour réduire le biais d'annotation, i.e. introduit par les annotateurs, sont :

- accompagner l'ensemble de données d'une description précise du protocole de récolte de ces instances, ainsi que des directives transmises aux annotateurs pour les étiqueter ;
- documenter les questionnements légaux et éthiques que se sont posés les auteurs de l'ensemble, ainsi que les solutions mises en œuvre pour offrir des réponses convenables à ces questions ;
- composer une équipe d'annotateurs aux profils variés, de sorte qu'ils soient sensibles, ou enclins, à des biais différents.
- penser à la forme du bruit d'annotation potentiel dans l'ensemble de donnée obtenu :

s'il ne respecte pas les hypothèses d'un bruit uniforme, le nombre d'exemples de test ne doit pas être prioritaire par rapport à la qualité de l'annotation fournie ;
— séparer les annotateurs de l'ensemble d'apprentissage de ceux de l'ensemble de test, pour éviter l'apparition d'un lien de corrélation indésirable ente les deux ensembles.

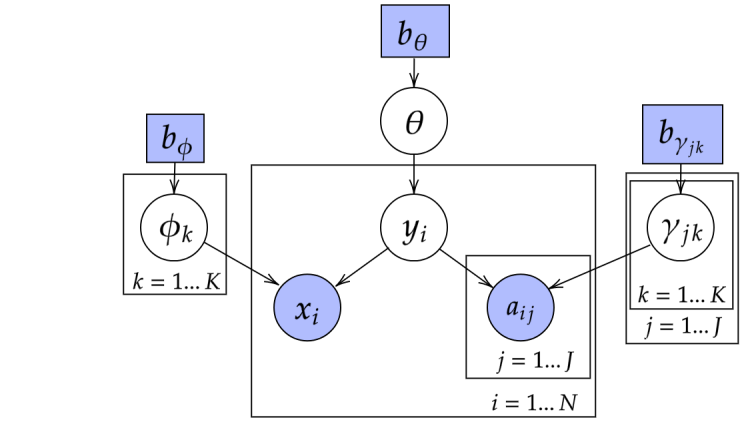
5.1.4 Évaluer les annotateurs et construire automatiquement la vérité-terrain

Pour augmenter la qualité de l'annotation des données et réduire le bruit d'annotation, il est également possible d'envisager l'utilisation de systèmes automatisés dédiés, embarqués au sein de plateformes d'annotation collaboratives. En effet, comme on observe en pratique que les annotateurs humains sont tous plus ou moins imparfaits, le principe de tels systèmes est d'utiliser plusieurs annotateurs pour le même exemple, de sorte à confronter leurs jugements et, d'une part, déterminer les profils d'erreurs de chaque annotateur, pour avoir une idée de leur fiabilité, et d'autre part, obtenir une annotation plus informative à terme, en attribuant par exemple un poids à chaque instance annotée, représentant la confiance qu'on lui octroie et dépendant de la fiabilité des annotateurs impliqués. L'utilisation d'un accord inter-annotateurs est d'ailleurs fondé sur le même principe. Pour appliquer de manière systématique ce genre de méthode qui consiste à estimer la fiabilité des annotateurs, et agréger leurs annotations pour produire une vérité-terrain de qualité supérieure, le modèle *item-réponse* a été développé par Dawid et Skene [DS79]. Ce modèle a connu de multiples extensions [Car08 ; PR13 ; Hov+13 ; LMW12 ; RY12 ; WD+11 ; Whi+09 ; Zho+12], et a été adapté dans des cadres différents. En particulier, nous pouvons nous servir du cadre bayésien pour en expliquer le principe [Fel+14]. La figure 5.1, provenant du même article, montre un modèle bayésien génératif du problème :

— le vecteur θ contient la proportion de chaque classe dans l'ensemble d'instances du problème de classification (inconnu en pratique) ;
— le vecteur γ_{jk} spécifie la distribution de probabilité sur les K étiquettes possibles pour que l'annotateur j annote une instance de la classe k avec une étiquette donnée (inconnu en pratique) ;
— le vecteur ϕ_k représente la proportion d'apparition de chaque caractéristiques des instances (inconnu en pratique) ; cela suppose que toutes les caractéristiques sont de type discret ; dans le cas contraire, des densités de probabilité peuvent être utilisées ;
— y_i est la vraie classe de l'instance i (inconnue en pratique) ;
— le vecteur x_i correspond aux caractéristiques de l'instance i ;
— a_{ij} est l'annotation attribuée par l'annotateur j à l'instance i ;
— les variables b_θ, $b_{\gamma_{jk}}$ et b_ϕ sont les distributions a priori des variables correspondantes, θ, γ_{jk} et ϕ_k, choisies manuellement.

Le graphe de la figure 5.1 représente un modèle génératif dans le sens où il permet d'assigner des valeurs de probabilité à chaque variable, comme si elles avaient été générée selon le processus suivant :

— en premier lieu, les vecteurs θ, ϕ_k et γ_{jk}, pour chaque classe k et chaque annotateur j, sont générés à partir de leurs distributions a priori ;
— ensuite, pour chaque instance i, la vraie classe y_i est générée par une distribution catégorique de paramètre θ, et les caractéristiques x_i sont générées par une distribution multinomiale paramétrée par ϕ_{y_i} ;
— enfin, les annotations de chaque annotateur pour l'instance i sont générées par une distribution multinomiale de paramètre γ_{j,y_i}.



$J :=$ *nombre d'annotateurs*
$K :=$ *nombre de classes*
$N :=$ *nombre d'instances*
$F :=$ *nombre de caractéristiques*

$$dim(\theta) = K \qquad dim(a_{ij}) = K$$
$$dim(\phi_k) = F \qquad dim(x_i) = F$$
$$dim(\gamma_{jk}) = K \qquad y_i \in \{1...K\}$$
$$dim(\gamma_j) = K \times K$$

$$\theta \sim SymDir(b_\theta)$$
$$\forall k : \phi_k \sim SymDir(b_\phi)$$
$$\forall j, k : \gamma_{jk} \sim Dir(b_{\gamma_{jk}})$$
$$\forall i : y_i | \theta \sim Cat(\theta)$$
$$\forall i : x_i | y_i, \phi \sim Multinom(|x_i|_1, \phi_{y_i})$$
$$\forall i, j : a_{ij} | y_i, \gamma_j \sim Multinom(|a_{ij}|_1, \gamma_{j,y_i})$$

FIGURE 5.1 – Caption

La formulation d'un modèle génératif spécifiant les distributions de probabilité de chaque variable permet d'appliquer des méthodes statistiques telles que l'inférence de Monte-Carlo par chaines de Markov pour estimer les valeurs les plus probables des variables cachées, à partir des informations sur le reste des variables. En particulier, en utilisant les distributions a priori choisies, ainsi que les vecteurs de caractéristiques des instances, et les étiquettes attribuées à chaque instance par différents annotateurs, il est possible d'estimer les vraies classes y_i, ainsi que les vecteurs γ_{jk} spécifiant la tendance à effectuer des erreurs d'annotations sur des instances de la classe k pour chaque annotateur j.

Malheureusement, la grande majorité des extensions et dérivés du modèle item-réponse ne semble pas avoir été implémentée et testée en pratique par des travaux postérieurs. Ceux de Felt et Haertel [Fel+14], dont le modèle génératif que nous venons de présenter est tiré, et qui proposent également une amélioration de leur modèle, capable de faire usage de *clusters* d'instances non annotées, sont évalués sur un problème de classification de documents ayant été annotés par un petit nombre d'annotateurs différents. Les auteurs obtiennent de meilleurs résultats que plusieurs autres méthodes du même genre. Par ailleurs, Lam et Stork [LS05] ont étudié la mise en œuvre d'un modèle analogue, avec un procédé permettant de choisir de manière optimale à quels annotateurs chaque instance doit être présentée, de sorte à maximiser le gain d'information pour le classifieur à entraîner. Cela permet d'appliquer ce type de technique avec économie, en réduisant le nombre d'annotateurs nécessaires à l'annotation de chaque instance, sans impacter la qualité de la vérité-terrain obtenue par inférence, et donc de l'apprentissage du classifieur.

De manière générale, les méthodes dont nous venons de parler font partie des procédures de nettoyage de données, que nous avons présenté au chapitre 3. Le fait d'obtenir une vérité-terrain plus qualitative avec le minimum d'efforts est évidemment séduisant pour notre problématique

des biais de prédiction et d'évaluation. Cependant, comme nous l'avons mentionné dans le chapitre 3, les procédures de nettoyage ne sont pas exemptes de défauts, en général. Étant donné les deux sortes de stratégies envisageables, l'une sévère, i.e. avec un bon taux de détection des mauvais exemples, mais un taux trop haut pour les bons, et l'autre plus prudente, i.e. préservant les bons exemples mais en retirant trop peu parmi les mauvais, l'équilibre est difficile à trouver :

— dans le premier cas, un trop haut taux de détection conduirait facilement à retirer des exemples essentiels au problème de classification ; cela pourrait réduire considérablement la représentativité des ensembles d'entraînement et de test, impliquant donc la formation de biais de prédiction ou d'évaluation encore plus importants, à cause du manque de variabilité dans les ensemble ;
— dans le second cas, si le taux de détection des mauvais exemples est trop bas, les biais de prédiction et d'évaluation engendrés par le bruit d'annotation ne seraient pas suffisamment réduits.

Les techniques issues du modèle item-réponse sont en ce sens préférables, car leur supériorité par rapport aux procédures de nettoyage communes, e.g. fonctionnant par vote de majorité, a été empiriquement montrée [Fel+14].

5.1.5 Robustesse de l'apprentissage des classifieurs

Les méthodes que nous avons vu jusqu'ici concerne la phase d'annotation des instances. Par rapport à la phase d'entraînement, les techniques de robustesse des algorithmes d'apprentissages (cf. chapitre 3) peuvent être utiles pour notre objectif. Mieux vaut prévenir que guérir : éviter la formation initiale d'un biais de prédiction au sein d'un classifieur, e.g. via des algorithmes robustes au bruit d'annotation, ne doit pas être négligé.

En pratique, la robustesse d'un algorithme d'apprentissage est évaluée *a priori*, en mesurant essentiellement la perte de performance que les classifieurs construits via cette méthode subissent lorsqu'ils apprennent sur des instances bruitées de manière contrôlée [SLH16 ; KZ94 ; SLH11], de sorte à ce que les utilisateurs de cet algorithme puissent être moins inquiétés par l'influence du bruit d'annotation affectant leurs données, au moment de l'appliquer sur un nouveau problème. Le fait que cette robustesse soit évaluée sur des ensembles bien précis, avec un bruit d'annotation artificiel, a des limitations. En effet, cela ne peut certifier que ces méthodes sont tout aussi robustes lorsqu'elles sont appliquées sur d'autres ensembles de données, avec un bruit d'annotation réel. Nous pouvons admettre que cette limitation ne concerne pas les modèles dont la robustesse au bruit est formellement étudiée, mais ce genre d'étude est rare, et se restreint à des modèles d'inférence et de bruits d'annotation simples (cf. chapitre 3).

En somme, les efforts sur la réduction des biais dû au bruit d'annotation ne sauraient se restreindre à la simple utilisation de méthodes robustes lors de la phase d'apprentissage. Mitiger l'impact du bruit d'annotation doit également être effectué lors des procédures de test, par des approches adaptées.

5.1.6 Conclusion

Cette première partie a couvert différentes options permettant de réduire le phénomène de bruit d'annotation ainsi que ses conséquences sur la phase d'évaluation, de manière préventive, i.e. avant que le classifieur ne soit prêt à être évalué.

Les plans d'action pour combattre le biais d'évaluation causé par le bruit d'annotation

On peut distinguer trois plans d'action pour combattre le biais d'évaluation lorsqu'il est dû au bruit d'annotation :

1. rendre la phase d'annotation plus qualitative, via l'application
   — de bonnes pratiques pour la construction des ensembles d'apprentissage et de test, e.g. la séparation des annotateurs de chaque ensemble ;
   — de procédures de nettoyage pour l'obtention d'une nouvelle vérité-terrain plus qualitative ; en particulier, les méthodes fondées sur le modèle item-réponse sont plus efficaces et complètes, étant donné qu'elles permettent aussi bien de construire cette vérité-terrain que d'estimer les profils d'erreurs des annotateurs [Fel+14] ; cela permet dans les deux cas de réduire les biais de prédiction et d'évaluation ;
2. rendre la phase d'apprentissage plus robuste en utilisant des algorithmes adaptés ;
3. rendre la phase d'évaluation plus robuste (cf. section 5.2).

Le premier point permet de réduire aussi bien les biais de prédiction que d'évaluation. Le deuxième point réduit en premier lieu le biais de prédiction, et par conséquent limite le danger de la présence d'un biais d'évaluation. Enfin, le troisième point, que nous présentons en section 5.2, s'attaque directement au problème du biais d'évaluation, sans se préoccuper de la réduction des biais de prédiction.

Nous pouvons terminer par mentionner que la meilleure prévention face aux erreurs que peuvent commettre les systèmes d'IA dans leurs environnements d'application est de limiter leur nombre et leur déploiement dans des secteurs critiques. Silberg et. al. [SM19] considèrent à ce propos qu'il est pour l'instant nécessaire de maintenir, dans le cas général, un jugement humain dans les processus de décisions automatisés. Enfin, C. Villani conseille même de proscrire l'automatisation des systèmes là où nous n'avons pas la capacité de certifier l'absence de biais alors que l'application le nécessite [Vil+18].

5.2 Robustesse de la phase d'évaluation au bruit d'annotation

Dans cette section, nous examinons comment l'évaluation peut être réalisée de manière plus robuste au bruit d'annotation et au biais dont il est responsable. Notre propos se divise en deux parties. Dans la première, nous considérons ce qui peut rendre plus robustes nos procédures d'évaluation usuelles. Nous présentons les propositions de différents auteurs sur le sujet et discutons de l'efficacité de ces méthodes, et nous terminons par détailler une méthode d'évaluation particulière proposée dans la littérature, naturellement robuste au bruit d'annotation étant donné qu'elle ne fait pas usage des annotations des instances. Dans la deuxième partie, nous parlons d'une approche différente de la pratique empirique de l'évaluation au sein du domaine de l'apprentissage automatique. En particulier, nous abordons la notion d'explicabilité des modèles.

5.2.1 Comprendre formellement le mécanisme de biais d'évaluation

La présence d'un bruit d'annotation dans nos données de test impacte le bon déroulement de nos procédures usuelles d'évaluation.

Cet énoncé n'est pas souvent pris en compte, car l'ensemble de test est en général considéré

dépourvu d'erreurs. L'étude de O.C. Jenkins et D. Lopresti vient cependant de montrer l'inverse [JLM20]. En accord avec ces auteurs, nous avons défendu que le bruit d'annotation est un problème réel dans les ensembles de test utilisés pour l'évaluation, et que par conséquent, s'il y a un risque qu'il fausse le jugement porté sur la qualité d'un système, cela ne peut être ignoré, en particulier pour des raisons éthiques et légales. Nous n'avons cependant jamais abordé la question suivante : sous quelle forme cela se répercute-t-il lors de l'évaluation, et à quelle importance, d'un point de vue quantitatif ?

Cette question est essentielle, car la compréhension formelle de l'impact du bruit d'annotation sur l'évaluation permet de mieux appréhender la nature du problème, et d'ouvrir des pistes de réflexion pour rendre la procédure d'évaluation plus robuste. Par exemple, en médecine, lorsque des tests sont réalisés sur un nouveau traitement pour pouvoir l'administrer aux patients, un grand nombre de biais peuvent faussement avantager la mesure de son efficacité. Plusieurs pratiques existent dans le domaine pour augmenter la robustesse des tests à ces biais. Une, en particulier, consiste à réaliser l'étude « à biais maximal », i.e. toujours considérer le pire des cas, la situation qui donne injustement le plus gros avantage au nouveau traitement, et lui attribuer par conséquent un handicap adéquat. Celui-ci n'est alors accepté que s'il apparaît meilleur que le traitement de référence, malgré le désavantage qui l'accable. Robustesse est ici synonyme de prudence : un traitement ne doit pas être mis sur le marché alors qu'il est moins bon que celui reconnu comme référence.

L'étude formelle de l'impact du bruit d'annotation sur la procédure d'évaluation usuelle fera l'objet du prochain chapitre (cf. chapitre 6), dans lequel nous nous inspirons du principe de comparaison à biais maximal pour examiner comment nous pouvons évaluer et comparer différents systèmes de manière prudente lorsque les exemples de test sont mal annotés. Pour le moment, nous nous concentrons sur d'autres pratiques permettant une évaluation plus robuste aux exemples mal annotés.

5.2.2 Évaluation pseudo-supervisée

La première option pour augmenter la robustesse de la procédure d'évaluation fait naturellement suite au principe que nous avons présenté en section 5.1.4, i.e. obtenir une vérité-terrain plus qualitative de manière automatique. Ainsi, l'évaluation est réalisée de manière « pseudo-supervisée », car les annotations utilisées ne proviennent pas directement de la main de l'homme. Cormack et. al. ont par exemple mis en œuvre ce genre de méthodes à des fins d'évaluation [CK09]. Ils observent, dans le cadre de l'identification de spam, que les erreurs d'annotation des exemples de test sont problématiques par rapport à la manière dont ils biaisent les mesures de performances de leurs systèmes, pouvant même amener à considérer un système meilleur qu'un autre à tort. Ils proposent donc deux versions d'une procédure d'évaluation, l'une automatique, l'autre nécessitant l'intervention occasionnel d'un juge humain, et telles que l'impact de ces exemples mal annotés est moindre. Pour cela, une hypothèse essentielle à la validité de la méthode est que les prédictions d'un ensemble de classifieurs sont récoltés et fusionnées pour chaque exemple, et que le résultat de cette fusion forme une meilleure annotation de l'exemple, par rapport à l'annotation initiale.

Des hypothèses similaires, i.e. admises dans le cadre d'une construction automatique de la vérité-terrain, se retrouvent également dans d'autres travaux [FL17a ; FL17b ; ZL15 ; Bou+07 ; ETS01]. L'objectif est le même : mener une procédure d'évaluation de manière pseudo-supervisée. Fedorchuk et. al. proposent par exemple aussi d'utiliser les prédictions d'un grand nombre de classifieurs [FL17b] pour obtenir de nouvelles annotations, à la manière d'un système de votes, et établissent une comparaison entre plusieurs procédures d'évaluation [FL17a], certaines super-

visées, i.e. faisant usage des annotations d'origine humaine, d'autres pseudo-supervisées. Leurs résultats montrent que la différence entre les deux n'est pas flagrante. Ils concluent que cela représente un point favorable à la pratique de l'évaluation pseudo-supervisée en tant que substitut, lorsque les instances annotées sont rarement accessibles dans le domaine d'application concerné.

Des travaux concernant l'évaluation pseudo-supervisée ont d'ailleurs été effectués pour ce genre de domaines, comme le suivi d'objets dans une vidéo [ETS01], les médias sociaux [ZL15] ou encore l'imagerie médicale [Bou+07], domaines où les exemples annotés peuvent être particulièrement difficiles à récolter. Cependant, ces méthodes sont en général trop orientées sur le domaine concerné, de par les hypothèses effectuées implicitement ou non sur la forme des instances des test, du bruit d'annotation ou de la structure de la procédure d'évaluation, et sont donc difficilement adaptables à notre problème, car nous sommes intéressés par l'évaluation de classifieurs dans le cas général.

L'évaluation pseudo-supervisée est donc intéressante lorsque les données annotées sont rares, mais elle peut également avoir un intérêt pour augmenter la robustesse de l'évaluation au bruit d'annotation, à condition que la méthode appliquée consiste à reconstruire une vérité-terrain avec un mécanisme destinée à identifier et éliminer les erreurs d'annotation initialement présentes dans l'ensemble de test.

5.2.3 Évaluation non supervisée

Dans le même ordre d'idée, est-il possible d'envisager l'évaluation de manière non supervisée, sans avoir besoin de quelconques annotations ? De façon surprenante, la réponse est affirmative, à condition d'effectuer des hypothèses précises. En effet, Raj et. al. [RSB11] proposent une méthode aisément adaptable à d'autres situations de classification, qui permet de comparer deux classifieurs sans avoir besoin d'utiliser les annotations des instances. Leur proposition a été effectuée pour le domaine de la reconnaissance vocale, où les exemples annotés sont difficiles à obtenir, mais elle peut également être considérée pour mener une évaluation véritablement insensible au bruit d'annotation.

Le cas d'application considéré par les auteurs concerne un problème de classification binaire, où l'on dispose de deux classifieurs que l'on désire comparer, sans avoir accès à l'annotation des exemples de test. Le principe est de supposer que l'on possède un troisième classifieur, dit de référence, avec pour seule contrainte d'avoir une performance *meilleure qu'un processus de classification aléatoire*. Notons respectivement ces classifieurs A, B, et R. Sous l'hypothèse que ces trois classifieurs *effectuent leurs prédictions de manière indépendante*, il est possible de concevoir un test statistique permettant de comparer la justesse des classifieurs A et B sur la base des prédictions de R.

Ce test correspond essentiellement à déterminer quel classifieur entre A et B est le plus souvent d'accord avec le classifieur de référence, et à conclure que ce classifieur est le meilleur. En effet, lorsqu'on regarde la probabilité qu'un classifieur ayant un taux de succès p soit d'accord avec la référence, dont le taux de succès est noté r, on obtient grâce à l'hypothèse d'indépendance des classifieurs :

$$pr + (1-p)(1-r) \tag{5.1}$$

La dérivée de cette expression par rapport à p est $2r - 1$. Cela signifie que si $r > 0.5$, i.e. R est meilleur qu'un classifieur aléatoire, alors le taux d'accord entre les deux classifieurs augmente linéairement avec p. Ainsi, le classifieur ayant le plus haut taux d'accord avec la référence a également le meilleur taux de succès.

	M2 positif	M2 négatif	Total
M1 positif	a	b	a+b
M1 négatif	c	d	c+d
Total	a+c	b+d	a+b+c+d

TABLE 5.1 – Exemple d'un tableau de corrélation entre les résultats obtenus par deux méthodes M1 et M2 pour l'application d'un test de McNemar.

Le test statistique présenté par les auteurs est inspiré du test de McNemar [McN47]. Le test de McNemar s'applique à un tableau de corrélation (cf. tableau 5.2.3). Si l'on note p_a, p_b, p_c et p_d les probabilités relatives à chaque case du tableau, l'hypothèse nulle du test correspond à supposer que les probabilités marginales du cas positif sont équivalentes $p_a + p_b = p_a + p_c$. En notant donc H_0 l'hypothèse nulle, et H_1 l'hypothèse alternative :

$$\begin{aligned} H_0 : &\quad p_b = p_c \\ H_1 : &\quad p_b \neq p_c \end{aligned} \tag{5.2}$$

La statistique permettant de tester cette hypothèse est :

$$\chi^2 = \frac{(b-c)^2}{b+c} \tag{5.3}$$

Sous l'hypothèse nulle, et en supposant que $b+c$ soit suffisamment grand, cette quantité suit une loi du χ^2 à un degré de liberté. Par conséquent, si sa valeur est supérieure à la valeur seuil choisie, e.g. 0.05, H_0 peut être rejetée.

Dans le cas de nos trois classifieur, il suffit de construire le tableau de corrélation adéquat pour l'application d'un test de McNemar. Pour cela, les auteurs regardent les quatre évènements suivants :

— $N_{A\overline{B}R}$: le nombre d'instances sur lesquelles A et R sont d'accord, et B ne l'est pas ;
— $N_{\overline{A}BR}$: le nombre d'instances sur lesquelles B et R sont d'accord, et A ne l'est pas ;
— $N_{AB\overline{R}}$: le nombre d'instances sur lesquelles A et B sont d'accord, et R ne l'est pas ;
— N_{ABR} : le nombre d'instances sur lesquelles A, B et R sont d'accord.

Le tableau 5.2.3 présentent les corrélations des accords entre les classifieurs A et B et le classifieur de référence R. Par conséquent, en calculant la statistique χ^2 et en la comparant à une valeur seuil, on peut décider de rejeter ou d'accepter l'hypothèse H_0 : $N_{A\overline{B}R} = N_{\overline{A}BR}$, ou autrement dit, les classifieurs A et B sont de performance équivalentes. Dans le cas où l'on rejetterait l'hypothèse, le signe de $N_{A\overline{B}R} - N_{\overline{A}BR}$ indique quel classifieur est meilleur que l'autre.

	$B = R$	$B \neq R$
$A = R$	N_{ABR}	$N_{A\overline{B}R}$
$A \neq R$	$N_{\overline{A}BR}$	$N_{AB\overline{R}}$

TABLE 5.2 – Tableau de corrélation pour la comparaison par un test de McNemar de deux classifieurs A et B sur la base d'un classifieur de référence R.

L'hypothèse d'indépendance des classifieurs faite initialement est essentielle pour la méthode. Elle est également très lourde, car il est difficile de supposer que trois classifieurs entraînés à la

même tâche soient indépendants. Éventuellement, la corrélation entre leurs prédictions peut être réduite s'ils ont été construits à l'aide de modèles d'inférence sensiblement différents, et qu'ils ont été entraînés sur des instances différentes.

5.2.4 Évaluation « online »

Jusqu'ici, les procédures d'évaluation dont nous avons parlées supposent toutes d'évaluer le système préalablement à sa mise en service, à l'aide d'instances de test préparées dans cette optique. Une autre approche consiste à évaluer le système directement dans son environnement d'application [FR21]. C'est ce qu'on appelle l'évaluation « online », ou évaluation en continu, pratique provenant du domaine de l'ingénierie logicielle. En particulier, une approche fréquente est le test A/B, consistant à tester différentes versions d'un même système dans l'environnement d'application [RR18]. En opposition, une évaluation par validation croisée est typiquement considérée comme de l'évaluation « offline ».

Dans un cadre d'évaluation « offline », l'inconvénient est la difficulté à construire un ensemble de test représentatif de l'environnement d'application futur (Cf. chapitre 2 section 2.3.1). Le manque de variabilité des données de test, ainsi que le bruit d'annotation, sont deux facteurs impactant négativement cette représentativité. En ce qui concerne l'évaluation « online », le problème de variabilité des données disparaît, ce qui est clairement avantageux.

Il peut cependant être difficile d'appliquer cette approche car :

— les instances de test ne sont pas annotées, il est donc nécessaire d'avoir soit un humain pour les interpréter, soit être en mesure d'évaluer la réussite du système par un indicateur alternatif, à l'image d'un processus d'apprentissage par renforcement ;
— lorsque le système est évalué dans un environnement réel, son comportement peut mettre en danger cet environnement selon l'application considérée.

En ce qui concerne le deuxième point, une possibilité consiste à évaluer le système en simulation. Cependant, le cas échéant, le problème de représentativité de l'environnement de test fait de nouveau surface, la question qui se pose étant la fidélité de l'environnement de simulation par rapport à l'environnement d'application réel.

5.2.5 Conclusion

Dans ce qui précède, nous avons présenté plusieurs procédures d'évaluation qui, bien qu'ayant été conçues pour des raisons différentes, peuvent sur le principe aider à réduire le biais d'évaluation dû au bruit d'annotation. En particulier, nous pouvons retenir :

— l'évaluation pseudo-supervisée, lorsqu'elle consiste à reconstruire une vérité-terrain exempte d'instances mal annotées, à l'aide e.g. de méthodes ensemblistes impliquant une procédure de vote entre plusieurs classifieurs ;
— l'évaluation non supervisé, avec la méthode de Raj [RSB11] qui, dans le même ordre d'idée, permet de comparer deux classifieurs par un test statistique sur la base des prédictions d'un troisième classifieur de référence ;

Ces deux méthodes permettent de réduire, pour la première, et d'éviter, pour la seconde, le biais d'évaluation dû au bruit d'annotation. Leur efficacité est cependant soumise à des contraintes fortes : l'évaluation pseudo-supervisée nécessite que la vérité-terrain obtenue automatiquement soit en effet de meilleure qualité que celle d'origine, ce qui n'est pas trivial. La méthode de Raj, quant à elle, demande à ce que les classifieurs comparés et le classifieur de référence considéré soit indépendant par rapport aux prédictions qu'ils effectuent.

L'évaluation « online » permet également de réduire le biais d'évaluation, mais elle ne concerne pas le biais causé par le bruit d'annotation. Plutôt, elle permet de contourner le biais dû au manque de représentativité d'un ensemble de test, i.e. le décalage existant entre les exemples de test et l'environnement d'application futur du classifieur, qui est la cause première du biais d'évaluation. Cependant, les données de test présentées au classifieur n'étant pas annotées, il est nécessaire de disposer d'une manière de valider ou non les prédictions effectuées, soit à l'aide d'un opérateur humain, soit par un indicateur de performance alternatif.

5.3 D'une évaluation empirique vers une évaluation rationnelle ?

Suite à l'utilisation croissante des systèmes d'IA en société, et à leurs comportements parfois inattendus et indésirables, la question du pourquoi s'est inévitablement posée. Comment expliquer que ces modèles, pourtant entraînés à l'aide d'une multitude d'exemples de la tâche à réaliser, puissent parfois prendre des décisions tellement différentes de ce qui est attendu ? La réponse immédiate est qu'en général, on ne sait pas. Plus précisément, pour les modèles obtenant les meilleurs performances, les réseaux de neurones profonds, nous ne savons pas expliquer clairement ce qui les mène à prendre de mauvaises décisions. A ce sujet, Mallat, dans les cours de la chaire « Science des données » du collège de France [Mal17], annonce que les réseaux de neurone sont des boîtes noires, dont notre recherche a perfectionné les performances à une vitesse impressionnante pendant la dernière décennie, trop vite pour que le champ des mathématiques puissent développer la théorie qui leur est associée de sorte à pouvoir expliquer leurs résultats. Malgré tout, être capable d'expliquer de manière *rationnelle* la décision prise par un modèle devient urgent, étant donné que nos moyens d'évaluer la performance de nos modèles échouent trop souvent à empêcher la mise en service de systèmes au comportement inadéquat, et à fournir une explication satisfaisante lors d'un incident causé par un tel système. En réponse, différentes techniques ont été et continuent à être développées dans le cadre de ce qui est communément nommé « l'explicabilité des réseaux de neurones » [SWM17 ; DSB17].

5.3.1 L'explicabilité des réseaux de neurones

Le champ de l'explicabilité en IA s'est récemment développé pour permettre l'obtention d'une explication humainement compréhensible des décisions et mécanismes de raisonnement des systèmes d'IA. En réalité, des techniques d'explicabilité existent depuis les débuts de l'IA [Goe+18]. Ce n'est cependant qu'avec le succès de l'apprentissage profond que l'importance de cette pratique est devenue claire.

Principe général de l'explicabilité

Le potentiel de ces techniques est large, comme l'expliquent Samek et. al. [SWM17] :

— permettre la vérification des biais du système, et en particulier respecter la législation, i.e. les problématiques qui nous intéressent ;
— améliorer davantage le système, car c'est en en comprenant ses faiblesses qu'on peut en concevoir de meilleures versions, et comparer différents classifieurs de manière plus précise ;
— apprendre à l'aide du système : maintenant que les modèles sont entraînés sur des millions d'exemple d'une tâche, il est fréquent qu'ils soient confrontés à des cas méconnus de l'expertise humaine, et comprendre comment ces modèles prennent leurs décisions dans ces cas précis peut permettre aux domaines concernés d'apprendre et d'avancer.

Nous avons suffisamment parlé et exemplifié le premier point lors du chapitre 4. Le deuxième point a un rapport direct avec le biais d'évaluation. Plusieurs auteurs ont en effet remarqué que des classifieurs pouvaient avoir la même performance, lorsqu'on les évalue par leurs justesses, et néanmoins s'appuyer sur des ensembles de caractéristiques totalement différents [Arr+16 ; Arr+17 ; Lap+16]. Dans une telle situation, identifier le modèle adéquat à un cadre d'application particulier nécessite des techniques d'explicabilité. Enfin pour le troisième point, Alpha Go [Koh20], l'intelligence artificielle qui a battu Lee Sedol, le meilleur joueur du monde de Go, en constitue un bon exemple. Lors de son duel avec le champion, au milieu de la partie, le système a joué un coup étrange, qui a surpris les spectateurs et a été considéré comme assez mauvais. Ce ne fut qu'à la fin de la partie qu'il devint clair pour tous les professionnels que c'était essentiellement ce coup qui avait mené Alpha Go à la victoire. Un de ces experts a annoncé à ce sujet que « ce coup n'était pas humain ». Suite à cet évènement historique dans l'histoire du jeu comme dans le domaine de la recherche en IA, ce coup a été étudié par de nombreux experts de Go [Koh20].

Il y a différentes techniques utilisables pour expliquer les décisions d'un modèle, mais toutes se fondent sur un même principe : étudier comment le modèle réagit aux données d'entrée, et l'expliquer d'une manière ou d'une autre. La réaction du modèle peut être observée en sortie, ou alors localement pour chaque filtre à l'intérieur du modèle, ou chaque neurone. Pour l'expliquer, des scores sont souvent attribués aux caractéristiques des instances en entrée, pour reconnaître lesquelles jouent un rôle majeur dans les prédictions du modèle. D'autres méthodes expliquent les prédictions d'un modèle par des exemples similaires présents dans l'ensemble d'instances [Gur+19 ; KKK16], ou encore produisent directement une explication rationnelle en langage humain, dans le cadre des modèles de traitement automatique du langage [YCC10]. Une taxonomie des différentes techniques est accessible dans le papier [Ary+19].

Un outil efficace : l'analyse de sensibilité

Beaucoup de ces techniques s'appuient sur de l'*analyse de sensibilité* [SVZ13 ; Bae+10]. L'analyse de sensibilité est un nom général regroupant différentes méthodes permettant d'étudier la sensibilité de la réponse d'un modèle, lorsque ses données en entrée varient. Plus précisément, on considère l'espace mathématique auxquelles les entrées du modèle appartiennent, et le modèle lui même est vu comme une fonction de cet espace vers un espace de sortie, en général $\mathbb{R}^n$ pour les scores d'un problème de classification à n classes. L'objectif de l'analyse de sensibilité est alors d'expliquer la prédiction d'un modèle pour une entrée $\mathbf{x}$, en utilisant les dérivées partielles de cette fonction par rapport à chaque caractéristique x_i, de sorte à obtenir un score R_i quantifiant l'importance individuelle de ces caractéristiques dans la prédiction du modèle :

$$R_i = \left\| \frac{\partial f(\mathbf{x})}{\partial x_i} \right\| \tag{5.4}$$

Ce genre de méthodes permet en général de créer une visualisation de l'importance des caractéristiques sous forme de carte thermique, ce qui est particulièrement efficace dans les problèmes de classification concernant la vision par ordinateur [Hen+16]. Il est néanmoins important de remarquer qu'une hypothèse essentielle à ces techniques est que les caractéristiques les plus importantes dans la décision du système sont celles auxquelles sa réponse est la plus sensible, lorsque celles-ci varient. Cela peut être problématique dans certains cas. Par exemple, imaginons qu'une voiture se trouve sur une image, et que le classifieur prédise correctement sa présence. De plus, devant la voiture se trouve un objet dissimulant une partie du véhicule. Il est probable que les pixels correspondant à cet objet soient considérés comme les plus important, étant donné qu'en les faisant varier de sorte à reconstruire la voiture dans son entièreté, l'incertitude du modèle dans

sa prédiction devrait diminuer au maximum, ce qui se refléterait dans le score de classification attribué à la catégorie voiture, i.e. sa réponse. Il y a cependant des moyens plus sophistiqués de mettre en œuvre les outils d'analyse de sensibilité, à l'image de la *rétro-propagation de pertinence par couche* [Bac+15]. Cette dernière méthode ne souffre pas de ce problème, car elle permet de directement décomposer la réponse du modèle en la contribution de chacune de ses entrées.

Des travaux font bon usage de ces techniques pour expliquer le comportement de réseaux de neurones complexes, à l'image des méthodes du nom de LIME [RSG16] et SHAP [LL17], qui ont été mises en œuvre pour détecter des biais discriminatoires au sein de modèles de prédiction à impact direct sur les citoyens. Bien sûr, ces méthodes sont encore à l'aube de leur développement, et possèdent de nombreuses imperfections et vulnérabilités à des situations malchanceuses ou malveillantes, comme le montrent Slack et. al. dans [Sla+20].

Évaluer la qualité d'une méthode d'explicabilité

Il existe donc de nombreuses alternatives pour tenter d'expliquer les modèles de réseaux de neurones, et dans un tel contexte, il est nécessaire d'être en mesure d'évaluer quelles méthodes sont de plus grande qualité, et permettent de fournir une meilleure explication des prédictions d'un modèle. Pour cela, Samek et. al. proposent une mesure de la qualité des méthodes produisant un score d'importance pour les caractéristiques des exemples, se fondant sur les principes suivants [Sam+16] :

- lorsqu'on perturbe les caractéristiques les plus importantes pour les prédictions du modèle, la performance de classification globale de ce modèle est censée diminuer de manière plus significative, que lorsqu'on perturbe des caractéristiques de moindre importance ;
- les scores obtenues via une méthode d'explicabilité permettent de former un classement d'importance des caractéristiques ;
- il est alors possible de perturber de manière itérative les caractéristiques dans leur ordre d'importance, et relever la diminution en performance du modèle à chaque étape ; le classement conduisant à la diminution de performance la moins importante en moyenne, correspond à la méthode de plus grande qualité.

Ainsi, le procédé pour évaluer la qualité de ces méthodes d'explicabilité, dont l'objectif est d'identifier les biais de prédiction d'un système, y compris ceux potentiellement causés par le bruit d'annotation, implique sans surprise de s'appuyer sur une mesure de performance empirique dépendant de l'annotation des instances de test. Par conséquent, l'évaluation de ces méthodes est également vulnérable au bruit d'annotation.

L'explicabilité peut être vu comme une tentative de changement de paradigme dans notre approche à l'évaluation. Ces solutions apparaissent en effet prometteuses pour identifier les biais de prédiction d'un modèle, de façon complémentaire aux procédures d'évaluation usuelles. Par exemple, dans le chapitre 4, nous avons parlé des liens de dépendance complexes qui apparaissent entre les différentes caractéristiques des instances, et donc de la difficulté à empêcher l'apparition de biais de prédiction, pouvant être dû à l'utilisation de caractéristiques discriminatoires. L'application de techniques d'explicabilité permettrait d'identifier plus facilement ce problème, dans la mesure où elles aident à déterminer sur quelles caractéristiques le système s'appuie lors de ses décisions.

5.3.2 Vérification formelle du comportement d'un modèle

Par ailleurs, et de façon complémentaire, un autre champ de recherche étudie la possibilité de prouver formellement si un systèmes d'IA possède un comportement adéquat à des spécifications

précises [Kat+17 ; Wen+18 ; Sin+18]. Le domaine de la preuve formelle est complexe, mais son application aux problématiques dont nous parlons depuis le chapitre précédent pourrait se révéler prometteur. En effet, être en mesure de spécifier formellement les attentes que l'on a pour un système, et de prouver que ce système y répond, serait certainement bienvenue dans le champ des méthodes d'évaluation à notre disposition.

Le principe de la vérification formelle de système d'IA est d'assurer que le système respectera toujours une propriété donnée, par rapport à une notion de perturbation des entrées. Cette notion de perturbation concerne en général les caractéristiques des entrées. On considère en effet qu'il est possible qu'une instance soit présentée au système avec des perturbations infinitésimales de ses caractéristiques, ce qui peut mener le système à se tromper dans sa prédiction, alors que pour un œil humain, l'instance n'a pas changé. La difficulté est alors de trouver l'ensemble des réponses que peut fournir le système selon un seuil précis de perturbation, car le nombre de possibilités différentes de perturbations de l'entrée est gigantesque. Dvijotham et. al. [Kri+18] parviennent à reformuler la question en un problème d'optimisation pouvant être traité analytiquement.

Dans notre cas, les perturbations que l'on aimerait considérer ne concernent pas les caractéristiques des instances, mais plutôt leurs annotations. Le problème serait donc de trouver l'ensemble des réponses possibles d'un système, selon des perturbations des annotations de son ensemble d'apprentissage. Cela semble constituer un défi encore plus grand que le précédent, car obtenir la réponse d'un système après perturbation nécessiterait de l'entraîner à nouveau. Des techniques d'approximation de la prédiction du système pourrait éventuellement être appliquées, par exemple en considérant un système de substitut rapide à entraîner et au comportement proche du système initial. Nous n'avons cependant pas connaissance de travaux de vérification formelle menée sous l'hypothèse de perturbations de cette forme. Cela constituerait donc une perspective de travail intéressante.

5.4 Conclusion

Nous avons présenté dans ce chapitre une synthèse des différentes pratiques d'évaluation envisageables pour répondre à la problématique du biais d'évaluation qu'implique la présence de bruit d'annotation dans un ensemble de test. La section 5.1 a abordé les principes et méthodes permettant de réduire le biais d'évaluation de manière préventive. La section 5.2 a présenté différentes pratiques d'évaluation moins affectées par le bruit d'annotation. Enfin, la section 5.3 a permis de montrer que des techniques d'évaluation complémentaires, fondées sur une approche plus rationnelle et attachées à la compréhension des mécanismes sous-jacent de raisonnement des système d'IA, se développent.

Néanmoins, les limitations de chacune de ces méthodes sont difficiles à ignorer. En particulier, plusieurs d'entre elles s'appuient sur des procédures de reconstruction automatique de l'annotation de l'ensemble de test, qui n'est pas assuré d'être de meilleur qualité que celle d'origine. D'autres méthodes impliquent des contraintes ou des hypothèses non triviales. C'est le cas de l'évaluation non supervisée de Raj [RSB11], qui nécessite l'indépendance entre les classifieurs évalués et le classifieur de référence. En ce qui concerne les méthodes d'explicabilité, bien qu'elles permettent d'évaluer de manière différente la qualité d'un système d'IA, la question secondaire qui se pose est l'impact du bruit d'annotation lorsqu'on les évalue à leur tour. Enfin, la vérification formelle des systèmes d'IA est encore peu développée, et difficile à adapter en l'état au problème du bruit d'annotation.

Si ces méthodes peuvent sur le principe être appliquées dans l'objectif de réduire le biais d'évaluation dû au bruit d'annotation, les limitations et hypothèses qu'elles impliquent peuvent

également être une cause de biais dans l'évaluation qu'elles produisent. Il n'y a donc pas de solution évidente et unique au problème du biais d'évaluation causé par le bruit d'annotation. La meilleure approche semble donc être d'appliquer de manière conjointe différentes méthodes d'évaluation, tout en respectant les bonnes pratiques mentionnées en section 5.1. En effet, à l'instar du principe des méthodes ensemblistes : combiner des méthodes affectées par différents biais peut les amener à se compenser. Bien que nous n'ayons pas connaissance de démonstration formelle de ce principe, il est fréquemment suivi en pratique. Dans le domaine de l'industrie aéronautique, par exemple, où la construction d'un avion s'accompagne d'innombrables procédures de test, l'adage suivant est souvent utilisé : « quelque chose vérifié par deux raisons indépendantes est encore plus vrai ». Dans ce domaine, l'utilisation conjointe de plusieurs méthodes de test indépendantes rend à terme l'évaluation plus fiable.

Chapitre 6

Dépendance formelle entre biais d'évaluation et bruit d'annotation

Sommaire

Le biais d'évaluation sur lequel nous nous interrogeons prend la forme du biais d'un estimateur précis, au sens mathématique du terme. Dans ce chapitre, de manière complémentaire aux travaux de Fedorchuk [FL17a ; FL17b] et Lam et al. [LS03], nous allons modéliser notre problématique dans le cadre de la théorie des estimateurs, ce qui nous permettra d'obtenir les expressions formelles du biais de plusieurs mesures usuelles, lorsque les exemples de test sont supposés bruités. Ces expressions nous serviront à obtenir les intervalles de confiance des mesures associées, de sorte à pouvoir tirer des conclusions sur la confiance que l'on peut avoir en l'estimation d'une mesure d'évaluation effectuée sur un ensemble d'instances contenant des erreurs d'annotation. Nous présenterons ensuite des expérimentations que nous avons réalisées pour comparer, en présence de bruit d'annotation, l'efficacité de différentes procédures d'évaluation.

6.1 Formulation du biais d'évaluation

Dans le but de modéliser le biais d'évaluation de l'estimateur d'une mesure, nous prenons l'exemple de la justesse d'un classifieur. Rappelons que la justesse correspond à la fraction d'instances classifiées correctement par le classifieur, i.e. la prédiction du classifieur coïncide avec l'annotation de l'instance. Nous étendons ensuite l'étude aux mesures de la précision, du rappel et de la f-mesure.

Notations

Pour cela, nous supposons que nous avons à disposition un ensemble d'entraînement $\mathcal{E}_A$ et un ensemble de test $\mathcal{E}_T$ pour entraîner et évaluer un classifieur $\mathcal{C}$. Ces ensembles contiennent des triplets $(\mathbf{x}, \mathbf{y}, \mathbf{c})$:

- $\mathbf{x}$ est un vecteur représentant les caractéristiques d'une instance ;
- $\mathbf{y}$ est une étiquette attribuée à cette instance, qui représente son interprétation (en général, par un humain) ;
- $\mathbf{c}$ est la véritable catégorie de l'instance, ou l'interprétation de l'instance ayant une valeur de vérité supérieure à l'étiquette ; $\mathbf{y}$ et $\mathbf{c}$ prennent valeur dans le même ensemble d'interprétation $\mathcal{I}$, mais ne sont pas forcément égales pour une même instance $\mathbf{x}$.

Dans ce qui suit, nous indexons ces triplets par l'indice $i \in [1, N]$, où $N = |\mathcal{E}_T|$ est la cardinalité de $\mathcal{E}_T$. Nous désignons indifféremment par $\mathbf{x}_i$, $\mathbf{x}$ ou simplement i une instance choisie arbitrairement.

6.1.1 Contexte

Nous nous plaçons dans le cadre où les seules informations qui nous sont accessibles sont les vecteurs de caractéristiques $\mathbf{x}_i$ et les annotations correspondantes $\mathbf{y}_i$. La véritable classe $\mathbf{c}_i$ d'une instance i est inconnue.

Les probabilités p et q

Après son apprentissage, $\mathcal{C}$ a une probabilité q de classifier de manière juste une nouvelle instance qui lui est présentée. C'est-à-dire que, pour une instance $\mathbf{x}_i$ choisie au hasard parmi l'ensemble de nos instances, q est la probabilité de prédire la classe $\mathbf{c}_i$ de $\mathbf{x}_i$. De plus, nous appelons p la probabilité que la prédiction de $\mathcal{C}$ pour $\mathbf{x}_i$ *soit en accord avec l'étiquette* $\mathbf{y}_i$.

La probabilité q correspond précisément à la justesse du classifieur, et notre objectif est de l'estimer à l'aide des instances de test à notre disposition.

6.1.2 Analyse

Considérons, pour toute instance i, la variable aléatoire de Bernoulli X_i prenant la valeur 1 avec une probabilité p, et 0 sinon. Lorsque $X_i = 1$, cela signifie que la réponse du classifieur $\mathcal{C}$ pour $\mathbf{x}_i$ est égale à l'étiquette $\mathbf{y}_i$. Sinon, cela veut dire que le classifieur a prédit une autre catégorie. L'espérance des variables aléatoires X_i est donc p.

L'hypothèse des annotations parfaites H

Nous notons H l'hypothèse que les annotations de nos instances soient toutes correctes, et qu'il n'y ait donc pas de bruit d'annotation. Cela signifie que pour toute instance i, $\mathbf{y}_i$ est égale à la véritable classe $\mathbf{c}_i$ de l'instance.

Supposons maintenant l'absence de bruit d'annotation. Dans ce cas, p correspond à la probabilité que $\mathcal{C}$ prédise effectivement la classe de l'instance qui lui est présentée, ce qui correspond à la définition de la justesse q. Sous l'hypothèse H, l'espérance des X_i vaut donc q.

Dans le cadre d'une procédure d'évaluation classique, l'estimateur utilisé pour la justesse d'un classifieur est la moyenne empirique suivante, aussi appelée estimateur de Monte-Carlo :

$$\hat{q} = \frac{1}{N} \sum_{i \in \mathcal{E}_T} X_i. \tag{6.1}$$

La somme est indexée par la taille de l'ensemble de test car seules les instances de test sont utilisées pour mesurer la justesse. La valeur que prend cette somme correspond au nombre d'instances de test pour lesquelles la prédiction de $\mathcal{C}$ est en accord avec l'étiquette, divisé par le nombre total d'instances N dans $\mathcal{E}_T$. L'espérance de cet estimateur vaut également p, identique à celle de chaque X_i. Par conséquent, sous l'hypothèse H où $p = q$, $\hat{q}$ est un estimateur non biaisé pour estimer la justesse.

Biais de l'estimateur de la justesse

Le biais de notre estimateur $\hat{q}$ est par définition la différence entre son espérance et la quantité que l'on veut estimer :

$$biais(\hat{q}) = \mathbb{E}(\hat{q}) - q \tag{6.2}$$

Sous l'hypothèse H, ce biais s'annule.

Supposons maintenant $\neg H$, i.e. la présence de bruit d'annotation. Dans ce cas, le biais de $\hat{q}$ devient non nul, et s'exprime de la manière suivante :

$$biais(\hat{q}) = \mathbb{P}(S) - \mathbb{P}(Q), \tag{6.3}$$

où S et Q sont des évènements de probabilités respectives p et q. En effet :

— S est l'évènement de Bernoulli des variables X_i, i.e. « *La prédiction du classifieur pour* $\mathbf{x}_i$ *est identique à* $\mathbf{y}_i$ » ;

— et Q est l'évènement « *La prédiction du classifieur pour* $\mathbf{x}_i$ *est identique à* $\mathbf{c}_i$ ».

A partir de cette modélisation, nous pouvons ensuite étudier les probabilités des évènements S et Q en fonction du bruit d'annotation, de sorte à établir un lien entre le biais de l'estimateur de la justesse et le bruit.

Cette modélisation n'est cependant pas restreinte à la mesure de la justesse. Dans le cas général, pour une mesure m empiriquement estimée à partir de $\mathcal{E}_T$ contenant du bruit d'annotation, la situation est la même. Nous distinguons deux estimations de m différentes :

1. l'estimation apparente m_{app}, i.e. celle qui peut être obtenue à l'aide des instances de test et de leur annotation ;

2. l'estimation correcte m_{cor}, celle que nous ne pouvons faire qu'en ayant accès aux véritables classes des instances de test.

Lorsque le classifieur prédit la catégorie d'une instance de test, sa prédiction peut accroître la valeur de l'une ou l'autre des estimations, ou des deux, ou bien d'aucune des deux. Par exemple, imaginons que nous disposons de 4 instances de test, dans un problème de classification à 2 classes a, b. Le tableau 6.1 illustre la situation.

Instance	Annotation	Classe	Prédiction
$\mathbf{x}_1$	a	a	a
$\mathbf{x}_2$	b	b	a
$\mathbf{x}_3$	a	b	a
$\mathbf{x}_4$	b	a	a

TABLE 6.1 – Exemple de 4 instances permettant d'illustrer les 4 situations possibles lors de l'estimation empirique d'une mesure à partir des prédictions de $\mathcal{C}$.

Lorsque le classifieur prédit a pour $\mathbf{x}_1$, il est en accord avec l'annotation et la classe de l'instance, sa prédiction contribue à accroître la valeur des deux estimations. Dans le cas de l'instance $\mathbf{x}_2$, sa prédiction est en désaccord avec l'annotation et la classe, elle n'accroît donc aucune des deux estimations. Pour l'instance $\mathbf{x}_3$, sa prédiction est en accord avec l'annotation de l'instance : cela augmente uniquement la valeur de l'estimation apparente. Enfin, dans le quatrième cas, sa prédiction est en accord avec la classe de l'instance, ce qui joue en faveur de l'estimation correcte.

Les évènements génériques B_m et E_m

Les évènements d'intérêt pour l'estimation apparente et l'estimation correcte sont donc respectivement :

- B_m, l'évènement correspondant à la prédiction du classifieur qui accroît la valeur de l'estimation *apparente* m_{app} ; dans le cas de la justesse, B_m correspond à l'évènement S ;
- E_m, l'évènement correspondant à la prédiction du classifieur qui accroît la valeur de l'estimation correcte m_{cor} ; la probabilité de cet évènement est directement égale à m.

Le biais de l'estimateur $\hat{m}$ est alors :

$$\begin{aligned} biais(\hat{m}) &= \mathbb{P}(B_m) - m = \mathbb{P}(B_m) - \mathbb{P}(E_m), \\ \text{où} \quad \mathbb{P}(E_m) &= m \end{aligned} \tag{6.4}$$

Les évènements B_m et E_m sont équivalents uniquement dans le cas où l'hypothèse H est vérifiée, i.e. il n'y a pas de bruit d'annotation.

6.2 Supposition de l'indépendance entre prédictions et bruit d'annotation

Pour obtenir l'expression du biais de l'estimateur d'une mesure m, il est nécessaire de définir B_m et E_m et de déterminer $\mathbb{P}(B_m)$. Pour cela, nous allons dans un premier temps supposer *l'indépendance entre les prédictions du classifieur et le bruit d'annotation*, et faire l'analyse pour

la mesure de la justesse. Mais avant cela, nous avons besoin de définir les notations de plusieurs évènements.

Notations des évènement d'intérêt

Les évènements suivants s'appliquent à un triplet quelconque $(\mathbf{x}, \mathbf{y}, \mathbf{c})$ de $\mathcal{E}_T$:
- S : « *La prédiction du classifieur pour* $\mathbf{x}$ *est identique à* $\mathbf{y}$ » ;
- Q : « *La prédiction du classifieur pour* $\mathbf{x}$ *est identique à* $\mathbf{c}$ » ;
- N : « *L'instance* $\mathbf{x}$ *est mal annotée, i.e.* $\mathbf{y} \neq \mathbf{c}$ » ;

De plus, pour toute catégorie k du problème de classification :
- P_k : « *La prédiction du classifieur pour* $\mathbf{x}$ *est la catégorie* k » ;
- C_k : « *La véritable classe de l'instance* $\mathbf{x}$ *est* k*, i.e.* $\mathbf{c} = i$ » ;
- A_k : « *L'annotation de l'instance* $\mathbf{x}$ *est* k*, i.e.* $\mathbf{y} = i$ ».

La catégorie k dans les définitions précédentes représente une catégorie arbitraire du problème de classification dont il est question. Par convention, si k n'est pas précisé, les évènements P, C et A concernent une catégorie quelconque, mais la même pour tous.

L'exemple de la justesse

Pour l'exemple courant, par souci de simplification, nous nous plaçons dans un cadre de classification binaire. Nous avons besoin de définir les évènements génériques B_m et E_m lorsque m correspond à la justesse q.

Définir l'évènement E_m revient à interpréter la mesure m de manière probabiliste. En ce qui concerne la justesse, elle peut être interprétée par la probabilité de l'évènement Q : $q = \mathbb{P}(Q)$. L'évènement E_q est donc Q. Ce genre d'interprétation probabiliste est possible pour d'autres mesures, et nous le verrons par la suite en ce qui concerne la prédiction et le rappel.

Nous avons de plus déjà défini l'évènement B_q dans ce qui précède. Il correspond à l'évènement S :

$$B_q = S = \text{« } \textit{La prédiction du classifieur pour } \mathbf{x} \textit{ est identique à } \mathbf{y} \text{ ».}$$

La probabilité de cet évènement peut être scindée en la somme suivante, à l'aide des évènements complémentaires N et $\overline{N}$:

$$\begin{aligned} \mathbb{P}(S) &= \mathbb{P}(S \cap N) + \mathbb{P}(S \cap \overline{N}) \\ &= \mathbb{P}(S \cap N) + \mathbb{P}(Q \cap \overline{N}) \end{aligned} \tag{6.5}$$

En classification binaire, étant donné qu'il n'y a que deux catégories possibles pour la classification des instances, les évènements S et $\overline{Q}$ sont équivalents. Cela permet la réécriture suivante :

$$\mathbb{P}(S) = \mathbb{P}(\overline{Q} \cap N) + \mathbb{P}(Q \cap \overline{N})$$

Dans [LS03], une modélisation similaire du problème est faite, et une hypothèse essentielle est posée à ce moment précis : les évènements Q et N sont indépendants. Cette hypothèse est justifiée par les auteurs en statuant qu'un classifieur fait ses prédictions indépendamment du fait qu'une instance est mal annotée ou non, lorsque les ensembles d'entraînement et de test sont disjoints.

Cette hypothèse permet de simplifier l'expression de la manière suivante :

$$\begin{aligned} \mathbb{P}(S) &= \mathbb{P}(\overline{Q})\mathbb{P}(N) + \mathbb{P}(Q)\mathbb{P}(\overline{N}) \\ &= (1 - \mathbb{P}(Q))\mathbb{P}(N) + \mathbb{P}(Q)(1 - \mathbb{P}(N)) \\ &= \mathbb{P}(Q) + \mathbb{P}(N)(1 - 2\mathbb{P}(Q)) \end{aligned} \tag{6.6}$$

La probabilité de l'évènement Q étant précisément la justesse q du classifieur, cela nous donne une expression pour le biais de l'estimateur $\hat{q}$ dans un cas de classification binaire :

$$\mathbb{P}(S) - \mathbb{P}(Q) = biais(\hat{q}) = \mathbb{P}(N)(1 - 2\mathbb{P}(Q)) \tag{6.7}$$

Il n'est pas très contraignant de supposer que $\mathbb{P}(Q) > 0.5$, car cela signifie que le classifieur possède au moins une meilleure justesse qu'un processus classifiant aléatoirement les instances du problème. Dans ce cas, $biais(\hat{q})$ est compris dans l'intervalle $[-\mathbb{P}(N), 0]$. Il est toujours négatif, ce qui implique que l'estimation de la justesse obtenue via l'estimateur $\hat{q}$ est une sous-estimation de sa véritable valeur. Ce résultat est également obtenu dans [LS03], dans un contexte de formalisation similaire.

Nous pouvons cependant conclure ceci uniquement dans un contexte de classification binaire, et en acceptant que les prédictions du classifieur soient indépendantes du bruit d'annotation présent dans les données de test.

6.3 Supposition de la dépendance entre prédictions et bruit d'annotation

Le résultat précédent n'est valable que si l'hypothèse de l'indépendance entre les évènements Q et N est vérifiée. Cette hypothèse n'est cependant acceptable que dans peu de cas pratiques. En effet, prenons un ensemble d'instances contenant du bruit d'annotation et considérons un exemple de test bruité $(\mathbf{x}, \mathbf{y}, \mathbf{c})$. Sous un modèle de bruit d'annotation par classe ou par caractéristiques (cf. chapitre 3), le fait qu'un annotateur ait mal annoté l'instance dépend de sa classe $\mathbf{c}$ ou de son vecteur de caractéristiques $\mathbf{x}$. Si l'on considère les instances similaires à notre exemple de test, et se trouvant dans l'ensemble d'entraînement, il y a de bonnes chances qu'elles soient également mal annotées, pour les mêmes raisons, i.e. leur classe ou leurs caractéristiques. Cela signifie que le classifieur, ayant été entraîné sur ces instances précises, risque de reproduire les mêmes erreurs d'annotation que les annotateurs au moment de prédire la catégorie de l'instance de test $\mathbf{x}$.

Une situation où l'hypothèse d'indépendance entre les évènements Q et N est supposée par défaut est le cas du modèle de bruit uniforme (cf. section 3.2.2). Ce modèle considère entre autres que le bruit d'annotation ne dépend pas des caractéristiques des instances, seules informations déterminant les prédictions du classifieur. Cependant, le modèle de bruit uniforme est connu pour être souvent trop simpliste en pratique [FV14]. Dans cette partie, nous allons obtenir des expressions formelles pour les biais des estimateurs de la justesse, la précision, le rappel et la f-mesure, sans supposer l'indépendance entre les évènements Q et N, et dans un cadre de classification multi-classes.

6.3.1 Biais de l'estimateur de la justesse

Dans le cas de la justesse, et en reprenant les notations de la section 6.2, nous avons besoin d'introduire trois indicateurs particuliers. Ces derniers concernent différents aspects de la façon dont le classifieur fait ses prédictions, relativement au bruit d'annotation.

Les indicateurs F_c, F_n et F_r

Les trois indicateurs dont nous nous servons pour caractériser le comportement d'un classifieur par rapport au bruit d'annotation sont :

— $F_c = \mathbb{P}(Q|\overline{N})$: cette probabilité représente le taux d'apprentissage de la vraie tâche

de classification, définie par le sous-ensemble des instances bien annotées ;
— $F_n = \mathbb{P}(S|N)$: représente le taux d'apprentissage du biais d'interprétation des annotateurs, défini par le sous-ensemble des instances mal annotées ;
— $F_r = \mathbb{P}(Q|N)$: représente la tendance du classifieur à privilégier la connaissance acquise par l'apprentissage de la vraie tâche de classification lors de la prédiction de la catégorie d'une instance mal annotée.

On peut ainsi formuler $\mathbb{P}(S)$ et $\mathbb{P}(Q)$ de la façon suivante :

$$\begin{aligned} \mathbb{P}(S) &= \mathbb{P}(S \cap N) + \mathbb{P}(S \cap \overline{N}) \\ &= \mathbb{P}(S|N)\mathbb{P}(N) + \mathbb{P}(S|\overline{N})\mathbb{P}(\overline{N}) \\ \mathbb{P}(Q) &= \mathbb{P}(Q \cap N) + \mathbb{P}(Q \cap \overline{N}) \\ &= \mathbb{P}(Q|N)\mathbb{P}(N) + \mathbb{P}(Q|\overline{N})\mathbb{P}(\overline{N}) \end{aligned} \tag{6.8}$$

Sachant que l'absence de bruit d'annotation $\overline{N}$ implique l'équivalence entre les évènements S et Q, nous pouvons dire que $\mathbb{P}(S|\overline{N}) = \mathbb{P}(Q|\overline{N})$.

Biais de l'estimateur de la justesse

On obtient donc l'expression du biais de l'estimateur de la justesse en fonction des indicateurs F_n et F_r :

$$\begin{aligned} biais(\hat{q}) = \mathbb{P}(S) - \mathbb{P}(Q) &= \mathbb{P}(S|N)\mathbb{P}(N) - \mathbb{P}(Q|N)\mathbb{P}(N) \\ &= \mathbb{P}(N)(F_n - F_r) \end{aligned} \tag{6.9}$$

Dans le cas particulier de la classification binaire, $F_r = 1 - F_n$, ce qui donne :

$$biais(\hat{q}) = \mathbb{P}(N)(2F_n - 1) \tag{6.10}$$

On en conclut que :
— si $F_r = F_n$: le biais est nul ; dans le cas binaire, $F_n = 0.5$, ce qui signifie que le classifieur prédit de manière aléatoire la catégorie d'une instance mal annotée ;
— si $F_r = 1$ (donc $F_n = 0$) : le biais est égal à $-\mathbb{P}(N)$; le classifieur est sous-estimé alors qu'il est parfaitement robuste au bruit d'annotation ;
— si $F_n = 1$ (donc $F_r = 0$) : le biais est égal à $\mathbb{P}(N)$; le classifieur est surestimé alors qu'il a complètement assimilé le biais d'interprétation des annotateurs.

En pratique, les trois indicateurs F_c, F_n et F_r ne sont pas accessibles, car cela reviendrait à connaître précisément quelles instances sont mal annotées. Leur utilité est ici d'ordre théorique, car ils permettent de faire le lien entre certains aspects de la performance du classifieur, e.g. à quel point ce dernier reproduit les erreurs d'annotation des annotateurs, et le biais de l'estimateur de sa justesse.

6.3.2 Biais des estimateurs de la précision et du rappel

Pour modéliser dans un cadre probabiliste les mesures de précision et de rappel par rapport à une catégorie k, ainsi que le biais des estimateurs correspondant, il est nécessaire de faire intervenir la notion de probabilité conditionnelle.

Interprétation probabiliste de la précision et du rappel

Fedorchuk et. al. [FL17b] propose en effet l'interprétation probabiliste suivante pour la précision p_r et le rappel r_p par rapport à une catégorie arbitraire k :

- $p_r = \mathbb{P}(C_k|P_k)$, la probabilité qu'une instance $\mathbf{x}$ de la classe k soit classifiée dans la catégorie k ;
- $r_p = \mathbb{P}(P_k|C_k)$, la probabilité qu'une instance $\mathbf{x}$ classifiée dans la catégorie k soit effectivement de la classe k.

Dans l'analyse que nous avons effectué en section 6.1.2, nous avons exprimé le biais d'une mesure m en fonction des évènements génériques B_m et E_m (cf. formule 6.4). Nous gardons ici le même état d'esprit, mais nous étendons la formalisation pour inclure la notion de probabilité conditionnelle.

Nous voulons donc des couples d'évènements (B_m^1, B_m^2) et (E_m^1, E_m^2), de sorte que le biais de $\hat{m}$ soit :

$$\begin{aligned} biais(\hat{m}) &= \mathbb{P}(B_m^1|B_m^2) - \mathbb{P}(E_m^1|E_m^2) \\ \text{où} \quad \mathbb{P}(E_m^1|E_m^2) &= m \end{aligned} \tag{6.11}$$

Dans ce contexte, le couple $(E_m^1|E_m^2)$ devient évident pour les mesures de précision et de rappel, résultant directement de leur interprétation probabiliste :

$$\begin{aligned} E_{p_r}^1 &= C_k \\ E_{p_r}^2 &= P_k \\ \\ \text{et} \quad E_{r_p}^1 &= P_k \\ E_{r_p}^2 &= C_k \end{aligned} \tag{6.12}$$

Les évènements B_m^1 et B_m^2 font intervenir l'annotation des instances au lieu de leur véritable classe. Ainsi, pour les obtenir, il suffit de considérer le couple (E_m^1, E_m^2), et de remplacer l'évènement C_k par A_k, en gardant le reste identique :

$$\begin{aligned} B_{p_r}^1 &= A_k \\ B_{p_r}^2 &= P_k \\ \\ \text{et} \quad B_{r_p}^1 &= P_k \\ B_{r_p}^2 &= A_k \end{aligned} \tag{6.13}$$

Les biais des estimateurs de la précision et du rappel sont donc :

$$\begin{aligned} biais(\hat{p_r}) &= \mathbb{P}(A|P) - \mathbb{P}(C|P) \\ biais(\hat{r_p}) &= \mathbb{P}(P|A) - \mathbb{P}(P|C) \end{aligned} \tag{6.14}$$

Les évènements A, C et P concernent tous la même catégorie k, elle n'est donc plus précisée à partir de maintenant pour ne pas alourdir les notations.

Dans le cas de la précision, l'expression peut se condenser de la manière suivante :

$$\begin{aligned}
\mathbb{P}(A|P) &= \mathbb{P}(A \cap N|P) + \mathbb{P}(A \cap \overline{N}|P) \\
\mathbb{P}(C|P) &= \mathbb{P}(C \cap N|P) + \mathbb{P}(C \cap \overline{N}|P) \\
&= \mathbb{P}(C \cap N|P) + \mathbb{P}(A \cap \overline{N}|P) \\
\mathbb{P}(A|P) - \mathbb{P}(C|P) &= \mathbb{P}(A \cap N|P) - \mathbb{P}(C \cap N|P) \\
&= \frac{1}{\mathbb{P}(P)}(\mathbb{P}(N \cap A)\mathbb{P}(P|A,N) - \mathbb{P}(N \cap C)\mathbb{P}(P|N,C))
\end{aligned} \tag{6.15}$$

Le calcul pour le rappel est moins simple :

$$\begin{aligned}
\mathbb{P}(P|A) &= \mathbb{P}(P \cap N|A) + \mathbb{P}(P \cap \overline{N}|A) \\
&= \mathbb{P}(N|A)\mathbb{P}(P|N,A) + \mathbb{P}(\overline{N}|A)\mathbb{P}(P|\overline{N},A) \\
&= (1 - \mathbb{P}(C|A))\mathbb{P}(P|N,A) + \mathbb{P}(C|A)\mathbb{P}(P|\overline{N},C) \\
\mathbb{P}(P|C) &= \mathbb{P}(P \cap N|C) + \mathbb{P}(P \cap \overline{N}|C) \\
&= \mathbb{P}(N|C)\mathbb{P}(P|N,C) + \mathbb{P}(\overline{N}|C)\mathbb{P}(P|\overline{N},C) \\
&= (1 - \mathbb{P}(A|C))\mathbb{P}(P|N,C) + \mathbb{P}(A|C)\mathbb{P}(P|\overline{N},C) \\
\mathbb{P}(P|A) - \mathbb{P}(P|C) &= \mathbb{P}(P|N,A) - \mathbb{P}(P|N,C) \\
&\quad - \mathbb{P}(A \cap C)(\frac{\mathbb{P}(P|N,A)}{\mathbb{P}(A)} - \frac{\mathbb{P}(P|N,C)}{\mathbb{P}(C)}) \\
&\quad + \mathbb{P}(A \cap C)\mathbb{P}(P|\overline{N},C)(\frac{1}{\mathbb{P}(A)} - \frac{1}{\mathbb{P}(C)})
\end{aligned} \tag{6.16}$$

Les indicateurs F_c, F_n et F_r peuvent également s'exprimer par rapport à une catégorie k donnée :

- $F_c^k = \mathbb{P}(P_k|\overline{N}, C_k)$: la probabilité de classifier dans la catégorie k une instance de la classe k bien annotée ;
- $F_n^k = \mathbb{P}(P_k|N, A_k)$: la probabilité de classifier dans la catégorie k une instance étiquetée en tant que k et mal annotée ;
- $F_r^k = \mathbb{P}(P_k|N, C_k)$: la probabilité de classifier dans la catégorie k une instance de la classe k mal annotée.

À l'exception des trois indicateurs précédents, les autres termes peuvent être estimés en situation réelle à partir de l'ensemble $\mathcal{E}_T$, ou à condition d'avoir déterminé la matrice de transition de bruit :

- $\mathbb{P}(A)$ s'estime par la fraction d'instances étiquetées en tant que i dans $\mathcal{E}_T$;
- $\mathbb{P}(P)$ s'estime par la fraction d'instances classifiées en tant que i dans $\mathcal{E}_T$;
- $\mathbb{P}(N \cap A)$, $\mathbb{P}(N \cap C)$, $\mathbb{P}(A \cap C)$ ou $\mathbb{P}(C)$ s'estiment à l'aide de la matrice de transition de bruit.

Biais des estimateurs de la précision et du rappel

Nous obtenons donc les expressions suivantes du biais des estimateurs de la précision et du rappel :

$$
\begin{aligned}
biais(\hat{p_r}) &= \mathbb{P}(A|P) - \mathbb{P}(C|P) = \frac{1}{\mathbb{P}(P)}(\mathbb{P}(N \cap A)F_n^k - \mathbb{P}(N \cap C)F_r^k) \\
biais(\hat{r_p}) &= \mathbb{P}(P|A) - \mathbb{P}(P|C) = F_n^k - F_r^k \\
&\quad - \mathbb{P}(A \cap C)(\frac{F_n^k}{\mathbb{P}(A)} - \frac{F_r^k}{\mathbb{P}(C)}) \\
&\quad + \mathbb{P}(A \cap C)F_c^k(\frac{1}{\mathbb{P}(A)} - \frac{1}{\mathbb{P}(C)})
\end{aligned}
\tag{6.17}
$$

La catégorie k n'est précisée que pour les indicateurs F_c^k, F_n^k et F_r^k, mais les évènements A, P et C sont aussi indexés par k, bien que nous ayons omis cette notation pour des raisons de visibilité.

Analyse sous des hypothèses d'indépendance supplémentaires

Que se passe-t-il pour le biais des estimateurs de la précision et du rappel dans le cas d'un modèle de bruit uniforme et de l'hypothèse d'indépendance entre les prédictions du classifieur et le bruit d'annotation ? Les relations d'indépendance à prendre en compte sont alors :

$$
\begin{aligned}
N \perp\!\!\!\perp C &\quad : \quad \text{modèle de bruit uniforme} \\
N \perp\!\!\!\perp P &\quad : \quad \text{indépendance entre prédictions et bruit d'annotation}
\end{aligned}
\tag{6.18}
$$

Dans ce contexte, les indicateurs F_c^k, F_n^k et F_r^k se simplifient :

$$
\begin{aligned}
F_n^k &= \mathbb{P}(P|N, A) = \mathbb{P}(P|A) \\
F_c^k &= \mathbb{P}(P|\overline{N}, C) = \mathbb{P}(P|C) \\
F_r^k = \mathbb{P}(P|N, C) &= \mathbb{P}(P|C) = F_c^k
\end{aligned}
\tag{6.19}
$$

Le biais de l'estimateur de la précision peut se réécrire de la façon suivante :

$$
\begin{aligned}
biais(\hat{p_r}) &= \mathbb{P}(A|P) - \mathbb{P}(C|P) \\
&= \frac{1}{\mathbb{P}(P)}(\mathbb{P}(N \cap A)\mathbb{P}(P|A, N) - \mathbb{P}(N \cap C)\mathbb{P}(P|N, C)) \\
&= \mathbb{P}(N|A) \quad \frac{\mathbb{P}(A)\mathbb{P}(P|A)}{\mathbb{P}(P)} \quad - \quad \frac{\mathbb{P}(N)\mathbb{P}(C)\mathbb{P}(P|C)}{\mathbb{P}(P)} \\
\mathbb{P}(A|P) - \mathbb{P}(C|P) &= \mathbb{P}(N|A)\mathbb{P}(A|P) - \mathbb{P}(N)\mathbb{P}(C|P)
\end{aligned}
\tag{6.20}
$$

En remaniant les membres de gauche et de droite, et en admettant que $\mathbb{P}(N) \neq 1$, nous pouvons obtenir l'expression plus condensée suivante :

$$
biais(\hat{p_r}) = \frac{\mathbb{P}(N|A) - \mathbb{P}(N)}{1 - \mathbb{P}(N)}\mathbb{P}(A|P) \tag{6.21}
$$

Maintenant, pour le biais de l'estimateur du rappel, le résultat est plus surprenant. En effet, supposons les deux relations d'indépendance de 6.18. Cela nous donne :

$$\begin{aligned}\mathbb{P}(P|A) - \mathbb{P}(P|C) &= \mathbb{P}(P|N,A) - \mathbb{P}(P|N,C) \\ &\quad - \mathbb{P}(A \cap C)\left(\frac{\mathbb{P}(P|N,A)}{\mathbb{P}(A)} - \frac{\mathbb{P}(P|N,C)}{\mathbb{P}(C)}\right) \\ &\quad + \mathbb{P}(A \cap C)\mathbb{P}(P|\overline{N},C)\left(\frac{1}{\mathbb{P}(A)} - \frac{1}{\mathbb{P}(C)}\right) \\ &= \mathbb{P}(P|A) - \mathbb{P}(P|C) \\ &\quad - \mathbb{P}(C|A)\mathbb{P}(P|A) + \mathbb{P}(A|C)\mathbb{P}(P|C) \\ &\quad + \mathbb{P}(C|A)\mathbb{P}(P|C) - \mathbb{P}(A|C)\mathbb{P}(P|C)\end{aligned} \tag{6.22}$$

En simplifiant, nous obtenons donc que :

$$\begin{aligned}(\mathbb{P}(P|A) - \mathbb{P}(P|C))\mathbb{P}(C|A) &= 0 \\ biais(\hat{r_p})\mathbb{P}(C|A) &= 0\end{aligned} \tag{6.23}$$

Cela signifie que si $\mathbb{P}(C|A) \neq 0$, i.e. il existe des instances étiquetées en tant que i qui appartiennent à la classe i, le biais de l'estimateur du rappel est nul, non impacté par le bruit d'annotation éventuellement présent. Bien sûr, ce résultat suppose les relations d'indépendance de 6.18 que nous avons énoncées plus haut, mais si la situation permet de les affirmer, mesurer la performance des modèles par leur rappel sur chaque classe permettrait de ne pas être biaisé par les instances bruitées.

Dans quelles situations ces hypothèses seraient-elles acceptables ? Nous avons déjà mentionné que l'hypothèse $N \perp\!\!\!\perp C$, qui revient à supposer un bruit d'annotation uniforme, est en pratique peu crédible. Elle reste néanmoins adaptée dans le cas précis où les instances et leurs annotations ont été d'abord transmises par un signal électronique bruité, avec des chances d'erreurs de transmission.

De la même manière, l'hypothèse $N \perp\!\!\!\perp P$ nous semble plutôt cavalière, car la présence d'un bruit d'annotation, même uniforme, impacte négativement la description du concept, et donc les prédictions réalisées par le classifieur. L'utilisation de cette hypothèse pourrait malgré tout être justifiée dans le cas où le classifieur apprendrait de façon non supervisée, mais nous ne nous sommes pas concentré sur cet aspect dans nos réflexions.

Nous avons donc obtenu des expressions pour le biais des estimateurs de précision et de rappel, que nous utiliserons en pratique dans les expérimentations présentées en section 6.5. Nous allons avant cela nous servir de ces expressions pour obtenir la valeur du biais de l'estimateur de la F-mesure, mesure dérivée à partir de la précision et du rappel.

6.3.3 Biais de l'estimateur de la F-mesure

La F-mesure f_m est un indicateur souvent utilisé dans les cas où la précision et le rappel doivent tout deux être suffisamment grands. D'après les interprétation probabiliste de la précision p_r et du rappel r_p (cf. section 6.3.2), nous obtenons l'expression suivante :

$$f_m = \frac{2p_r r_p}{p_r + r_p} = \frac{2\mathbb{P}(P|C)\mathbb{P}(C|P)}{\mathbb{P}(P|C) + \mathbb{P}(C|P)} \tag{6.24}$$

L'expression du biais de son estimateur $\hat{f_m}$ peut s'obtenir à partir du biais des estimateurs $\hat{p_r}$ et $\hat{r_p}$ obtenu en section 6.3.2 :

$$
\begin{aligned}
biais(\hat{f_m}) &= \mathbb{E}(\hat{f_m}) - f_m \\
&= \frac{2\mathbb{P}(P|A)\mathbb{P}(A|P)}{\mathbb{P}(P|A)+\mathbb{P}(A|P)} - \frac{2\mathbb{P}(P|C)\mathbb{P}(C|P)}{\mathbb{P}(P|C)+\mathbb{P}(C|P)} \\
&= \frac{2}{1+\frac{\mathbb{P}(A)}{\mathbb{P}(P)}}\mathbb{P}(A|P) - \frac{2}{1+\frac{\mathbb{P}(C)}{\mathbb{P}(P)}}\mathbb{P}(C|P) \\
&= \frac{2\mathbb{P}(A|P)(1+\frac{\mathbb{P}(C)}{\mathbb{P}(P)}) - 2\mathbb{P}(C|P)(1+\frac{\mathbb{P}(A)}{\mathbb{P}(P)})}{(1+\frac{\mathbb{P}(A)}{\mathbb{P}(P)})(1+\frac{\mathbb{P}(C)}{\mathbb{P}(P)})} \\
&= \frac{2(\mathbb{P}(A|P)-\mathbb{P}(C|P)) + \frac{2}{\mathbb{P}(P)}(\mathbb{P}(A|P)\mathbb{P}(C) - \mathbb{P}(C|P)\mathbb{P}(A))}{(1+\frac{\mathbb{P}(A)}{\mathbb{P}(P)})(1+\frac{\mathbb{P}(C)}{\mathbb{P}(P)})} \\
&= \frac{2(\mathbb{P}(A|P)-\mathbb{P}(C|P)) + \frac{2}{\mathbb{P}(P)}(\mathbb{P}(P|A)\frac{\mathbb{P}(A)\mathbb{P}(C)}{\mathbb{P}(P)} - \mathbb{P}(P|C)\frac{\mathbb{P}(A)\mathbb{P}(C)}{\mathbb{P}(P)})}{(1+\frac{\mathbb{P}(A)}{\mathbb{P}(P)})(1+\frac{\mathbb{P}(C)}{\mathbb{P}(P)})} \\
biais(\hat{f_m}) &= \frac{2(biais(\hat{p_r}) + \frac{\mathbb{P}(A)\mathbb{P}(C)}{\mathbb{P}(P)^2}biais(\hat{r_p}))}{(1+\frac{\mathbb{P}(A)}{\mathbb{P}(P)})(1+\frac{\mathbb{P}(C)}{\mathbb{P}(P)})}
\end{aligned}
\tag{6.25}
$$

Les termes $\mathbb{P}(A)$ et $\mathbb{P}(P)$ peuvent être obtenus simplement à partir des prédictions du classifieur et des annotations. Le terme $\mathbb{P}(C)$, quant à lui, peut être estimé à condition d'avoir accès à la matrice de transition de bruit.

En plus de l'intérêt théorique de comprendre formellement comment le biais d'un estimateur se comporte, cette expression a une utilité en pratique, tout comme celles des biais précédents. En effet, elle permet de calculer un encadrement du biais de l'estimateur de la mesure, ce qui peut servir à mener l'évaluation de classifieur de façon plus prudente. Nous montrons cet aspect dans un cadre expérimental en section 6.5. Bien que nous n'abordons dans ce cadre que les mesures de justesse, et de précision et rappel par rapport à chaque classe, le principe est identique en ce qui concerne la F-mesure.

6.4 Intervalle de confiance corrigé

Maintenant que nous avons formalisé l'impact du bruit d'annotation sur la procédure de test en terme de biais des estimateurs de performance, nous allons nous intéresser à ce que cela implique pour la confiance par rapport aux estimations de performance effectuées. Une manière de quantifier cette confiance est d'utiliser des intervalles de confiance (cf. chapitre 2). Nous rappelons qu'un intervalle de confiance est un intervalle pour lequel on peut affirmer qu'il contient la mesure que l'on tente d'estimer, avec un risque de se tromper connu et minime. Le calcul des bornes de cet intervalle dépend habituellement de la quantité d'instances de test utilisées pour estimer la mesure, ainsi que de l'estimation obtenue. Par exemple, pour une estimation de la justesse égale à 80% et obtenue avec 100 instances de test, l'intervalle de confiance est centrée autour de 80%, et sa largeur est plus grande que si la même estimation avait été obtenue avec 1000 instances de test.

Cependant, cela est valable seulement si l'estimation obtenue est supposée non biaisée. Dans le cas contraire, la valeur du biais a-t-elle une influence sur la confiance que l'on peut avoir en

l'estimation faite ? Dans ce qui suit, nous allons nous placer dans le cadre où nous disposons d'une estimation, obtenue à partir d'un estimateur avec un biais non nul, et nous allons déterminer l'expression des bornes d'un intervalle de confiance pour cette estimation.

6.4.1 Formalisation

Pour une variable normale X de moyenne nulle et de variance unitaire, nous savons qu'il est aisé d'obtenir, pour chaque valeur de z, la probabilité c correspondante telle que :

$$\mathbb{P}(-z < X < z) = c.$$

Soit une mesure m. Par définition du biais, l'espérance de l'estimateur $\hat{m}$ vaut $m + biais(\hat{m})$. Sous l'hypothèse que $\hat{m}$ s'exprime comme la moyenne de la somme de variables aléatoires de Bernoulli B_i, nous avons que :

$$\begin{aligned} \hat{m} &= \frac{1}{N_{\hat{m}}} \sum_{i=1}^{N_{\hat{m}}} B_i, \\ \mathbb{E}(\hat{m}) &= m + biais(\hat{m}), \\ \mathbb{V}(\hat{m}) &= \frac{\mathbb{E}(\hat{m})(1 - \mathbb{E}(\hat{m}))}{N_{\hat{m}}} \end{aligned} \tag{6.26}$$

Les estimateurs de la justesse, de la précision et du rappel peuvent être exprimés de cette manière :

— pour la justesse, $N_{\hat{q}}$ est la quantité totale d'instances de test, et l'essai de Bernoulli est un succès lorsque le classifieur prédit la catégorie correspondant à l'annotation $\mathbf{y}$ de l'instance $\mathbf{x}$;
— pour la précision par rapport à la catégorie i, $N_{\hat{p_r}}$ est la quantité d'instances de test que le classifieur a prédit dans la catégorie i, et l'essai de Bernoulli est un succès lorsque l'annotation $\mathbf{y}$ de l'instance $\mathbf{x}$ est effectivement i ;
— pour le rappel par rapport à la catégorie i, $N_{\hat{r_p}}$ est la quantité d'instances de test pour lesquelles $\mathbf{y} = i$, et l'essai de Bernoulli est un succès lorsque le classifieur prédit la catégorie i.

Nous pouvons ainsi centrer et réduire $\hat{m}$ avant d'appliquer la relation précédente :

$$\mathbb{P}(-z < \frac{\hat{m} - \mathbb{E}(\hat{m})}{\sqrt{\mathbb{V}(\hat{m})}} < z) = c \tag{6.27}$$

Pour une estimation e obtenue via $\hat{m}$, et en notant β le biais de $\hat{m}$, nous avons :

$$-z < \frac{e - m - \beta}{\sqrt{\frac{(m+\beta)(1-m-\beta)}{N_{\hat{m}}}}} < z \tag{6.28}$$

En passant à la valeur absolue, puis en élevant au carré la relation 6.28, on peut multiplier des deux côtés par le dénominateur, simplifier et regrouper par puissance de m, pour obtenir :

$$\begin{aligned} &m^2 + \Big(2\beta - \frac{\frac{z^2}{N_{\hat{m}}} + 2e}{1 + \frac{z^2}{N_{\hat{m}}}}\Big)m \\ &+ \Big(\beta^2 - \frac{\frac{z^2}{N_{\hat{m}}} + 2e}{1 + \frac{z^2}{N_{\hat{m}}}}\beta + \frac{e^2}{1 + \frac{z^2}{N_{\hat{m}}}}\Big) = 0 \end{aligned} \tag{6.29}$$

Expression des bornes d'un intervalle de confiance pour un estimateur biaisé

Résoudre l'équation précédente donne les solutions suivantes, correspondant aux deux bornes de l'intervalle de confiance :

$$-\beta + \frac{e + \frac{z^2}{2N_{\hat{m}}} \pm z\sqrt{\frac{e}{N_{\hat{m}}} - \frac{e^2}{N_{\hat{m}}} + \frac{z^2}{4N_{\hat{m}}^2}}}{1 + \frac{z^2}{N_{\hat{m}}}} \tag{6.30}$$

Cette expression correspond quasiment en tout point à celle des bornes d'un intervalle de confiance calculé sans prendre en compte le biais dû au bruit d'annotation dans l'ensemble de test (cf. chapitre 2). La seule différence est qu'on a retranché aux deux bornes la valeur β de ce biais.

6.4.2 Analyse

Intuitivement, on pourrait penser que la présence d'un bruit d'annotation devrait augmenter l'incertitude des estimations obtenues. Si l'on considère que la largeur de l'intervalle de confiance représente cette incertitude, cette intuition se révèle fausse : le bruit d'annotation n'a pas d'impact direct sur la taille d'un intervalle de confiance. Cette taille est uniquement déterminée par la variance de l'estimateur utilisé, ainsi que le risque alpha considéré. En effet, une petite variance et/ou un gros risque alpha mène à un petit intervalle de confiance, tandis qu'une grande variance et/ou un petit risque alpha mène à un grand intervalle de confiance.

La variance d'une somme de variables de Bernoulli i.i.d. dépend de l'espérance de ces variables, ainsi que du nombre de variables sommées. Mathématiquement, la variance peut donc être haute pour deux raisons :

- (i) le nombre de variables sommées, i.e. le nombre d'instances de test, est faible ;
- (ii) l'espérance de l'estimateur est proche de 0.5.

Or, comme nous l'avons précédemment vu (cf. section 6.3.1), le bruit d'annotation peut avoir les deux impacts suivants, selon la situation :

- (1) lorsqu'il implique un biais négatif,
  - l'estimateur biaisé voit son espérance diminuer par rapport à celle de l'estimateur idéal, i.e. l'espérance se rapproche de 0.5 [14] ;
  - selon (ii), sa variance *augmente* ;
  - par conséquent, les mesures calculées sous-estiment la performance réelle du classifieur d'une part, et d'autre part, l'incertitude autour des estimations est plus grande qu'elle ne le devrait ;
- (2) lorsqu'il implique un biais positif,
  - l'estimateur biaisé voit son espérance s'accroître par rapport à celle de l'estimateur idéal, i.e. l'espérance s'éloigne de 0.5 ;
  - selon (ii), sa variance *diminue* ;
  - par conséquent, non seulement les mesures calculées surestiment la performance réelle du classifieur, mais il y a aussi moins d'incertitude autour des estimations faites, quand bien même ces estimations sont en réalité perturbées par la présence de bruit d'annotation ; ;

14. Nous supposons que, dans le cas général, un classifieur entraîné atteint une performance supérieure à 0.5.

Quantifier l'incertitude d'une estimation par un intervalle de confiance peut donc être mis en défaut par la mauvaise qualité de l'annotation des instances. Par exemple, l'utilisation d'un intervalle de confiance suppose, entre autres, que l'augmentation du nombre de données de test permet d'améliorer la qualité de l'estimation, car cela diminuent la variance de l'estimateur. En général, cette supposition est valable. En effet, lorsqu'on augmente la taille de l'ensemble de test, cela implique nécessairement que l'ensemble des instances de test se rapprochent de l'ensemble des instances du problème. Les estimations ainsi effectuées reflètent mieux la performance réelle du classifieur. Cependant, la situation est plus difficile à contrôler lorsqu'on est en présence de bruit d'annotation.

Illustrons cela par le cas artificiel suivant. Supposons que l'on dispose :

— d'un classifieur de justesse réelle égale à 80%,
— de 1000 instances de test,
— contenant 20% de bruit d'annotation, i.e. 200 instances sont mal annotées.

Parmi ces 200 mauvaises instances :

— le classifieur parvient à prédire la classe réelle pour 50 d'entre elles,
— et prédit pour chacune des 150 autres une catégorie qui n'est ni la classe réelle, ni l'étiquette lui étant attribuée.

Pour ce classifieur, nous pouvons donc calculer les valeurs des indicateurs F_n et F_r, ainsi que le biais de l'estimateur de la justesse, à l'aide de l'expression 6.9 :

$$\begin{aligned} F_n &= 0 \\ F_r &= 0.25 \\ biais(\hat{q}) &= 0.2 * (0 - 0.25) = -0.05 \end{aligned} \tag{6.31}$$

Une valeur négative pour le biais signifie donc que l'incertitude d'une estimation de la justesse est plus haute que dans le cas où les instances seraient parfaitement annotées, en accord avec la situation (1) décrite plus haut. Cela n'a rien de dérangeant : la présence d'un bruit d'annotation rend plus incertaine l'estimation des mesures de performance.

Supposons maintenant que l'on double la taille de notre ensemble de test, en ajoutant 1000 nouvelles instances, pour améliorer la précision de notre estimation. Parmi ces nouvelles instances :

— 200 sont mal annotées ;
— les prédictions du classifieur sur ces 200 instances correspondent toujours à leurs étiquettes.

On obtient donc :

$$\begin{aligned} F_n &= \frac{200}{400} = 0.5 \\ F_r &= \frac{50}{400} = 0.125 \\ biais(\hat{q}) &= 0.2(0.5 - 0.125) = 0.075 \end{aligned} \tag{6.32}$$

Cette fois, le biais est positif. Selon la situation (2) décrite précédemment, l'incertitude au sujet des estimations faites est ainsi réduite, du fait de la présence de bruit d'annotation. Sur le principe, cela est contraire à l'intuition que l'on pourrait avoir sur la question. Ainsi, si en ajoutant des instances de test supplémentaires, nous faisions empirer la situation, i.e. le biais se rapprocherait de sa valeur maximale 0.2, cela aurait pour conséquence de diminuer d'autant

plus la variance de notre estimateur et donc de resserrer les intervalles de confiance autour d'une mesure biaisée de la justesse, proche de 82%, là où la justesse réelle est, rappelons-le, de 80%.

Jusqu'ici, nous avons donc obtenu les expressions formelles du biais des estimateurs de la justesse, de la précision, du rappel et de la f-mesure, lorsque l'ensemble de test contient des instances mal annotées. Ces expressions ont été obtenues dans un cadre général, sans avoir supposé de relations d'indépendance particulières, ou autre hypothèse simplificatrice. Nous avons de plus étudié la conséquence de tels biais sur la notion d'intervalle de confiance, et nous avons obtenu le résultat contre-intuitif que la présence d'un bruit d'annotation n'augmente pas nécessairement l'incertitude au sujet des estimations de mesures de performances, et qu'elle peut même la diminuer dans certains cas. Dans la suite, nous présentons l'intérêt pratique des expressions formelles des biais que nous avons obtenues.

6.5 Illustration de l'impact et la gestion du biais d'évaluation sur une application de classification d'images

Dans la deuxième partie de ce chapitre, nous développons un cas pratique montrant le problème induit par le bruit d'annotation lorsqu'on désire comparer des classifieurs, ainsi que les garde-fous que l'on peut mettre en place à l'aide des expressions des biais que nous avons obtenues.

Nous nous plaçons dans la situation particulière où un classifieur est déjà *établi* et opérationnel pour une tâche donnée, et où l'on dispose d'un autre classifieur *candidat* pour le remplacer. Par ailleurs, nous choisissons d'éviter le risque que, à cause d'un biais dans l'évaluation dû au bruit d'annotation, le classifieur candidat soit jugé meilleur que le classifieur établi à tort, et donc prenne sa place. Ce contexte peut se justifier par un point de vue industriel, où il est bien plus grave de remplacer un classifieur établi à tort, que de le maintenir alors que le classifieur candidat était en fait plus performant. Nous montrons les résultats obtenus lors d'une comparaison ignorant la problématique du bruit d'annotation, et nous la confrontons à des comparaisons plus prudentes, qui prennent en compte la présence d'exemples mal annotés.

6.5.1 Principe de l'expérience

Notre expérience se divise en trois phases, chacune étant contrôlable et reproductible :

1. obtenir un ensemble d'instances contenant du bruit d'annotation qui a été introduit de manière artificiel, mais qui soit le plus réaliste possible ;
2. entraîner plusieurs classifieurs sur ces instances bruitées ;
3. comparer deux à deux ces classifieurs avec différentes méthodes de comparaison, que nous détaillons en section 6.5.4, et présenter des statistiques synthétisant l'*efficacité* de chacune de ces méthodes.

Dans le cadre de cette expérience, nous nous intéressons aux *erreurs de jugement* pouvant être commise lors de la comparaison de deux classifieurs (cf. section 6.5.4). Nous considérons qu'une erreur de jugement est commise lorsque nous choisissons de garder le classifieur le moins bon parmi les deux qui sont comparés.

Concrètement, l'*efficacité* d'une méthode correspond donc à sa capacité à éviter les erreurs de jugement inacceptables, i.e. celle correspondant au choix du classifieur candidat, alors qu'en réalité, le classifieur établi est meilleur. Nous insistons sur le fait que l'objectif final de l'expérience est de mener une méta-comparaison *de plusieurs méthodes de comparaison de classifieurs*. Parmi les méthodes appliquées, nous en confrontons deux en particulier :

— la première ignore simplement le bruit d'annotation présent dans l'ensemble de test, et mène la comparaison de manière classique ;
— la deuxième, plus prudente, prend en compte la présence de bruit d'annotation en utilisant les expressions des biais dérivées précédemment en section 6.3.

Notre manière prudente de comparer deux classifieurs est, sur le principe, inspirée de la pratique de l'évaluation dans des domaines comme la médecine, où les conséquences d'un point de vue législatif sont très lourdes en cas d'erreur de jugement. Dans ces domaines, les sources de biais de l'évaluation sont identifiées au mieux et sont traitées de manière spécifiques. En particulier, les spécialistes en médecine intègrent clairement dans leurs procédures d'évaluation l'éventuelle présence de différents biais, comme par exemple celui découlant de la manière de traiter les données manquantes dans leurs études de cohorte. Par conséquent, des garde-fous stricts peuvent être mis en place pour s'assurer que ces biais ne font pas pencher la balance du mauvais côté. L'hypothèse du biais maximal, ou méthode du pire des cas, en est l'exemple le plus fort. Parfois considérée comme trop restrictive, cette méthode permet malgré tout de maintenir une prudence maximale lors de la procédure d'évaluation [CM00]. Ainsi, pour comparer deux traitements, l'un établi depuis longtemps, l'autre nouveau et moins connu, le principe consiste à envisager le pire des cas, i.e. le cas qui donnerait injustement le plus gros avantage au nouveau traitement, et la comparaison effectuée prend en compte et compense cet avantage résultant de ce cas précis (bien que rien ne certifie que le pire des cas ait véritablement lieu). Le nouveau traitement est alors choisi pour remplacer l'ancien seulement s'il se révèle sensiblement meilleur que ce dernier, malgré le désavantage qui lui a artificiellement été attribué lors de la comparaison.

6.5.2 Obtention des vérité-terrains bruitées

Avant d'en arriver à pouvoir comparer des classifieurs sur des instances bruitées, nous devons tout d'abord avoir accès à de telles instances, pour lesquelles nous connaissons les exemples mal annotés, de sorte à pouvoir réaliser l'expérience de manière contrôlée. Pour cela, il y a deux possibilités. La première est de posséder un ensemble d'instances, au préalable annoté par des annotateurs ayant un risque non négligeable de commettre des erreurs, et de le faire revoir entièrement par des annotateurs experts, de sorte à disposer d'une annotation bruitée ainsi que d'une annotation de meilleure qualité. Deux problèmes apparaissent si l'on veut appliquer cette option. D'une part, un tel ensemble est rare étant donné le coût de sa construction, car il doit être doublement annoté. D'autre part, s'il existe, cet ensemble serait de taille relativement faible, par rapport à l'effort nécessaire pour le construire, et il ne représenterait qu'un exemple parmi d'autres de la forme que peut prendre en pratique le bruit d'annotation, offrant donc peu de variabilité dans le contexte de l'expérience. En effet, expérimenter sur la comparaison de classifieurs en présence de bruit d'annotation, mais ne disposer que d'un ensemble avec un unique bruitage des instances, est assez limitant. En revanche, il est indéniable que cette approche possède un avantage important : elle permet d'obtenir un bruit d'annotation *réaliste*.

Inversement, l'autre option consiste à prendre un ensemble d'instances, que nous assimilons à la vérité-terrain absolue, et à le bruiter de façon artificielle par une procédure automatique. Contrairement à la première option, cette solution permet de générer autant de vérité-terrains bruitées que l'on veut, et est utilisable avec n'importe quel ensemble de départ. Cependant, obtenir une structure réaliste du bruit d'annotation avec une telle procédure n'est pas trivial. Dans l'objectif d'obtenir des résultats expérimentaux aussi proche que possible de la réalité, nous avons choisi d'introduire notre bruit d'annotation via un modèle de bruit par caractéristiques, que nous détaillons dans ce qui suit.

L'ensemble d'instances

Nous utilisons les instances de CIFAR10, pour différentes raisons.

— Tout d'abord, c'est un incontournable du domaine de la classification d'images, utilisé dans la plupart des études à ce sujet, et avec lequel les classifieurs construits sont évalués habituellement par une simple mesure de leur justesse.

— Il possède également un bon équilibre entre le nombres d'exemples qu'il contient, et la difficulté du problème de classification que ces exemples représentent :

  — il est formé de 60000 images au total, chacune contenant une entité appartenant à une des 10 catégories suivantes : avion, voiture, oiseau, chat, cerf, chien, grenouille, cheval, bateau, camion. Chaque image (figure 6.1) ;

  — chaque image contient une (et une seule) entité, et est de taille 32x32 pixels de couleur ;

  — chaque catégorie est représentée par 6000 images, 5000 pour l'entraînement, 1000 pour le test.

  Des travaux ont récemment identifié les erreurs d'annotations réelles contenues dans CIFAR10, ainsi que pour CIFAR100 dont nous aurons l'usage dans le chapitre 7 [NAM21]. Le bruit d'annotation moyen dans ces ensembles est estimé à 3.4%. Cette étude n'était cependant pas disponible à la date à laquelle nos expériences ont été réalisées.

En comparaison, l'ensemble SVHN consiste en plus de 600000 photographies couleur de numéro d'adresse, mais représente un problème de classification un peu plus simple, i.e. la reconnaissance d'un chiffre de 0 à 9. À l'opposée, l'ensemble PascalVoc contient des images en couleur de 20 catégories différentes d'objet du quotidien, et de taille approximative 300x500, constituant donc un problème de classification plus complexe encore que celui de CIFAR10, mais ne contient que 10000 exemples au total. Par conséquent, CIFAR10 s'est imposé comme le meilleur choix.

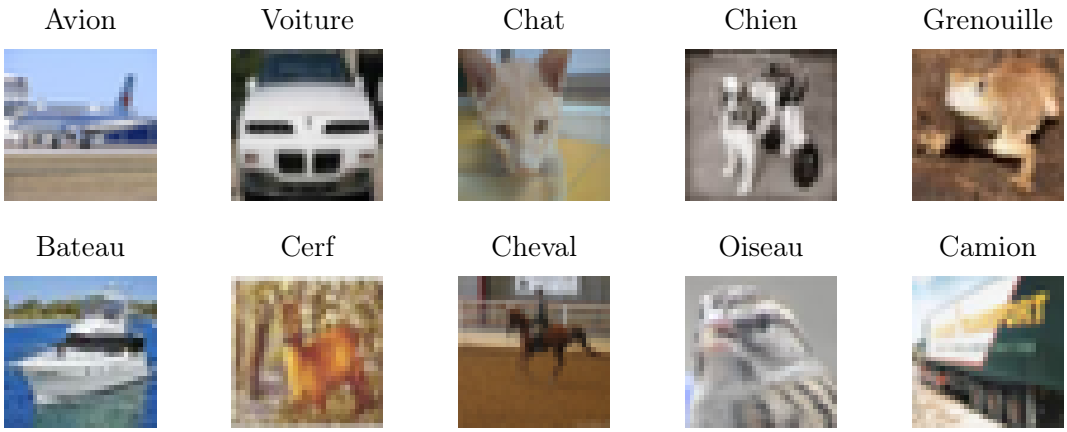


FIGURE 6.1 – Exemples d'images des 10 catégories de l'ensemble CIFAR10

Nous prenons donc l'annotation d'origine de CIFAR10 comme la vérité-terrain absolue de notre problème. En général, la vérité-terrain absolue n'est habituellement pas accessible dans un problème de classification réel. Cependant nous avons d'une part besoin de supposer que nous y avons accès dans le cadre de notre expérience. D'autre part, considérer que la vérité absolue est représentée par les annotations d'origine de CIFAR10 n'a qu'un impact négligeable : même si ces dernières contiennent en fait quelques erreurs, l'important dans notre contexte est la différence entre ce que nous mesurons sur les annotations d'origines, comparé à ce que nous mesurons sur les annotations perturbées par notre bruit artificiel.

Les différents modèles de bruit

Le point suivant à aborder est alors : comment peut-on obtenir une annotation bruitée des données de CIFAR10, de manière à ce que la structure du bruit d'annotation soit aussi proche de la réalité que possible. Lorsqu'on considère les trois différents modèles de bruit que l'on a présenté au chapitre 3, l'un des trois nous semble disqualifié d'office : le modèle de bruit uniforme. En effet, nous ne pensons pas qu'une procédure, qui modifie les annotations de manière aléatoire et indépendante des instances, permette d'obtenir un bruit d'annotation réaliste. Nous faisons ainsi l'hypothèse que le fait qu'un annotateur commette une erreur, et l'erreur précisément commise, dépendent de l'exemple qu'il doit annoter.

Il reste donc deux modèles potentiels : le bruit par classes et le bruit par caractéristiques. Le premier a l'avantage d'être facile à mettre en place. Il suffit de spécifier manuellement une matrice de bruitage, similaire à la matrice de transition de bruit, mais où les cellules représentent la probabilité qu'à une instance d'une classe donnée soit attribuée une étiquette donnée, ou autrement dit, la probabilité de confondre une classe par une autre. En voici une en guise d'exemple :

$$\begin{bmatrix} 1 & & & & & & & & & \\ & 1 & & & & & & & & \\ & & 0.9 & & 0.1 & & & & & \\ & & & 1 & & & & & & \\ & & 0.05 & & 0.95 & & & & & \\ & & & & & 1 & & & & \\ & & & & & & 1 & & & \\ & & & & & & & 1 & & \\ & & & & & & & & 1 & \\ & & & & & & & & & 1 \end{bmatrix}$$

Dans la matrice précédente, chaque ligne et chaque colonne correspondent à une des 10 catégories de CIFAR10. Les cellules laissées vides contiennent un 0. La première ligne, par exemple, signifie que les instances de la classe 1 ont 100% de chances d'être annotées correctement, indiqué par le 1 sur la diagonale. En ce qui concerne les erreurs d'annotation potentielles, il y a 10% de chances de confondre la classe 3 avec la classe 5, et 5% de chances de faire l'inverse. Cette matrice peut être construite de manière ad hoc, en attribuant des valeurs de probabilité raisonnables pour chaque couple ligne-colonne. Dans notre exemple, on pourrait imaginer que les classes 3 et 5 correspondent aux catégories « cheval » et « cerf » dans CIFAR10. La matrice de bruitage (ou de manière équivalente, la matrice de transition de bruit) peut également être construite en utilisant des annotateurs humains pour fournir une nouvelle annotation d'une partie de l'ensemble des instances, puis en comparant cette nouvelle annotation à l'annotation d'origine, de sorte à obtenir des estimations des probabilités de confusion pour chaque couple de classes. Une fois la matrice de bruitage obtenue, la procédure de bruitage consiste simplement à itérer sur l'ensemble d'instances, en les ré-annotant selon leur classe et la distribution de probabilité décrite par la ligne correspondante dans la matrice.

Pour notre expérience, nous n'avons pas fait le choix d'un modèle de bruit par classes. En effet, nous ne disposons pas d'annotateurs humains pour estimer une matrice de bruitage, et nous ne désirons pas la construire manuellement de façon arbitraire. De plus, nous ne sommes pas satisfaits par les liens de dépendance décrits par ce modèle, car nous pensons cela insuffisant pour conduire à des erreurs d'annotation réalistes. De ce point de vue là, le modèle de bruit uniforme, quant à lui, est encore moins satisfaisant que le modèle de bruit par classe, étant donné l'absence totale de dépendance entre une erreur d'annotation et l'instance concernée. Nous avons

donc pensé à une manière de bruiter nos instances par rapport à leurs caractéristiques, en accord avec le modèle de bruit le plus complexe. Le principe est d'entraîner des réseaux de neurones pour jouer le rôle d'annotateurs artificiels, de sorte à pouvoir annoter différemment les instances de CIFAR10. Il y a donc deux étapes à réaliser :

— obtenir un nombre suffisant d'annotateurs artificiels ;
— les utiliser dans le cadre d'un procédure de vote pour bruiter les instances de CIFAR10.

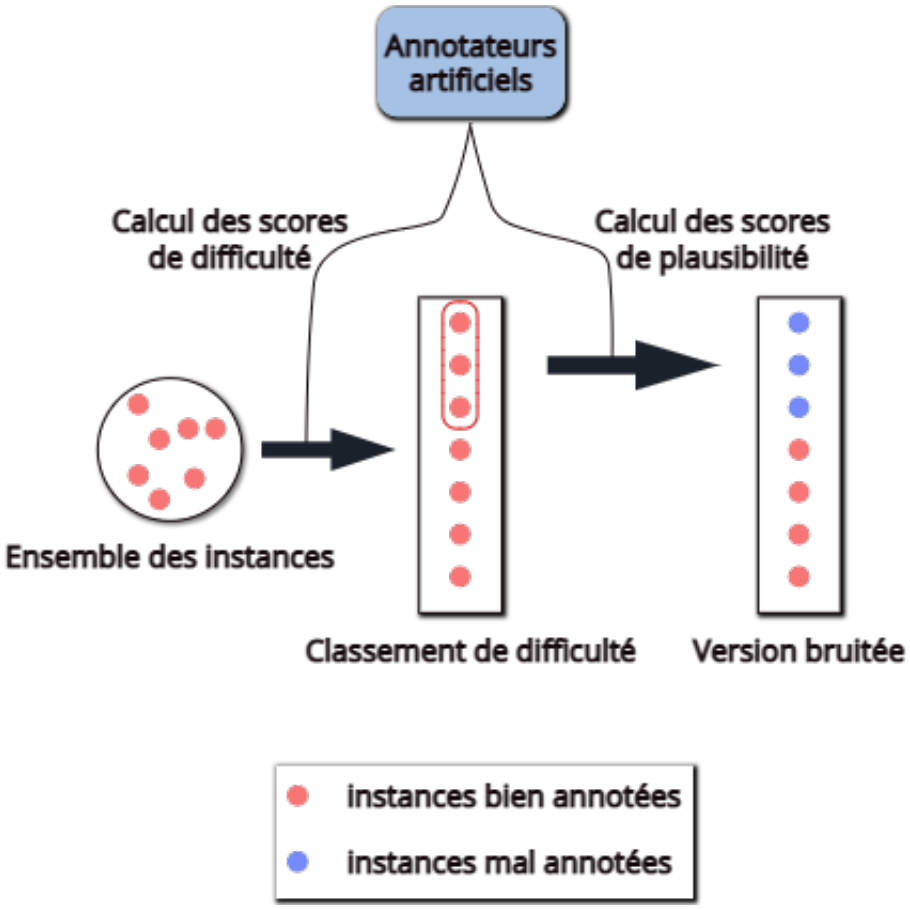


FIGURE 6.2 – Schéma explicatif de la procédure de vote des annotateurs artificiels permettant d'obtenir une version bruitée à $x\%$ d'un ensemble d'instances.

Construire les annotateurs artificiels

Dans notre expérience, nous avons utilisé une centaine d'annotateurs artificiels. Notre objectif est que ces annotateurs soient suffisamment différents les uns des autres pour en terme de prédiction et d'erreurs réalisées, de sorte que la procédure de vote que nous voulons appliquer par la suite reste pertinente.

Pour les construire, nous avons mis à profit un modèle particulier de réseaux de neurones convolutifs : la Fabrique de Réseaux Convolutifs (FRC). La structure de ce modèle est complexe, et nous l'expliquons en profondeur dans le chapitre 7. Pour l'instant, il est suffisant de savoir que cette structure est déterminée par deux hyperparamètres : le nombre de couches et le nombre de chaînes. Manipuler ces paramètres permet de générer des structures sensiblement différentes

Algorithme 1 : Algorithme de construction des annotateurs artificiels.

entrées : $\mathcal{E} = \mathcal{E}_A \cup \mathcal{E}_T$: l'ensemble d'instances, séparé en ensemble d'apprentissage et de test ;
(l_{min}, l_{max}) : le nombre de couche minimal (resp. maximal) pour la FRC ;
n : la puissance de 2 correspondant aux nombre de chaînes de la FRC ;
p : le pourcentage selon lequel la FRC est élaguée avant son entraînement ;
ξ : critère d'éligibilité dans le groupe d'annotateurs artificiels $\mathbb{G}$;
ϕ : critère de fin d'entraînement de la FRC ;
N : le nombre d'annotateurs artificiels à obtenir ;
e_{min} : nombre minimal d'epochs d'entraînement qu'un annotateur artificiel doit avoir effectué.

sortie : L'ensemble $\mathbb{G}$ des annotateurs artificiels

$\mathbb{G} = \emptyset$;
while $|\mathbb{G}| < N$ **do**
  Initialiser une FRC avec $l = random(l_{min}, l_{max})$ couches, et 2^n chaînes ;
  Élaguer p% de la FRC ;
  while $\phi(FRC)$ *est faux* **do**
    Entraîner pendant 1 epoch sur $\mathcal{E}_A \cup \mathcal{E}_T$;
    if *Nombre d'epochs effectuées* $> e_{min}$ *et* $\xi(FRC)$ *est vrai* **then**
      Ajouter la FRC à $\mathbb{G}$;
    end
  end
end

dans les prédictions qu'elles apprennent à faire. Les FRC faisant office d'annotateurs artificiels sont construites de la manière suivante (cf. algorithme 1) :

- le nombre de couches de la FRC est choisi aléatoirement entre 4 et 16 ; nous avons choisi ces bornes en nous fondant sur les essais réalisés par Verbeek dans [SV16] ;
- le nombre de chaînes est une puissance de 2 comprise entre 2^3 et 2^7, choisie également aléatoirement ;
- un pourcentage p est aussi choisi de manière aléatoire entre 0% et 90%, et représente la fraction de la structure que l'on élague (cf. chapitre 7) ;
- la structure finale est alors entraînée sur la totalité des instances de CIFAR10 ; cela signifie qu'elle est également testée sur des instances faisant partie de son ensemble d'apprentissage ; nous mesurons en effet la qualité d'un annotateur artificiel par son erreur de substitution, car sa capacité à généraliser sur de nouvelles instances ne lui est d'aucune utilité ;
- l'entraînement se déroule comme suit : un algorithme de gradient stochastique est appliqué sur la structure avec un pas d'apprentissage de 0.1, un moment de 0.9 et une dégradation de pondération de 0.0005 ; l'implantation est faite en python avec le module PyTorch ; le modèle est entraîné pendant 50 epochs, puis le pas d'apprentissage est divisé par 10 pour les 30 epochs suivantes, et par 100 pour les epochs passées la 80ÈME ; l'entropie croisée est utilisée comme fonction de perte ;
- à partir de l'epoch 10, et après chaque epoch suivante, un critère d'éligibilité ξ permet de vérifier si l'état actuel du modèle peut être ajouté au groupe $\mathbb{G}$ des annotateurs artificiels : si le taux d'erreurs du modèle, mesuré sur la totalité des instances sans bruit d'annotation,

est inférieur à 40%, et proche d'un multiple de 5, le modèle est ajouté dans son état actuel à $\mathbb{G}$, et s'il existe déjà dans le groupe une version du même modèle avec le même taux d'erreurs, ayant été ajoutée à une epoch précédente, il la remplace ; discrétiser les taux d'erreurs des FRC obtenues selon les multiples de 5 sert simplement à obtenir de potentiels annotateurs artificiels de qualité variable, commettant de 0% à 40% d'erreurs ;

— l'entraînement s'arrête soit lorsque le nombre maximum d'epochs a été effectué, soit de manière prématurée lorsqu'une erreur a lieu, e.g. la fonction de perte explose, ou alors lorsque le modèle ne s'améliore plus et que son taux d'erreurs stagne durant trop d'epochs consécutives ;

— la procédure est alors répétée sur une nouvelle FRC générée aléatoirement, jusqu'à ce qu'un nombre suffisant d'annotateurs artificiels soit atteint.

Algorithme 2 : Algorithme d'introduction de bruit d'annotation.

entrées : η : taux de bruit désiré dans l'ensemble des instances ;
$\mathbb{G}$: l'ensemble des annotateurs artificiels ;
$\mathcal{E}$: l'ensemble des instances à bruiter ;
$\mathcal{I}$: l'ensemble des catégories du problème de classification.
sortie : L'ensemble d'instances $\mathcal{E}$ bruité à η%
for $K \in \mathbb{G}$ **do**
  Attribuer un score de légitimité l_K à K ;
end
$\mathbb{K} = []$ est une liste vide ;
for $i \in \mathcal{E}$ **do**
  L_i = dictionnaire associant à chaque catégorie c les annotateurs artificiels ayant classifié i dans c ;
  À l'aide de L_i, récupérer les annotateurs ayant mal classifié i ;
  calculer d_i, le score de difficulté d'interprétation de i, en sommant les légitimités l_K de ces annotateurs ;
  Ajouter à $\mathbb{K}$ le couple (i, d_i) ;
end
Trier $\mathbb{K}$ selon les valeurs décroissantes des d_i ;
$\mathbb{B}_\eta$: ensemble formé par les instances i des $\eta * |\mathbb{K}|$ premiers couples de $\mathbb{K}$;
for $i \in \mathbb{B}_\eta$ **do**
  for $c \in \mathcal{I}$ **do**
    if $c =$ *annotation de* i **then**
      continue ;
    end
    Récupérer à l'aide de L_i les annotateurs ayant classifié i dans la catégorie c ;
    Calculer $s_{c,i}$, le score de plausibilité de l'étiquette c pour i, en sommant les légitimités l_K de ces annotateurs ;
  end
  Modifier l'annotation de i par la catégorie ayant obtenu le plus grand score $s_{c,i}$;
end

Bruiter les instances

Une fois les annotateurs artificiels obtenus, nous les utilisons dans le cadre d'une procédure de vote pour construire une version de CIFAR10 bruitée à $\eta\%$ (Figure 6.2). La procédure se déroule comme suit (cf. algorithme 2) :

- attribuer un score de *légitimité* l_K à chaque annotateur K, représentant la confiance que l'on peut attribuer à ses annotations ; nous utilisons pour cela simplement l'erreur de substitution de l'annotateur ;
- pour chaque instance i de CIFAR10, récolter les prédictions des annotateurs artificiels, et calculer un score d_i représentant la difficulté d'interprétation de i ; pour cela, nous sommons les légitimités des annotateurs *qui ont mal classifié* i par rapport à la vérité-terrain absolue ;
- construire le classement $\mathbb{K}$ de difficulté des instances, en ordonnant les instances par rapport à leur score de difficulté, du plus grand au plus petit ;
- pour bruiter un pourcentage η de l'ensemble CIFAR10, considérer les sous-ensemble $\mathbb{B}_\eta$ formé des $\eta\%$ des instances les plus difficiles selon $\mathbb{K}$;
- pour une instance $i \in \mathbb{B}_\eta$, pour chaque catégorie c différente de la classe de i, obtenir un score $s_{i,c}$ représentant la *plausibilité* que l'étiquette c soit attribuée à i ; pour cela, nous sommons les légitimités des annotateurs *qui ont classifié* i *dans la catégorie* c ;
- choisir la catégorie ayant le plus grand score de plausibilité comme nouvelle annotation pour i ;
- faire de même pour les autres instances de $\mathbb{B}_\eta$.

Procédure d'introduction de bruit par caractéristiques

Nous avons donc développé une procédure permettant d'obtenir une version bruitée à $\eta\%$ d'un ensemble d'instances, de sorte que les erreurs d'annotations dépendent effectivement des caractéristiques de ces instances. Cette procédure est reproductible, et contrôlable par différents paramètres :

- la qualité des annotateurs artificiels peut être ajustée, en modifiant leur modèle d'apprentissage ou l'algorithme d'entraînement, et en considérant un critère d'éligibilité adapté à la situation ;
- les définitions des scores de légitimité d'un annotateur artificiel, de difficulté d'interprétation d'une instance, et de plausibilité d'une étiquette pour une instance, peuvent être modifiés aisément ;
- le taux de bruit introduit dans l'ensemble peut être choisi librement, sans devoir réentraîner d'autres annotateurs artificiels, ou effectuer à nouveau les calculs liés aux scores susmentionnés : en effet, déterminer le classement de difficulté des instances, ainsi que la plausibilité des étiquettes pour chaque instance, n'est nécessaire à faire qu'une seule fois, du moment que l'on sauvegarde les résultats.

De plus, il y a une cohérence entre une version de l'ensemble bruitée $\eta\%$, et une autre à $\xi\%$, où $\xi \neq \eta$. Par exemple, passer de 5% à 10% de bruit se fait simplement en bruitant les 5% des instances suivantes dans le classement de difficulté $\mathbb{K}$, et non en bruitant 10% d'instances complètement différentes. En guise d'exemple, la figure 6.3 montre la matrice de transition de bruit que nous obtenons pour la version bruitée à 10%.

	avion	voit.	oiseau	chat	cerf	chien	gren.	cheval	bateau	camion
avion	1135	3	15	10	0	0	6	4	15	12
voit.	10	1160	1	2	0	6	0	0	10	11
oiseau	48	0	1024	22	30	22	35	12	4	3
chat	17	3	52	936	25	92	40	20	6	9
cerf	20	0	33	28	1063	7	18	27	4	0
chien	5	2	31	120	32	947	16	47	0	0
gren.	9	0	33	24	7	7	1110	6	3	1
cheval	9	0	14	27	16	21	2	1103	2	6
bateau	24	8	7	3	1	1	0	2	1146	8
camion	17	20	2	3	2	2	3	3	11	1137

FIGURE 6.3 – Matrice de transition de bruit pour la version bruitée à 10% de CIFAR10. Les lignes représentent les vraies classes des instances, les colonnes correspondent aux étiquettes bruitées.

6.5.3 Obtention des classifieurs à comparer

Une fois les annotateurs artificiels obtenus, et les versions bruitées de CIFAR10 générées, nous pouvons nous intéresser au problème de la comparaison de classifieurs. Nous voulons donc entraîner 100 classifieurs sur les instances de CIFAR10, dans le but de les comparer deux à deux par la suite sur différentes versions bruitées de l'ensemble de test.

Il y a cependant un point important à prendre en compte. Le fait de ne pouvoir utiliser qu'une centaine de classifieurs, obtenus via des structures similaires , i.e. les FRC, ne favorisent pas l'aspect statistique de notre expérience. Dans l'idéal, pour pouvoir tirer des conclusions statistiques intéressantes sur les résultats des comparaisons deux à deux, il faudrait pouvoir faire un tirage aléatoire dans une population idéale $\mathcal{P}^*$ de classifieurs variées, contenant des classifieurs ayant différentes performances, et différents biais de prédiction. Malheureusement, $\mathcal{P}^*$ est en pratique inaccessible, et probablement non-dénombrable.

Par ailleurs, pour des raisons de contrôle et de reproductibilité de nos expériences, nous avons choisi de nous limiter à l'utilisation de la même structure d'apprentissage (FRC) pour chaque classifieur. Ainsi, en entraînant nos classifieurs à partir de la même structure, du même algorithme d'apprentissage et de données identiques, la population $\mathcal{P}$ à partir de laquelle nous tirons notre échantillon de classifieurs est grandement réduite par rapport à $\mathcal{P}^*$. C'est pourquoi nous mettons en place différents procédés pour maximiser la variabilité dans l'échantillon des 100 classifieurs que nous voulons obtenir. Nous admettons en conséquence la possibilité d'un biais expérimental, mais il est hors du champ de la thèse d'essayer de le résoudre.

Concrètement, voici les deux principales règles que nous respectons :

— chaque classifieur est entraîné sur une seule version de CIFAR10, mais cette version n'est pas la même selon le classifieur considéré ; ainsi, certains classifieurs sont entraînés sur la version sans bruit, tandis que d'autres le sont sur la version bruitée à e.g. 30% ;

— nous appliquons un critère pour choisir les 100 classifieurs que nous garderons pour la suite ; ce critère permet d'assurer la variabilité des indicateurs F_c^k, F_n^k et F_r^k, qui synthétisent les prédictions d'un classifieur par rapport au bruit d'annotation.

Les mesures de justesse des 100 classifieurs que nous choisissons varient de 55% à 90%. Bien que de nos jours, avec un modèle d'apprentissage profond efficace, il est aisé d'obtenir un score dépassant les 95% de justesse sur CIFAR10, dans le cadre de notre expérience, il n'y aurait que peu d'intérêt à comparer uniquement des couples de classifieurs présentant constamment les

mêmes taux de justesse, proche de 100% et égaux à 1% près. Nous préférons donc plutôt obtenir des classifieurs à performances variées, quitte à ce que la justesse de certains soit beaucoup plus basse.

Protocole d'entraînement

Le protocole d'entraînement est similaire à celui mis en place pour les annotateurs artificiels, nous n'allons donc pas le répéter ici. Les différences sont les suivantes :

— les instances sont divisées en ensemble d'entraînement et de test ; nous n'avons pas gardé la division initiale de CIFAR10 en ensemble d'apprentissage et de test. Plutôt, nous avons considéré 80% des instances pour l'entraînement et le reste pour l'évaluation, en veillant à maintenir la même distribution des classes dans chaque ensemble ;

— le critère d'éligibilité des classifieurs est différent de celui pour les annotateurs artificiels, et divisé en deux parties ; le premier critère est appliqué durant l'entraînement des classifieurs, de la même manière que pour les annotateurs artificiels ; cependant, le choix qu'il implique n'est pas définitif, il constitue seulement un premier filtre ; une fois qu'un nombre suffisant de classifieurs a franchi ce premier filtre, un deuxième critère est appliqué, réduisant le nombre de classifieurs à 100.

Les critères d'éligibilité

Nous expliquons ici le principe des critères d'éligibilité qui nous permettent de construire un groupe de classifieurs variés destinés à être comparés, que nous appellerons « groupe des élus » à partir de maintenant.

Nous considérons pour cela l'ensemble d'indicateurs que nous avons présentés dans la section 6.3.2 : F_c^k, F_n^k et F_r^k, pour chaque catégorie k du problème de classification. Pour rappel, ces indicateurs représentent respectivement, pour un classifieur donné, la fraction d'instances qu'il a correctement prédit parmi celles bien annotées, la fraction d'instances mal annotées pour lesquelles il a prédit l'étiquette bruitée, et la fraction d'instances mal annotées pour lesquelles il est parvenu à retrouver la vraie catégorie. Nous les appellerons dorénavant « indicateurs de prédiction », et nous les noterons de la manière compacte suivante : $(F_c^k, F_n^k, F_r^k)_{k\in\mathcal{I}}$, où $\mathcal{I}$ est l'ensemble des catégories. Dans la partie précédente, nous avons vu que les biais des estimateurs des mesures telles que la justesse, la précision, le rappel ou encore la F-mesure dépendaient de ces indicateurs. Dans l'objectif de choisir des classifieurs offrant par la suite des situations de comparaisons variables, nos critères d'éligibilité sont donc adaptés. En effet, choisir des classifieurs qui présentent des indicateurs de prédictions variés implique que les biais des estimateurs de performance diffèrent selon le classifieur considéré.

Voici donc le principe de ces deux critères. Le premier est appliqué pendant l'entraînement d'un classifieur, et conduit à un choix non définitif. Prenons par exemple un classifieur en entraînement, à une epoch quelconque. A la fin de l'epoch, le critère est appliqué sur le classifieur, que nous appelons dans ce contexte le « classifieur éligible » $\mathcal{C}_2$:

— nous parcourons l'ensemble des classifieurs $\mathcal{C}_1$ faisant déjà partie du groupe des élus, de sorte à vérifier que $\mathcal{C}_2$ n'est pas trop proche d'un autre classifieur ayant déjà été élu ;

— pour cela, pour chaque classifieur $\mathcal{C}_1$ élu, nous calculons l'erreur quadratique moyenne e_q entre le vecteur $V_{\mathcal{C}_1}$ des indicateurs de prédictions de $\mathcal{C}_1$, et le vecteur $V_{\mathcal{C}_2}$ des indicateurs de $\mathcal{C}_2$;

— si e_q est trop petit, i.e. plus petit qu'un seuil de proximité τ défini manuellement, le critère échoue, et l'entraînement continue ;

— sinon $\mathcal{C}_2$ est ajouté au groupe des élus ;
— par ailleurs, si $e_q < \tau$, mais que $\mathcal{C}_1$ se trouve être une version moins performante de $\mathcal{C}_2$, i.e. ayant été élue à une epoch précédente, $\mathcal{C}_2$ remplace $\mathcal{C}_1$ dans le groupe des élus.

Nous fixons arbitrairement le seuil de proximité τ à 10^{-4}. Ce paramètre n'est cependant pas réellement déterminant dans la procédure, car le premier critère ne constitue qu'un filtre de choix grossier. La véritable phase de choix des 100 modèles survient après la formation d'un groupe des élus suffisamment grand. En effet, nous construisons de cette manière un groupe contenant plus de 400 élus.

Nous appliquons ensuite un deuxième critère pour choisir les 100 classifieurs impliquant le plus de variabilité. En une dimension, le problème est similaire à choisir un groupe de 100 points sur une ligne en contenant 400, de sorte à ce que les 100 points choisis soient le plus écartés les uns des autres (cf. Figure 6.4). Pour cela, nous choisissons d'utiliser la notion de variance de la manière suivante :

— un des 400 élus est choisi aléatoirement pour appartenir au groupe final, de sorte à initialiser la procédure ;
— des vecteurs sont construits pour chaque indicateur de type $(F_c^i, F_n^i, F_r^i)_{i\in\mathcal{I}}$ au fur et à mesure qu'un élu est ajouté au groupe final : par exemple, au départ, le vecteur $v_{F_c^1}$ ne contient qu'un élément, i.e. la valeur de F_c^1 pour le premier élu choisi ;
— le deuxième élu est choisi de manière à maximiser la moyenne m_1 des variances des vecteurs v_d pour chaque indicateur d ;
— à partir de là, une deuxième classe de vecteurs s_d est construite pour chaque indicateur d, en effectuant les différences successives des composantes des vecteurs initiaux : e.g. le vecteur v_d est ordonné de la plus petite composante à la plus grande, et en le notant $(v_d^{(1)}, ..., v_d^{(k)})$, le vecteur s_d est alors $(s_d^{(2)} - s_d^{(1)}, ..., s_d^{(k)} - s_d^{(k-1)})$;
— chaque élu supplémentaire est alors choisi dans l'objectif de maximiser la moyenne m_1 tout en minimisant la moyenne m_2 des variances des vecteurs s_d.

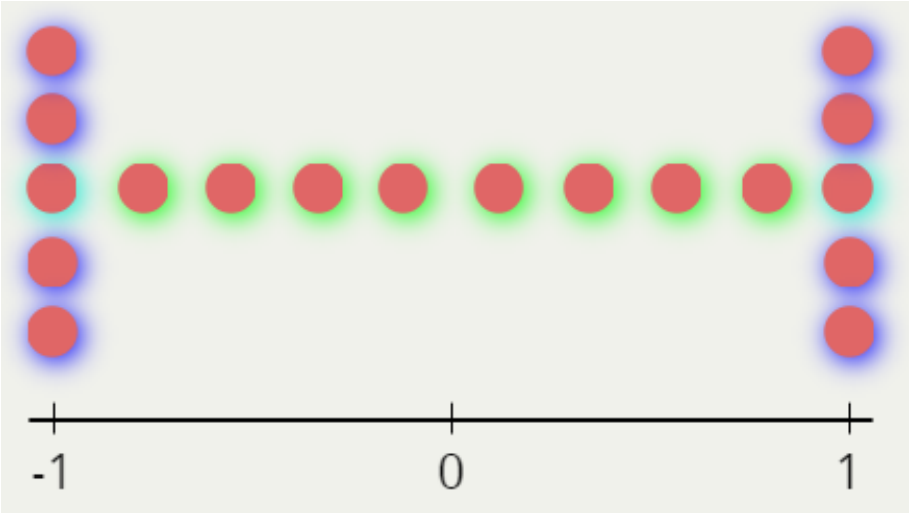


FIGURE 6.4 – Exemple d'échantillons de 10 points choisis parmi 18 points disposés sur une ligne de -1 à 1. Les points à la verticale ont tous la même abscisse. Si l'on veut uniquement maximiser la variance de l'échantillon, il faut choisir les 5 points à gauche et les 5 points à droite, à la verticale. Si l'on veut que l'échantillon choisi soit celui des 10 points horizontaux, il faut de plus minimiser l'écart entre chaque point.

Les classifieurs parmi les 400 élus initialement ont été entraînés sur des versions bruitées différentes de CIFAR10. En moyenne, il y a 70 classifieurs pour les valeurs suivantes de bruit d'annotation : 1%, 5%, 10%, 20% et 30%. Après la phase de choix finale, parmi les 100 classifieurs restants, la répartition est beaucoup plus déséquilibrée. En effet, il reste :

— 68 classifieurs entraînés sur la version bruitée à 1% de CIFAR10 ;
— 4 pour la version bruitée à 5% ;
— 6 pour la version à 10% ;
— 8 pour la version à 20% ;
— 14 pour la version à 30% ;

Cela n'est cependant pas dérangeant, l'aspect qui nous intéresse étant la variabilité des indicateurs de prédictions des classifieurs élus. La figure 6.5 montre la dispersion de ces indicateurs pour les 100 classifieurs finalement élus. Les valeurs de chaque indicateur sont représentées par les points sur chaque ligne, à raison d'un point par classifieur. Il y a donc 100 points par ligne. Les indicateurs sont ordonnées de 0 à 9 pour les 10 classes du problème, de bas en haut, et ont été calculé sur l'ensemble de test contenant 1% de bruit. Les indicateurs notés P_k représentent simplement la fraction d'exemples prédits dans chaque classe par chacun des classifieurs, ils ne font pas directement partie des indicateurs de prédictions définis en section 6.3, et sont montrés à titre indicatif.

Il est intéressant d'observer que de nombreux classifieurs prédisent fréquemment l'étiquette bruitée des exemples mal annotés, étant donné les points proche de 1 pour les indicateurs de type F_n. Les classifieurs ont par ailleurs logiquement plus de difficultés à retrouver la vraie classe des exemples mal annotés, ce qui est montré par la dispersion des indicateurs de type F_r, mais certains semblent tout de même y parvenir correctement.

Selon la dispersion globale des indicateurs d'intérêt, les 100 classifieurs arborent effectivement des comportements différents par rapport au bruit d'annotation. Les indicateurs de type F_n varient approximativement de 0 à 1, ceux de type F_c varient en moyenne de 0.25 à 1, tandis que ceux de type F_r varient de 0 à 0.8 au maximum. De cette manière, les estimateurs de performance de chaque classifieur présentent des biais différents, ce qui enrichit les situations de comparaisons deux à deux que nous pouvons effectuer.

6.5.4 Comparaison des classifieurs élus

Une fois le groupe des 100 classifieurs élus construit, nous pouvons passer à la phase de comparaison deux à deux de ces classifieurs. Notre objectif ici est de confronter différentes méthodes de comparaison, pour identifier la plus efficace en présence de bruit d'annotation. Par conséquent, pour chaque méthode de comparaison Ψ, nous parcourons l'ensemble des couples $(\mathcal{C}_1, \mathcal{C}_2)$ de classifieurs élus, $\mathcal{C}_1 \neq \mathcal{C}_2$, et déterminons le résultat de la comparaison. Chaque méthode Ψ est une fonction qui prend en argument un couple de classifieurs $(\mathcal{C}_1, \mathcal{C}_2)$, et qui en choisit un seul en sortie.

Nous distinguons trois situations possibles pour la comparaison entre $\mathcal{C}_1$ et $\mathcal{C}_2$, et adoptons les notations suivantes pour les décrire :

— $\mathcal{C}_1 \triangleright_\Psi \mathcal{C}_2$: la comparaison montre avec suffisamment de certitude que $\mathcal{C}_1$ est meilleur que $\mathcal{C}_2$;
— $\mathcal{C}_1 \triangleleft_\Psi \mathcal{C}_2$: inversement, la comparaison montre avec suffisamment de certitude que $\mathcal{C}_2$ est meilleur que $\mathcal{C}_1$;
— $\mathcal{C}_1 \triangle_\Psi \mathcal{C}_2$: la comparaison ne permet pas de montrer de manière suffisamment certaine que l'un est meilleur que l'autre.

De plus, nous nous plaçons dans le contexte où, lorsque deux classifieurs sont comparés, le

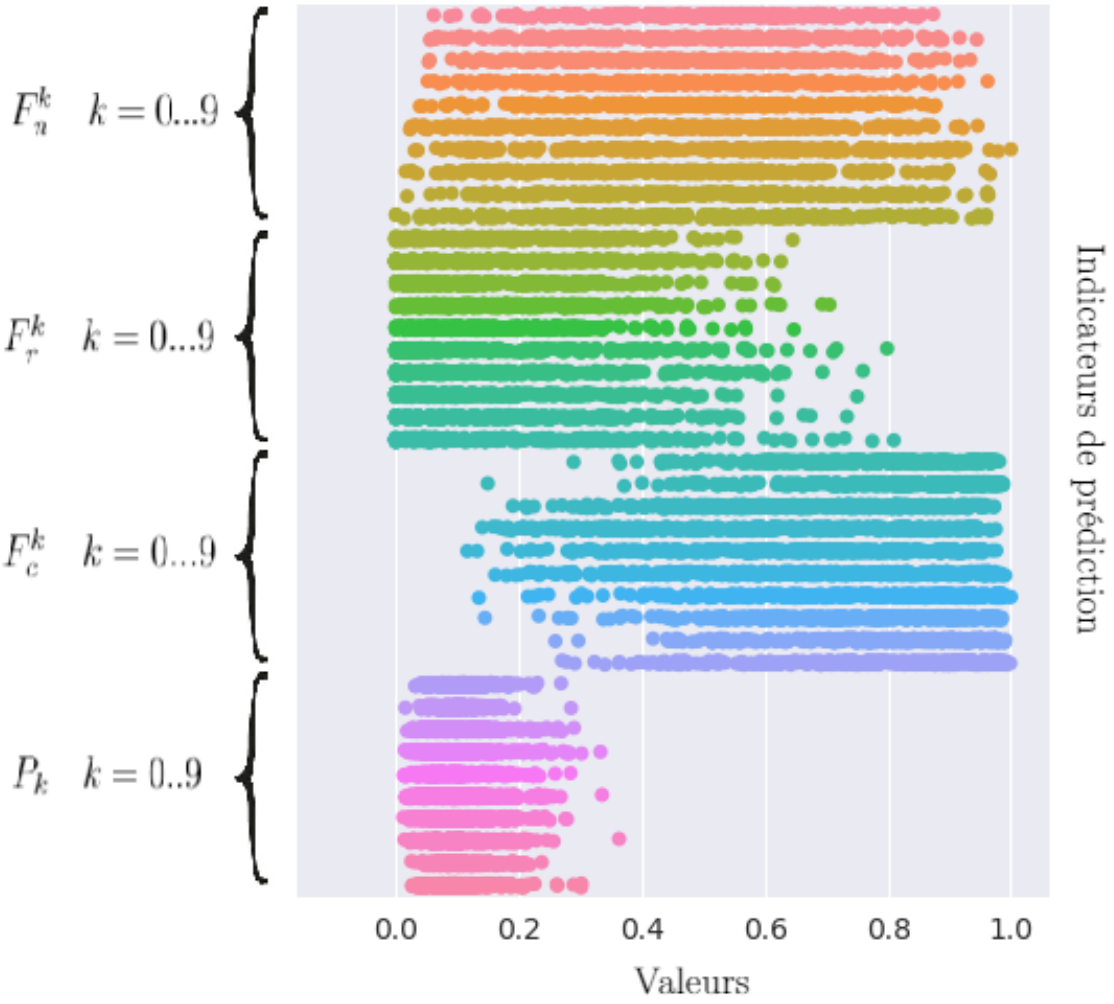


FIGURE 6.5 – Dispersion des indicateurs de prédiction des 100 classifieurs élus.

premier a fait ses preuves et est établi depuis un certain temps, tandis que le second est candidat à le remplacer. Cela signifie que, lorsqu'on considère le couple $(\mathcal{C}_1, \mathcal{C}_2)$, $\mathcal{C}_1$ prend le rôle du classifieur établi, tandis que $\mathcal{C}_2$ a celui du classifieur candidat. Ainsi, les fonctions Ψ sont toutes *non commutatives* : comparer $(\mathcal{C}_1, \mathcal{C}_2)$ n'est pas équivalent à comparer $(\mathcal{C}_2, \mathcal{C}_1)$, car le classifieur établi change d'un cas à l'autre.

Forme générale d'une méthode de comparaison Ψ

Dans le cas général, une méthode de comparaison Ψ prend la forme suivante :

$$\Psi(\mathcal{C}_1, \mathcal{C}_2) = \begin{cases} \mathcal{C}_1 & \text{si } \mathcal{C}_1 \triangleright_\Psi \mathcal{C}_2 \\ \mathcal{C}_1 & \text{si } \mathcal{C}_1 \triangle_\Psi \mathcal{C}_2 \\ \mathcal{C}_2 & \text{si } \mathcal{C}_1 \triangleleft_\Psi \mathcal{C}_2 \end{cases}, \tag{6.33}$$

$$\text{et} \quad \Psi(\mathcal{C}_1, \mathcal{C}_2) \neq \Psi(\mathcal{C}_2, \mathcal{C}_1).$$

Dans notre contexte, le classifieur établi $\mathcal{C}_1$ possède le privilège de l'ancienneté. Autrement dit, quelle que soit Ψ, lorsque $\mathcal{C}_1 \triangle_\Psi \mathcal{C}_2$, $\mathcal{C}_1$ est maintenu.

Définition des deux types d'erreurs de jugement

Une erreur de jugement correspond au résultat d'une comparaison Ψ qui diffère du résultat de la comparaison idéale. La comparaison idéale est notée Ψ^*. Elle correspond à proprement parler à une comparaison non perturbée par du bruit d'annotation. Par conséquent, nous l'obtenons en utilisant les instances de test munies de leur étiquettes d'origine. Les autres méthodes Ψ n'ont évidemment pas accès à ces annotations non bruitées.

Les erreurs de type I et II

Il y a donc deux erreurs de jugements qu'une méthode de comparaison Ψ peut commettre :
- l'erreur de type I signifie que le classifieur établi est lésé : $\Psi(\mathcal{C}_1, \mathcal{C}_2) = \mathcal{C}_2$, alors que $\Psi^*(\mathcal{C}_1, \mathcal{C}_2) = \mathcal{C}_1$;
- l'erreur de type II signifie inversement que le classifieur candidat est lésé : $\Psi(\mathcal{C}_1, \mathcal{C}_2) = \mathcal{C}_1$, alors que $\Psi^*(\mathcal{C}_1, \mathcal{C}_2) = \mathcal{C}_2$.

Notre expérience est conçue pour établir quelle méthode de comparaison Ψ permet le plus efficacement d'éviter les erreurs de type I. En particulier, la méthode d'évaluation prudente que l'on expérimente est pensée spécifiquement pour empêcher ce type d'erreur. Cependant, nous analysons également ce qu'offrent d'autres méthodes Ψ, dans le cas où l'objectif est plutôt d'obtenir un bon compromis entre les taux d'erreurs des deux types, bien que l'erreur de type I reste la priorité quelle que soit la méthode appliquée.

Principe d'une méthode de comparaison Ψ

Chaque méthode de comparaison Ψ est appliquée selon le schéma suivant :

1. une mesure est choisie pour l'estimation des performances des classifieurs ;
2. pour chaque classifieur, une estimation de cette mesure est réalisée *sur les instances de test* ;
3. pour chaque estimation, des *intervalles de confiance* à 5% sont calculés ;
4. on parcourt ensuite l'ensemble des couples $(\mathcal{C}_1, \mathcal{C}_2)$ pour déterminer le résultat de $\Psi(\mathcal{C}_1, \mathcal{C}_2)$, en les comparant sur la base des intervalles de confiances i^1_{conf} et i^2_{conf} correspondant :
   - s'ils ont une région en commun, nous concluons que $\mathcal{C}_1 \triangle_\Psi \mathcal{C}_2$;
   - si le minimum de i^1_{conf} est supérieure au maximum de i^2_{conf}, nous concluons que $\mathcal{C}_1 \triangleright_\Psi \mathcal{C}_2$;
   - dans le cas contraire, nous concluons que $\mathcal{C}_1 \triangleleft_\Psi \mathcal{C}_2$.

Les deux parties de la procédure notées en italique correspondent aux endroits où deux méthodes de comparaison peuvent différer. Par exemple, pour la méthode de comparaison idéale, les instances de test sont prises avec leur véritable annotation, contrairement aux autres méthodes qui considèrent les étiquettes bruitées. Deux méthodes Ψ_1 et Ψ_2 obtiennent également des intervalles de confiance différents pour la même estimation, comme nous l'expliquons dans ce qui suit.

Les différentes méthodes de comparaison

Voici les différentes méthodes de comparaison que nous appliquons :

- **référence** : comparaison idéale où les classifieurs sont évalués sur les instances de test non bruitées ;
- **classique** : les classifieurs sont évalués et comparés simplement en mesurant leur performance sur les instances à disposition, i.e. mal annotées, mais sans prendre de précaution particulière par rapport au bruit d'annotation ;
- **prudente-v1** : conçue pour éviter à tout prix les erreurs de jugement de type I, cette méthode met en œuvre le principe de comparaison à biais maximal ;
- **prudente-v2** : version différente de la méthode précédente, plus tolérante, que nous abordons en détails dans ce qui suit ;
- **classique avec procédures de nettoyage** : comme son nom l'indique, cette méthode est similaire à la méthode **classique**, mais au préalable, une procédure de nettoyage des exemples de test est appliquée.

Dans la suite, Ψ^* désigne toujours la comparaison de référence, et les opérateurs de comparaison des classifieurs pour cette méthode sont notés $\triangleright^*$, $\triangleleft^*$ et $\triangle^*$. La notation Ψ est utilisée pour les autres méthodes, sans distinction particulière selon la méthode lorsque le contexte ne présente pas d'ambiguïté, et les opérateurs correspondant sont notés $\triangleright_\Psi$, $\triangleleft_\Psi$ et $\triangle_\Psi$. Chaque méthode est expliquée en détails dans la section courante.

L'efficacité de chaque méthode est quantifiée par son taux d'erreurs de type I, et notre objectif est de confronter l'efficacité de la méthode **classique** à celle des autres méthodes, en présence de bruit d'annotation. Les méthodes prudentes sont précisément conçues pour éviter les erreurs de type I, au détriment d'une augmentation significative du nombre d'erreurs de type II. En ce sens, ces méthodes sont plutôt adaptées à une application dans un milieu industriel, et non académique. Les méthodes comprenant l'utilisation d'une procédure de nettoyage permettent d'obtenir un bon compromis entre les taux d'erreurs des deux types, sous certaines conditions en ce qui concerne la qualité du nettoyage.

Sur la superposition d'intervalles de confiance

La comparaison entre deux classifieurs donnés est effectuée ici par une comparaison des intervalles de confiance obtenus pour leurs performances respectives. En réalité, la méthode conventionnelle pour comparer deux classifieurs est plutôt d'appliquer un test d'hypothèse statistique comme le test de student (cf. section 2.3.5).

Cependant, les deux notions sont liées. En effet, pour un test statistique dont l'objectif est de déterminer si un paramètre inconnu est significativement différent de 0, avec un niveau de confiance de 95%, il est équivalent de déterminer un intervalle de confiance à 95% pour ce paramètre, et de vérifier si cet intervalle contient 0 ou non [CH79].

Notre situation est néanmoins quelque peu différente. Le test que l'on désire effectuer ne concerne pas une unique estimation : nous disposons de deux estimations distinctes, et nous voulons savoir si l'une est significativement supérieure à l'autre, pour un niveau de confiance arbitraire c. Dans ce cas, utiliser les intervalles de confiance à $(1-c)\%$ de ces estimations, et vérifier s'ils se superposent ou non, n'est pas équivalent à mener un test statistique avec le même niveau de confiance. En effet, la superposition des intervalles de confiance est une méthode plus conservatrice : il se peut que l'on obtienne que les deux quantités d'intérêt ne sont pas significativement différentes, alors qu'un test statistique aurait pu conclure l'inverse [WH02].

En pratique, cela signifie que les résultats des comparaisons obtenus correspondent à un niveau de confiance supérieure à 95%. Knol et. al. [KPG11] ont établi les calculs nécessaires pour connaître le véritable niveau de confiance correspondant à une comparaison menée sur la base de la superposition de deux intervalles de confiance, lesquels ont été obtenus pour un niveau de confiance de départ de 95%, et à partir d'estimations provenant de variables normales :

$$1 - c = 2 \times \Phi\left(- 1.96 \times \frac{1+\rho}{\sqrt{1 - 2\gamma\rho + \rho^2}} \right) \tag{6.34}$$

Cette formule met en relation le véritable niveau de confiance c avec le coefficient de corrélation γ des deux variables d'intérêt, ainsi que ρ, le quotient entre leurs écarts-types. La fonction Φ est la fonction de répartition de la loi normale centrée réduite. Par exemple, si l'on suppose que les deux variables sont indépendantes, i.e. $\gamma = 0$, et ont les mêmes écarts-types, i.e. $\rho = 1$, on obtient que $c \approx 99.5\%$. La constante 1.96 est simplement la valeur de z correspondant au niveau de confiance initialement choisi pour les intervalles de confiance. Il est donc possible, en connaissant les valeurs de γ et ρ, d'ajuster c à la valeur désirée en choisissant le niveau de confiance initiale de façon appropriée, modifiant ainsi la constante 1.96. Par exemple, pour obtenir $c = 95\%$, lorsque $\gamma = 0$ et $\rho = 1$, z doit valoir 1.3859, ce qui implique de calculer des intervalles de confiance à 16.6% de risque.

Pour nos expériences, il n'est cependant pas nécessaire d'établir une comparaison à 95%. L'essentiel est que pour un couple de classifieurs donné, le niveau de confiance soit le même pour chaque méthode de comparaison appliquée sur ce couple, ce qui est précisément le cas.

La méthode prudente-v1

Les méthodes désignées par prudentes suivent le même principe :
- mesurer la performance des classifieurs sur les instances de test bruitées ;
- considérer la situation de comparaison [15] qui défavorise le plus le classifieur établi ; nous appelons cette situation « le pire des cas » ;
- calculer les biais des estimateurs des deux classifieurs selon cette situation ;
- corriger les intervalles de confiance par les valeurs opposées de ces biais, de sorte à compenser le désavantage hypothétique attribué au classifieur établi ;
- comparer les classifieurs via leurs intervalles de confiance corrigés.

Une fois le pire des cas identifié et supposé, le principe d'une méthode prudente est donc simplement de corriger les intervalles de confiance des estimations obtenues selon le biais de chaque estimateur. La comparaison des deux classifieurs fait alors usage des intervalles de confiance corrigés.

Le pire des cas (prudente-v1)

Ce qui différencie les deux méthodes prudentes, c'est le pire des cas qu'elles considèrent. En ce qui concerne la méthode **prudente-v1**, le pire des cas correspond à la situation suivante :
- le biais de l'estimateur de la performance de $\mathcal{C}_1$ atteint sa valeur *minimale*, e.g. $-\mathbb{P}(N)$ pour la mesure de la justesse ; concrètement, cela signifie que $\mathcal{C}_1$ prédit *la vraie classe* de toutes les instances bruitées ;

15. la combinaison entre bruit d'annotation existant et prédictions des classifieurs

— le biais de l'estimateur de la performance de $\mathcal{C}_2$ atteint sa valeur *maximale*, e.g. $+\mathbb{P}(N)$ pour la mesure de la justesse ; concrètement, cela signifie que $\mathcal{C}_2$ prédit *l'étiquette* de toutes les instances bruitées ;

Dans le cadre de la méthode **prudente-v1**, nous nous servons donc d'un encadrement du biais de l'estimateur de performance, pour définir le pire des cas : ce dernier correspond concrètement au cas où le biais de l'estimateur de performance du classifieur établi $\mathcal{C}_1$ est le plus petit possible, tandis que celui du classifieur candidat $\mathcal{C}_2$ est le plus grand possible. En ce qui concerne la mesure de la justesse, un encadrement valable est simplement $[-\mathbb{P}(N), \mathbb{P}(N)]$.

L'application de cette méthode nécessite par conséquent d'avoir une idée de la valeur de certaines quantités pouvant être extraites de la matrice de transition de bruit, e.g. $\mathbb{P}(N)$ lorsque la mesure de performance est la justesse. Pour cela, on peut estimer approximativement cette matrice à partir de l'ensemble de données à disposition. En effet, comme nous l'avons mentionné en section 6.5.2, il est possible de faire annoter par de nouveaux annotateurs experts une partie des instances, de sorte à obtenir des annotations de qualité, et de confronter les nouvelles et anciennes annotations pour construire une matrice de transition de bruit. Cette matrice serait exacte pour le sous-ensemble d'instances ayant bénéficié d'une nouvelle annotation, et permettrait d'avoir une approximation de la matrice de transition de bruit pour l'ensemble complet.

Exemple d'application de la méthode prudente-v1

Pour la méthode **prudente-v1**, voici un exemple du déroulement de la procédure de comparaison, lorsque la mesure de performance choisie est la justesse. Nous disposons d'un couple de classifieurs $(\mathcal{C}_1, \mathcal{C}_2)$, nous mesurons leurs justesses $Q_1 = 0.72$ et $Q_2 = 0.78$ sur 1000 instances de test, et calculons leurs intervalles de confiance à 5% de risque i^1_{conf} et i^2_{conf} :

$$i^1_{conf} = [0.696, 0.743] i^2_{conf} = [0.758, 0.801]$$

Dans la section 6.3.1, nous avons obtenu l'expression du biais de l'estimateur de la justesse, que nous rappelons ici :

$$biais(\hat{q}) = \mathbb{P}(N)(F_n - F_r)$$

Ce biais dépend de trois quantités : la fraction globale d'exemples bruités $\mathbb{P}(N)$, ainsi que F_n et F_r. Les deux dernières quantités étant dépendantes du classifieur considéré, un encadrement grossier du biais d'évaluation est donc $[-\mathbb{P}(N), \mathbb{P}(N)]$.

Nous avons donc besoin d'une estimation de $\mathbb{P}(N)$, pouvant être obtenue par exemple à partir de la matrice de transition de bruit. En accord avec le principe d'évaluation à biais maximal, **prudente-v1** suppose la situation où l'estimation de la justesse du classifieur établi $\mathcal{C}_1$ a été défavorisée au maximum par le bruit d'annotation, i.e. le biais de l'estimateur de la justesse de $\mathcal{C}_1$ vaut $-\mathbb{P}(N)$, tandis que celui du classifieur candidat $\mathcal{C}_2$ vaut $\mathbb{P}(N)$. Ainsi, dans le calcul des intervalles de confiance associés, i^1_{conf} est corrigé en lui ajoutant $\mathbb{P}(N)$, et de même i^2_{conf} est corrigé en lui retranchant $\mathbb{P}(N)$. La comparaison du couple $(\mathcal{C}_1, \mathcal{C}_2)$ s'effectue selon ces intervalles de confiance corrigés.

Supposons que l'ensemble de test contiennent en réalité 3% de bruit d'annotation. Dans ce cas, il se peut que les estimations faites par la méthode **classique** soient en fait biaisées d'une

valeur d'au plus 0.03. Ainsi, les intervalles de confiance suivants auraient éventuellement pu être obtenus en ayant accès aux instances non bruitées, ou par la méthode **référence**, si l'on se place dans le cadre de nos expériences :

$$i^1_{conf}(+0.015) = [0.711, 0.758]$$
$$i^2_{conf}(-0.005) = [0.753, 0.796]$$

En ce qui concerne la méthode **prudente-v1**, nous estimons approximativement le taux de bruit à 5%. Les intervalles de confiance obtenus après correction seraient (cf. équation 6.30 en section 6.4) alors :

$$i^1_{conf}(+0.05) = [0.746, 0.793]$$
$$i^2_{conf}(-0.05) = [0.708, 0.751]$$

Selon les intervalles de confiance obtenus par chaque méthode :
- **classique** conclut que $\mathcal{C}_1 \triangleleft_\Psi \mathcal{C}_2$;
- **prudente-v1** conclut que $\mathcal{C}_1 \triangle_\Psi \mathcal{C}_2$ car les intervalles obtenus se superposent ;
- **référence** décide également que $\mathcal{C}_1 \triangle^* \mathcal{C}_2$, bien que la superposition des intervalles soient moins flagrante que pour **prudente-v1**.

Par conséquent, **classique** commet dans ce cas précis une erreur de type I, alors que **prudente-v1** l'évite.

La méthode prudente-v2

La méthode **prudente-v2** consiste à prendre un point de vue moins stricte que pour **prudente-v1**. En ce qui concerne cette dernière, il est peu probable que le pire des cas qu'elle considère ait vraiment lieu. En effet, ce cas n'est même pas possible si les désaccords entre les deux classifieurs ne couvrent pas au moins l'ensemble des instances bruitées.

Le pire des cas (prudente-v2)

Le pire des cas de la méthode **prudente-v2** est moins extrême que pour **prudente-v1**. De plus, là où **prudente-v1** considère d'abord un encadrement du biais, qui correspond en conséquence à certaines contraintes sur les désaccords entre les classifieurs, ainsi que les erreurs d'annotation des instances de test, **prudente-v2** fonctionne de la manière inverse :
- nous regardons sur quelles instances les deux classifieurs sont en désaccords ;
- nous supposons que la véritable classe de ces instances, en pratique inconnue, est en accord avec la prédiction du classifieur établi $\mathcal{C}_1$;
- toutes les autres instances sont supposées non bruitées ;

De cette manière, nous construisons artificiellement un « pire des cas », où $\mathcal{C}_1$ est sous-estimé par rapport à sa véritable performance, et où $\mathcal{C}_2$ est surestimé. Cela revient à construire une vérité-terrain absolue hypothétique, qui favorise $\mathcal{C}_1$. Les biais de chaque estimateur peuvent alors être calculés de façon exacte, étant donné que lorsque la vérité-terrain absolue est supposée connue, les indicateurs F_c, F_n et F_r sont accessibles.

Voici concrètement comment les biais des estimateurs sont calculés, une fois que les performances des classifieurs ont été estimées :
- nous construisons tout d'abord la nouvelle annotation des instances :

1. nous définissons un taux de prudence pc, qui nous permet de choisir les $pc\%$ instances de test les plus difficiles selon notre classement de difficulté $\mathbb{K}$ (cf. section 6.5.2) ; le sous-ensemble $\mathbb{K}_{pc}$ que nous obtenons correspond aux instances pour lesquelles nous nous permettons de modifier l'annotation dans la nouvelle vérité-terrain ;

2. nous regardons ensuite les instances sur lesquelles $\mathcal{C}_1$ et $\mathcal{C}_2$ sont en désaccord : pour chacune de ces instances, si elle appartient à $\mathbb{K}_{pc}$, l'étiquette que nous lui attribuons dans la nouvelle vérité-terrain est la catégorie prédite par $\mathcal{C}_1$;

— la vérité-terrain ainsi obtenue est en faveur de $\mathcal{C}_1$; en supposant qu'elle correspond à la vérité-terrain absolue, nous nous plaçons dans une situation où le classifieur établi $\mathcal{C}_1$ est défavorisé par l'annotation bruitée de départ (sur laquelle nous avons initialement estimé la performance de nos classifieurs) ;
— comme nous disposons aussi bien de l'annotation bruitée que d'une annotation considérée parfaite, nous pouvons construire la matrice de transition de bruit correspondante, et estimer toutes les quantités nécessaires au calcul des biais.

Cette méthode est moins stricte que la méthode **prudente-v1**. En effet, la situation artificiellement construite dans ce qui précède est loin d'être le pire des cas que **prudente-v1** considère. Plus précisément, **prudente-v1** considère une situation où le classifieur établi est défavorisé au maximum, sans regarder si cela est possible par rapport aux désaccords existant entre les deux classifieurs. La méthode **prudente-v2** permet de considérer une situation défavorable au classifieur établi, mais compatible avec les désaccords des classifieurs.

De plus, en ajustant le paramètre pc, le pire des cas de **prudente-v2** peut être contrôlé de façon à être plus ou moins défavorable au classifieur établi. Si $pc = 100\%$, la situation considérée le défavorise au maximum, i.e. pour toutes les instances où il y a un désaccord, alors que si $pc = 0\%$, aucun désavantage n'est supposé pour le classifieur établi. Le cas échéant, la méthode **prudente-v2** coïncide avec la méthode **classique**.

Méthode de comparaison avec procédure de nettoyage

Enfin, au sujet des méthodes de comparaison faisant intervenir des procédures de nettoyage (cf. section 3.4.2), nous avons choisi d'implanter ces dernières de façon artificielle. Cela nous permet en effet de maintenir aisément l'aspect contrôlable et reproductible de nos expériences. En simulant le comportement de ces procédures, nous gardons la possibilité de contrôler leur qualité. Précisément, nous simulons une procédure de nettoyage par un processus aléatoire, parcourant l'ensemble de données, et ayant :

— une probabilité de détection réussie d'un exemple mal annoté, comprise entre 0 et 1 ;
— une probabilité de détection incorrecte d'un exemple bien annoté, comprise entre 0 et 1 ;
— et, pour simuler les procédures qui corrigent l'annotation des exemples détectés, une probabilité de correction réussie d'un exemple détecté, comprise entre 0 et 1 ; si la correction échoue, l'annotation de l'exemple concerné est choisi aléatoirement parmi les autres classes ;

Bien sûr, le fait qu'une détection ou qu'une correction soit réussie ou non est déterminé par rapport à l'annotation initiale de référence de CIFAR10. Nous mettons ainsi en place des procédures de nettoyage de qualités variées, certaines supprimant de l'ensemble de test les exemples détectés, d'autres tentant de les corriger, et pour chacune d'elle, nous appliquons la comparaison **classique** sur l'ensemble de test obtenu après application de la procédure.

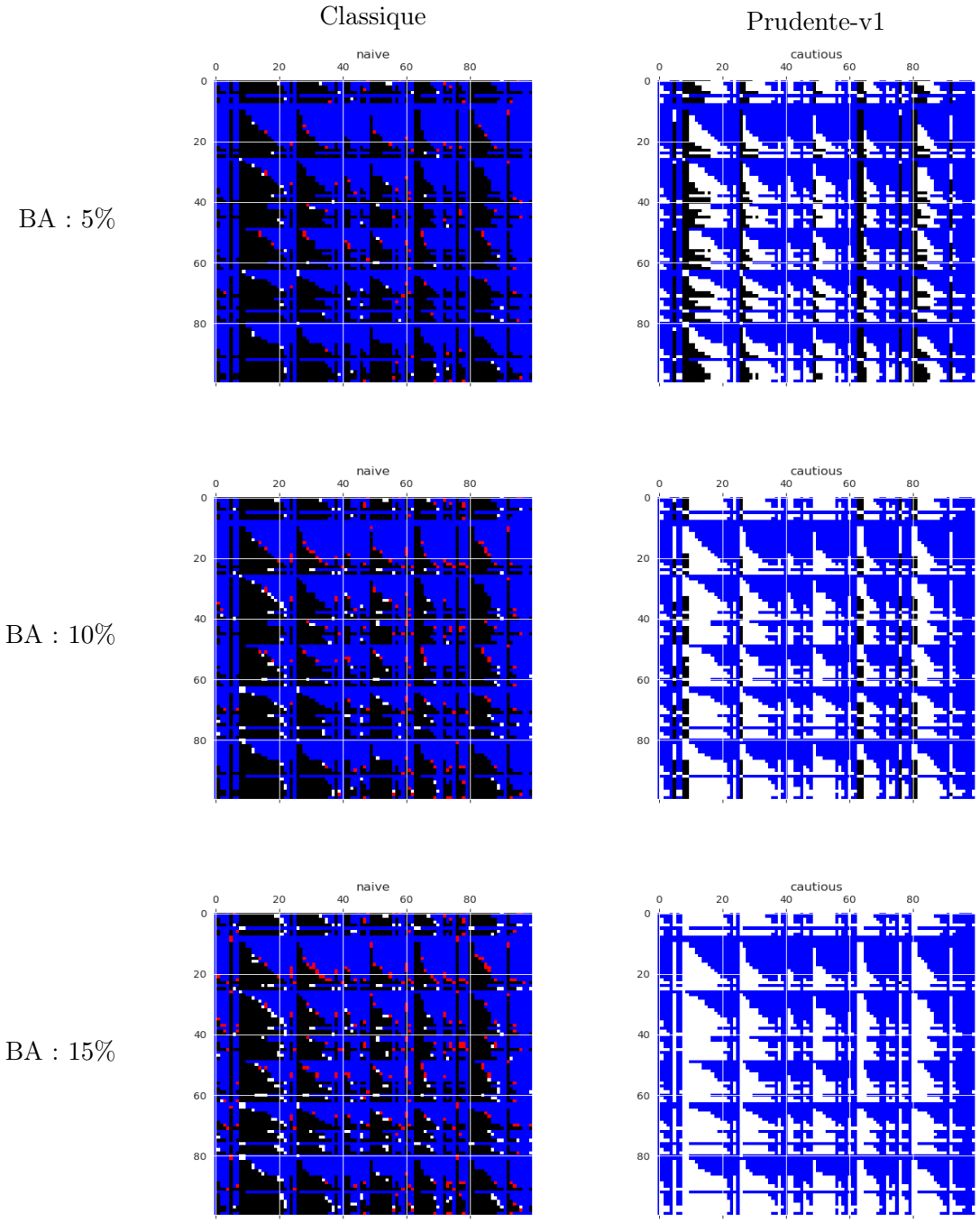


FIGURE 6.6 – Matrices de comparaison entre les méthodes **classique** et **prudente-v1** par rapport à la méthode de **référence**, pour 5%, 10% et 15% de bruit d'annotation.

6.5.5 Analyse des résultats

Pour résumer, notre expérience se déroule de la manière suivante :

— la comparaison de référence Ψ^* est d'abord effectuée, i.e. les classifieurs sont comparés deux à deux sur la version propre de CIFAR10 ;

— ensuite, pour chaque version bruitée de CIFAR10, chaque méthode de comparaison Ψ est

appliquée ; **prudente-v2** est appliquée avec un taux de prudence $pc = 10\%$;
- le résultat de chaque méthode Ψ est compilé dans une matrice de taille 100x100, de sorte à ce que la cellule (i, j) représente la différence entre $\Psi(\mathcal{C}_i, \mathcal{C}_j)$ et $\Psi^*(\mathcal{C}_i, \mathcal{C}_j)$; il y a 4 possibilités :
  - une erreur de type I est commise : code couleur *rouge* ;
  - une erreur de type II est commise : code couleur *blanc* ;
  - Ψ^* et Ψ choisissent le classifieur candidat $\mathcal{C}_j$: code couleur *noir* ;
  - Ψ^* et Ψ choisissent le classifieur établi $\mathcal{C}_i$: code couleur *bleu* ;
- la ligne i d'une matrice regroupe, pour la méthode Ψ correspondante, les valeurs $\Psi(\mathcal{C}_i, .)$ avec les 99 autres classifieurs, $\mathcal{C}_i$ *prenant le rôle du classifieur établi* ;
- une fois les matrices obtenues, les statistiques suivantes sont calculées pour chaque méthode de comparaison, excepté la méthode de référence :
  - le pourcentage de fois où Ψ commet une erreur de type I ;
  - le pourcentage de fois où Ψ commet une erreur de type II ;
  - le pourcentage de fois où Ψ est en accord avec Ψ^*.

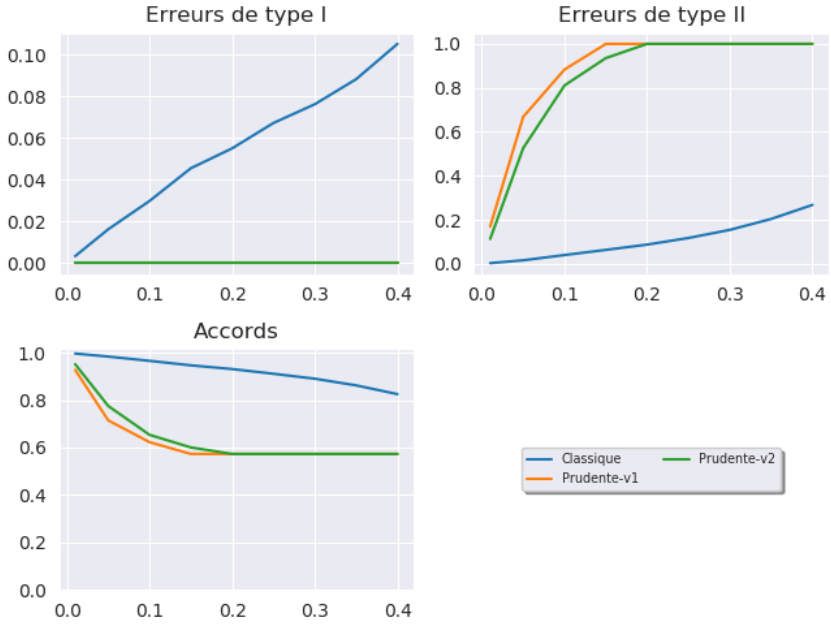


FIGURE 6.7 – Tracé de différentes statistiques pour chaque méthode de comparaison Ψ, en fonction du bruit d'annotation. Les statistiques considérées sont calculées en confrontant ces méthodes à la comparaison de référence Ψ^*.

Méthodes de comparaison prudentes

Nous présentons ici les résultats obtenus pour les méthodes **prudente-v1** et **prudente-v2**, uniquement pour la mesure de la justesse. La figure 6.6 montre les matrices obtenues pour les méthodes **classique** et **prudente-v1**, pour les version de CIFAR10 bruitées à 5%, 10% et 15%.

On observe que les méthode prudentes sont extrêmement strictes sur leur manière de juger. En effet, elles doutent beaucoup plus souvent de quel classifieur est le meilleur, ce qui correspond en pratique à obtenir des intervalles de confiance qui se superposent plus fréquemment. Cela les conduit à choisir le classifieur établi dans de nombreux cas. En particulier, on voit que lorsqu'on

considère un taux de bruit de 15%, **prudente-v1** ne se risque plus à choisir le classifieur candidat (codes couleur bleu et blanc pour la dernière matrice de la figure 6.6).

En conséquence, les méthodes prudentes ne commettent jamais d'erreurs de type I, mais en contrepartie, le nombre d'erreurs de type II est considérable. Même lorsque l'ensemble de test ne contient que 5% de bruit, le taux d'erreurs de type II des méthodes prudentes est supérieur à 50% (tableau 6.2). Cela signifie que, lorsque **référence** juge le classifieur établi meilleur, ces méthodes sont tout le temps d'accord, et lorsque **référence** juge le classifieur candidat meilleur, elles sont d'accord moins d'une fois sur deux.

TABLE 6.2 – Pourcentage d'erreurs de type I et II commises par les méthodes **classique**, **prudente-v1** et **prudente-v2**, ainsi que le taux d'accords avec la méthode de **référence**, pour 5% de bruit d'annotation.

-	Erreurs de type I	Erreurs de type II	Accords
classique	1.62%	1.58%	98.39%
prudente-v1	0.00%	66.82%	71.45%
prudente-v2	0.00%	52.61%	77.53%

La méthode **prudente-v2** est malgré tout plus souple que **prudente-v1** sur ce point. Tout d'abord, pour un ensemble de test à 5% de bruit, elle commet 52.61% d'erreurs de type II, là ou **prudente-v1** en commet 66.82% (tableau 6.2). Mais surtout, **prudente-v2** est ici appliquée avec une valeur du taux de prudence $pc = 10\%$. Comme nous l'avons précédemment expliqué, sa souplesse peut être diminuée ou augmentée en configurant ce taux de prudence. Le diminuer permettrait ainsi de faire en sorte que le pourcentage d'erreurs de type II de **prudente-v2** soit beaucoup plus faible. Cependant, avec une valeur de pc trop petite, le pourcentage d'erreurs de type I peut également ne plus être nul. Le choix du taux de prudence pc se faisant pour l'instant de manière ad hoc, tomber sur la valeur idéale relèverait plutôt du hasard.

La figure 6.7 montre un tracé des statistiques d'intérêt en fonction de la quantité d'instances de test bruitées. Bien que globalement, la méthode **classique** est largement plus souvent d'accord avec la comparaison de référence, les erreurs de type I ne sont entièrement évitées que par les méthodes prudentes. Par conséquent, selon la nécessité, propre à l'application, d'éviter de remplacer à tort un classifieur établi, cette contrepartie peut être acceptable, en particulier pour la méthode **prudente-v2**, avec son taux de prudence pouvant être configuré manuellement.

Méthodes de comparaison avec procédures de nettoyage

Nous présentons ici les résultats obtenus avec les méthodes de comparaison incluant l'application préalable d'une procédure de nettoyage des instances. Ces procédures de nettoyage détectent avec plus ou moins de réussite les instances bruités dans l'ensemble de données, et les retirent, ou alors corrigent leurs annotations, là aussi avec un taux de succès variable.

Le tableau 6.3 montre les statistiques d'intérêt obtenues sur l'ensemble de test bruité à 5%, pour chacune de ces méthodes, ainsi que pour la méthode **classique**, qui n'implique pas de procédures de nettoyage. On peut remarquer différentes choses :

— un taux d'échecs trop grand, e.g. $e \geq 0.4$, augmente le nombre d'erreurs des deux types de façon plus importante qu'un faible taux de succès, e.g. $s = 0.4$;
— les procédures de nettoyage qui corrigent les annotations plutôt que de simplement retirer les instances détectées, i.e. le paramètre c est défini, permettent de diminuer le taux

TABLE 6.3 – Statistiques d'intérêt obtenues pour les méthodes incluant une procédure de nettoyage. Une procédure de nettoyage est notée NETT, et est définie par les paramètres s pour taux de succès, e pour taux d'échecs, et éventuellement c pour taux de bonnes corrections. Les lignes en vert indiquent les fois où les erreurs des deux types sont plus faibles que dans le cas de la méthode **classique**. Les lignes en orange indiquent les cas représentant le meilleur compromis entre un faible taux d'erreurs de type I et un taux d'erreurs de type II acceptable.

	Erreurs de type I	Erreurs de type II	Accords
classique	1.62%	1.58%	98.39%
NETT($s = 0.75$, $e = 0.25$)	1.59%	1.32%	98.53%
NETT($s = 0.75$, $e = 0.25$, $c = 0.25$)	0.34%	5.79%	97.33%
NETT($s = 0.75$, $e = 0.25$, $c = 0.5$)	0.56%	3.90%	98.01%
NETT($s = 0.75$, $e = 0.25$, $c = 0.75$)	0.51%	2.34%	98.71%
NETT($s = 0.6$, $e = 0.1$)	1.53%	1.02%	98.69%
NETT($s = 0.6$, $e = 0.1$, $c = 0.25$)	0.71%	2.60%	98.48%
NETT($s = 0.6$, $e = 0.1$, $c = 0.5$)	0.76%	2.06%	98.69%
NETT($s = 0.6$, $e = 0.1$, $c = 0.75$)	0.86%	1.32%	98.94%
NETT($s = 0.9$, $e = 0.4$)	2.20%	1.68%	98.02%
NETT($s = 0.9$, $e = 0.4$, $c = 0.25$)	0.51%	9.22%	95.77%
NETT($s = 0.9$, $e = 0.4$, $c = 0.5$)	0.32%	5.65%	97.40%
NETT($s = 0.9$, $e = 0.4$, $c = 0.75$)	0.25%	2.77%	98.68%
NETT($s = 0.6$, $e = 0.4$)	2.35%	2.32%	97.67%
NETT($s = 0.6$, $e = 0.4$, $c = 0.25$)	0.34%	9.39%	95.80%
NETT($s = 0.6$, $e = 0.4$, $c = 0.5$)	0.60%	6.08%	97.06%
NETT($s = 0.6$, $e = 0.4$, $c = 0.75$)	0.37%	3.05%	98.48%
NETT($s = 0.9$, $e = 0.6$)	3.02%	3.00%	96.99%
NETT($s = 0.9$, $e = 0.6$, $c = 0.25$)	0.18%	16.58%	92.82%
NETT($s = 0.9$, $e = 0.6$, $c = 0.5$)	0.39%	8.87%	95.99%
NETT($s = 0.9$, $e = 0.6$, $c = 0.75$)	0.42%	4.28%	97.93%
NETT($s = 0.4$, $e = 0.1$)	1.50%	1.30%	98.59%
NETT($s = 0.4$, $e = 0.1$, $c = 0.25$)	0.83%	2.65%	98.39%
NETT($s = 0.4$, $e = 0.1$, $c = 0.5$)	0.88%	2.18%	98.57%
NETT($s = 0.4$, $e = 0.1$, $c = 0.75$)	1.09%	1.51%	98.73%

d'erreurs de type I, même pour un faible taux de bonne correction, mais augmentent en général le taux d'erreurs de type II de manière significative ;

— en ce qui concerne les erreurs de type I, le fait de corriger les annotations contrebalance l'impact négatif d'un grand taux d'échecs : les méthodes avec une procédure de nettoyage sans correction et avec $e \geq 0.4$ font jusqu'à deux fois plus d'erreurs de type I que la méthode **classique**, tandis que lorsque ces procédures de nettoyage sont appliquées avec $c \geq 0.25$, le taux d'erreurs de type I est au moins trois fois plus petit.

Cela indique que lorsque l'objectif est de comparer des classifieurs, et que l'on désire nettoyer l'ensemble de test d'éventuels instances mal annotées, il est préférable de favoriser un faible taux d'échecs, au détriment du taux de succès de la procédure de nettoyage. De cette manière, les erreurs de type I et de type II sont censées diminuer. De plus, si l'objectif est de minimiser au maximum les erreurs de type I, il vaut mieux que la procédure de nettoyage tente de corriger les annotations. Dans le cas où les erreurs de type II sont tout de même importantes à éviter, il est

préférable de simplement retirer de l'ensemble de test les instances détectées par la procédure de nettoyage.

Mesures de précision et de rappel

Nous présentons enfin les résultats obtenus en ce qui concerne les mesures de précision et de rappel. A cette fin, nous nous intéressons uniquement aux méthodes de comparaison **classique** et **prudente-v1**.

TABLE 6.4 – Résultats des méthodes **classique** et **prudente-v1** lorsque la précision par rapport à une classe (ici la classe 0) est utilisée comme mesure de performance, avec 5% de bruit d'annotation.

	Erreurs de type I	Erreurs de type II	Accords
classique	1.24%	1.50%	98.64%
prudente-v1	0.00%	11.94%	94.36%

TABLE 6.5 – Résultats des méthodes **classique** et **prudente-v1** lorsque le rappel par rapport à une classe (ici la classe 0) est utilisée comme mesure de performance, avec 5% de bruit d'annotation.

	Erreurs de type I	Erreurs de type II	Accords
classique	1.30%	1.35%	98.68%
prudente-v1	0.00%	30.51%	85.57%

Les tableaux 6.4 et 6.5 montrent les pourcentages d'erreurs des deux types, ainsi que le taux d'accord avec la méthode de référence, lorsque les mesures utilisées sont la précision et le rappel, calculées ici par rapport à la classe 0. On voit ainsi que, contrairement au cas où la mesure utilisée est la justesse, les taux d'erreurs de type II ne sont pas si élevés pour la méthode **prudente-v1**, et n'atteignent que 11.94% pour la mesure de la précision en particulier.

Ces deux mesures sont donc moins influencées que la justesse par la présence de bruit d'annotation. En particulier, lorsqu'on veut comparer des classifieurs avec des instances mal annotées, avec l'objectif d'éviter les erreurs de type I grâce à une méthode de comparaison prudente, la précision est un bon indicateur de performance, étant donné que le biais de son estimateur est suffisamment faible pour maintenir un taux d'erreurs de type II bas.

6.5.6 Perspectives d'amélioration

Notre cadre applicatif présente plusieurs pistes d'amélioration. La première est évidente : bien que nous ayons introduit un bruit d'annotation dépendant des caractéristiques des instances, de sorte à ce qu'il soit le plus réaliste possible, ce bruit reste artificiel. Il serait intéressant d'évaluer la capacité de notre procédure d'introduction de bruit artificiel à générer un bruit d'annotation réaliste.

De plus, par soucis d'économie de temps, nous n'avons pu réaliser nos expériences sur d'autres ensembles de données, ou avec un autre classement de difficulté pour l'annotation des instances. Enfin, et pour la même raison, dans notre protocole de comparaison, les classifieurs que nous avons chacun entraînés sur une version bruitée précise de CIFAR10, sont ensuite évalués sur d'autres

versions contenant un taux de bruit différent. En situation réelle, les ensembles d'entraînement et de test des classifieurs contiennent le même type de bruit d'annotation, et c'est précisément dans ce cas que la performance d'un classifieur peut être surestimée alors qu'il reproduit les erreurs d'annotation des annotateurs. Bien que les versions bruitées que nous générons pour CIFAR10 contiennent un bruit d'annotation similaire, l'idéal serait d'entraîner un groupe de classifieurs à comparer différent pour chaque version bruitée de CIFAR10.

Nous pouvons aussi mentionné un point pouvant paraître problématique : considérer la comparaison de référence comme étant parfaite et non biaisée est inexacte. D'une part, rien ne nous assure que l'annotation de CIFAR10 ne contient à l'origine aucune erreur, et d'autre part, le bruit d'annotation n'est pas l'unique source de biais dans l'évaluation de performance de classifieurs. En effet, le manque de variabilité dans les exemples, ou la surreprésentation d'un aspect arbitraire, peuvent en être des causes tout aussi importantes, si ce n'est plus [RH05]. Par exemple, le fait que les images soient en basse résolution (32x32 pixels), ou encore que chaque image ne contienne qu'une seule entité d'une des 10 classes, est une source de biais par rapport à l'apprentissage et à l'évaluation de la tâche de reconnaissance d'objet dans une scène quelconque. Cependant, si l'objectif de l'expérience est d'évaluer l'impact du bruit d'annotation sur plusieurs méthodes de comparaison, en les appliquant dans des situations où le nombre d'instances mal annotées croît, il n'est pas nécessaire de s'intéresser à la validité initiale de l'ensemble de données utilisé. En effet, du moment que l'on dispose d'un ensemble représentant un concept arbitraire, peu importe si ce concept ne représente pas exactement la réalité. L'important est l'analyse relative des méthodes de comparaison appliquées lorsque ce concept, même fictif, est ou n'est pas perturbé par la présence de bruit d'annotation.

Pour finir, les calculs que nous effectuons en ce qui concerne le biais de la mesure d'évaluation pour la méthode **prudente-v1** nous demandent de connaître certaines quantités présentes dans la matrice de transition de bruit, dont le taux de bruit d'annotation global dans l'ensemble de test. Cette matrice n'étant pas connue en pratique, il serait nécessaire, dans un cas réaliste, de l'estimer au préalable, par exemple en utilisant une petite partie des données que l'on transmettrait à des experts pour vérifier les annotations, ce qui permettrait de la construire à partir des erreurs ainsi repérées et de l'extrapoler à l'ensemble de données complet. Cependant, cela concerne uniquement la méthode **prudente-v1** : **prudente-v2** et les méthodes axées sur l'application d'une procédure de nettoyage de données n'en ont pas besoin.

6.6 Conclusion

Cette expérience nous a permis d'étudier l'impact du bruit d'annotation sur la comparaison de classifieurs, dans le contexte particulier où l'un des deux classifieurs est à privilégier. Le bruitage de l'annotation a été réalisé de sorte à dépendre des caractéristiques des instances. Nous avons entraîné une centaine de classifieurs aux comportements variés face aux instances de CIFAR10, comportements que nous avons qualifiés à l'aide des indicateurs F_c, F_n et F_r. Ensuite, pour la comparaison deux à deux de ces classifieurs, nous nous sommes intéressés à deux types d'erreurs de jugement envisageables, et avons observé les apports de différentes méthodes de comparaison sur la base de ces erreurs. En résumé, lorsqu'on compare un classifieur établi avec un autre classifieur sur des instances mal annotées, qui impliquent un biais dans l'estimation de performance :

— la méthode **prudente-v1** se sert des valeurs limites que peut atteindre ce biais. Elle suppose d'abord que le classifieur établi est lésé au maximum par rapport à l'autre classifieur, i.e. la performance mesurée pour le classifieur établi est plus basse qu'en réalité tandis que

celle mesurée pour l'autre est plus haute. Elle compense alors cet handicap en décalant adéquatement les intervalles de confiance obtenus pour chacune des mesures. Cette méthode permet d'éviter constamment les erreurs de type I, mais fait énormément d'erreurs de type II (environ 66% d'erreurs avec un ensemble de test à 5% de bruit) comparé à la méthode **classique** (1.58% d'erreurs dans le même contexte).

— La méthode **prudente-v2** tente de compenser de manière plus fine l'éventuel handicap du classifieur établi, en se concentrant uniquement sur les instances de désaccord entre les deux classifieurs, où plutôt sur une fraction pc de ces instances. Le taux de prudence pc permet de contrôler la finesse de la méthode. Pour $pc = 10\%$, le nombre d'erreurs de type II reste néanmoins très élevé, d'une valeur de 52.61%. Ces deux méthodes prudentes sont cependant les seuls à être en mesure d'éviter totalement les erreurs de type I.

— Les méthodes de comparaison avec application d'une procédure de nettoyage mesure la performance des classifieurs en ayant nettoyé l'ensemble de test au préalable, i.e. retiré ou corrigé les instances de test détectées par la procédure de nettoyage. Dans l'objectif de comparer nos deux classifieurs en évitant les erreurs de type I et II, le taux d'échecs de détection, i.e. le nombre de fois où la procédure détecte des instances qui ne sont pas bruitées, doit être minimisé, même si cela implique d'affaiblir le taux de succès. De plus, appliquer une procédure corrigeant les annotations permet de réduire d'avantage le nombre d'erreurs de type I, mais augmentent le nombre d'erreurs de type II. Ces méthodes sont préférables aux méthodes prudentes lorsqu'il n'est pas nécessaire d'atteindre 0% d'erreurs de type I, étant donné qu'elles commettent beaucoup moins d'erreurs de type II. cependant, elles demandent d'avoir une procédure de nettoyage fonctionnant correctement.

— Une autre possibilité est de mesurer la performance par la précision ou le rappel, et non la justesse. En effet, le bruit d'annotation biaise de manière moins importante les estimateurs de ces deux mesures, en particulier celui de la précision. Ainsi, pour 5% de bruit, là où la méthode **prudente-v1** commet plus de 60% d'erreurs de type II avec la justesse, elle n'en commet plus que 11.94% avec la précision par rapport à la classe 0. Pour l'application des méthodes prudentes, il peut donc être intéressant de choisir des mesures comme la précision, moins influencées que la justesse par la présence d'instances mal annotées.

Chapitre 7

Cas applicatif : élagage d'un réseau de neurones profond en présence de bruit d'annotation

Sommaire

Lors des expériences présentées en section 6.5, nous avons eu besoin d'entraîner un grand nombre de classifieurs aux comportements variés, d'une part pour le bon fonctionnement de notre procédure d'introduction de bruit via des annotateurs artificiels, d'autre part pour construire différents classifieurs à comparer. Nous avons pour cela fait usage de réseaux de neurone à

la structure complexe, appelés des Fabriques de Réseaux Convolutifs (FRC). Pour varier les classifieurs obtenus à partir de cette structure, nous avons initialisé puis élagué aléatoirement [16] chaque FRC, avant de procéder à l'entraînement sur les données bruitées.

En s'intéressant plus particulièrement au processus d'élagage, il apparaît que, là aussi, le bruit d'annotation a potentiellement un impact, problème non répertorié dans la littérature. En effet, l'élagage de réseaux de neurones permet d'identifier les parties d'un réseau les moins utiles à la bonne réalisation de la tâche, de sorte à les retirer du réseau pour réduire sa taille sans trop dégrader ses performances. Pour identifier ces parties, on utilise principalement deux types d'indicateurs : le premier se calcule uniquement à partir des poids du réseau, tandis que le second a en plus besoin d'instances annotées. La question que l'on se pose est donc : quelle est l'impact du bruit d'annotation sur l'efficacité de ces indicateurs ?

Cette étude entre également en perspective de réflexions au sujet du principe d'évaluation non supervisée (cf. section 5.2.3). Il est en effet intéressant de se demander si l'évaluation de classifieur peut être réalisée par l'utilisation d'indicateurs indépendant des annotations des instances, i.e. non supervisé. Dans cette expérience, l'un des deux indicateurs d'élagage fait office d'indicateur non supervisé, car il se calcule à partir des poids du réseau. Vérifier s'il est effectivement insensible à la présence de bruit d'annotation permettrait donc d'avoir un début de réponse à la question précédente.

Dans ce chapitre, nous compilons donc les contributions suivantes :

- présentation d'un algorithme d'élagage adapté à la structure particulière d'une FRC, qui agit à deux échelles différentes :
  - une échelle à bas niveau, où les poids sont élagués ;
  - et une échelle à plus haut niveau, où les liens (cf. 7.4.1) sont élagués ;
- étude de l'impact du bruit d'annotation sur l'efficacité de deux indicateurs, l'un supervisé, l'autre non supervisé, tout deux étant habituellement utilisés pour l'élagage de réseaux de neurone.

7.1 Introduction

Malgré la haute efficacité des modèles issus de l'apprentissage profond, trouver quelle structure de réseau est la mieux adaptée à un problème d'apprentissage précis peut se révéler laborieux. De nombreux hyperparamètres entrent en jeu, et déterminer la meilleure combinaison est chronophage et demande de l'expérience et de l'intuition. Pour rendre plus simple cette étape de sélection du modèle, Verbeek et. al. ont proposé une structure complexe, nommée Fabrique de Réseaux Convolutifs (FRC) [SV16]. Cette structure contient un grand nombre de réseaux de neurones convolutifs (RNC) classiques, organisés en treillis (Figure 7.3), de sorte à ce qu'un chemin linéaire commençant depuis l'entrée de la structure, et allant jusqu'à sa sortie, corresponde à un RNC parmi une multitude de possibilités. Une FRC peut être définie par seulement deux hyperparamètres, et entraînée par la méthode classique de rétro-propagation du gradient. Par conséquent, l'étape de sélection de modèle est grandement facilitée, étant donné que l'entraînement d'une FRC correspond à entraîner en même temps de nombreux RNC, et que le nombre d'hyperparamètres est réduit.

Cependant, une FRC contient tout de même une quantité importante de poids, bien que cela reste insignifiant par rapport aux nombres de poids de la totalité des RNC contenus dans la struc-

16. Bien sûr, élaguer un réseau de neurone ne se fait pas de façon aléatoire si l'on ne veut pas trop dégrader la performance atteignable par le réseau obtenu. Dans nos expériences de la section 6.5, une haute performance des classifieurs entraînés n'était pas nécessaire.

ture, lorsqu'ils sont pris individuellement. Par conséquent, l'élagage est une pratique intéressante pour cette structure. La recherche sur l'élagage de réseaux a pris de l'importance ces dernières années. Cette pratique a été d'abord étudié dans [LDS90 ; HP89 ; HS93]. L'élagage était essentiellement appliqué à des réseaux entraînés au préalable, puis suivi d'une phase d'entraînement supplémentaire, communément appelée *fine-tuning* en anglais, pour adapter les valeurs des poids restants. Plus récemment, de nouvelles approches ont été proposées, entre autres des procédures itératives alternant entre une phase d'entraînement et une phase d'élagage [CI18], ou encore des techniques permettant d'élaguer un réseau directement après l'initialisation aléatoire de ses poids [FC18 ; Zho+19 ; Ram+19], de sorte à ce que l'apprentissage soit dès le départ effectué sur une petite structure. En ce qui concerne l'élagage de FRC, Verbeek et. al. montrent comment on peut élaguer une FRC déjà entraînée sans trop perdre en performance, mais cela implique une phase d'apprentissage très longue, le nombre de poids à apprendre étant considérable. Nous n'avons pas connaissances d'études plus poussées ayant été menées sur ce sujet.

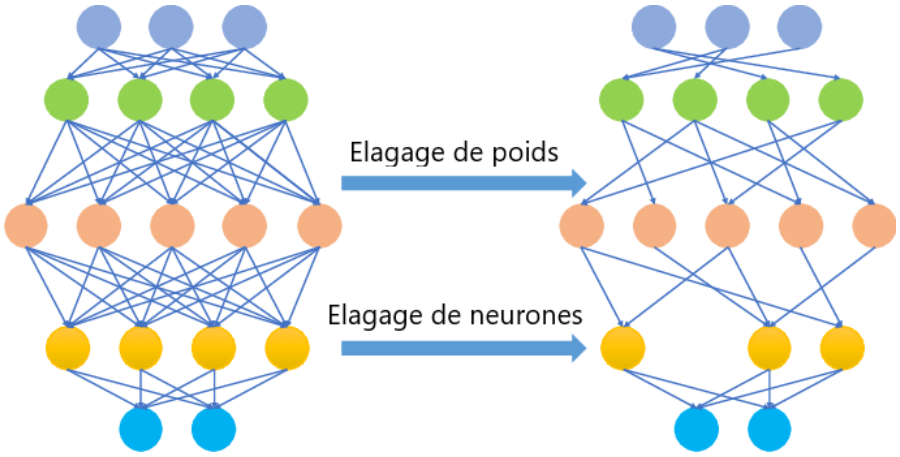


FIGURE 7.1 – Principe de l'élagage de réseaux de neurones, illustrés sur un perceptron multicouches [Sak20]. L'élagage de neurones correspond simplement à l'élagage de tous les poids en lien avec les neurones concernés.

Dans ce chapitre, nous présentons dans un premier temps nos travaux sur l'élagage de FRC, et nous montrons en particulier comment il est possible d'élaguer petit à petit la structure de manière itérative, ou alors suffisamment tôt lors de l'apprentissage, en subissant une perte de performance minime. Ainsi, cela permet de réduire considérablement la complexité de la structure en terme de poids à entraîner.

De plus, notre méthode d'élagage peut s'appliquer aussi bien aux poids de la structure qu'à des éléments de taille plus importante, appelés les liens de la FRC (cf. section 7.3). Non seulement l'élagage de lien est adapté pour facilement visualiser les chemins maintenus au sein de la structure post-élagage, mais surtout, nous montrons que combiner l'élagage de poids et de liens permet d'obtenir de meilleurs résultats.

Nous étudions enfin l'impact de la qualité de l'annotation des instances sur le processus d'élagage. Nous présentons cet impact sur deux critères d'élagage que nous détaillerons dans la suite, le premier étant a priori indépendant des instances, car il n'en a pas besoin pour déterminer ce qui doit être élaguer dans le réseau, alors que l'autre s'en sert.

Le reste du chapitre est organisé de la manière suivante : la section 7.2 introduit le principe de l'élagage ; la section 7.3 présente en détails le fonctionnement d'une FRC ; nous expliquons le principe des algorithmes d'élagage que nous avons appliqués sur nos FRC en section 7.4 ; la

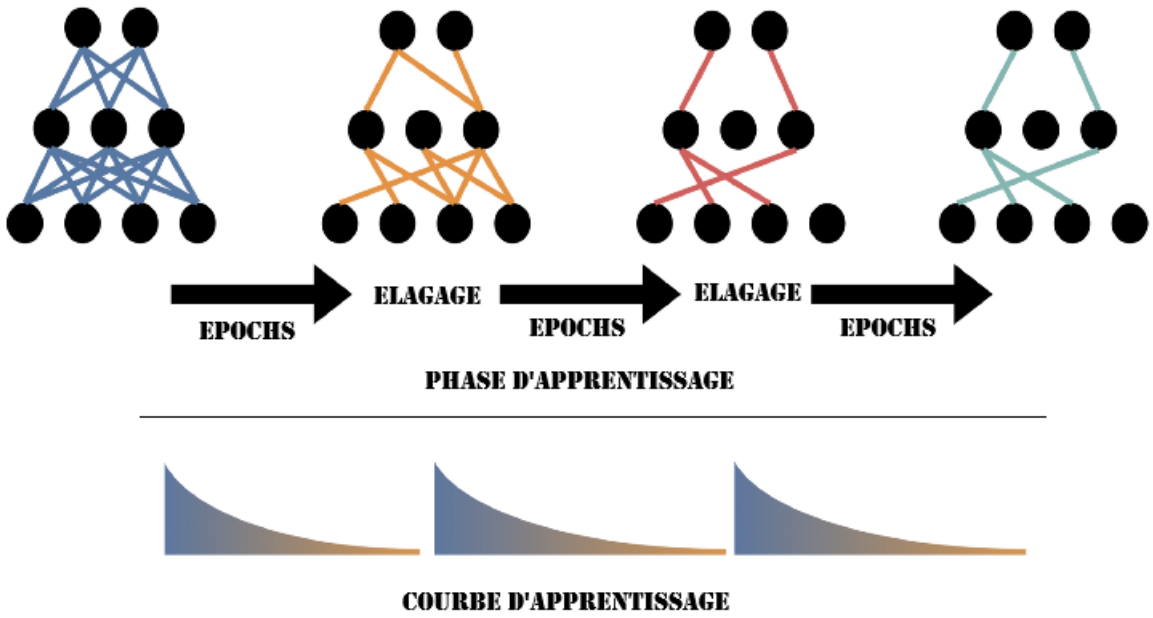


FIGURE 7.2 – Principe du processus d'élagage, appliqué ici de manière itérative. Le réseau, intact, est tout d'abord entraîné pendant quelques epochs, puis une étape d'élagage supprime quelques poids. Le processus se répète alors plusieurs fois, jusqu'à avoir atteint la taille désirée pour le réseau. Les dégâts de chaque étape d'élagage sont visibles sur la courbe d'apprentissage, par l'augmentation correspondante du taux d'erreurs du réseau.

section 7.5 détaille notre approche pour l'implantation de la structure de FRC et des algorithmes d'élagage en PyTorch ; la section 7.6 présente notre protocole expérimental, et nous analysons les résultats obtenus sans bruit d'annotation dans la section 7.7, et avec bruit d'annotation dans la section 7.8. Nous terminons par discuter les limites et perspectives de ces travaux en section 7.10.

7.2 Élagage de réseaux de neurones

Pour réduire la taille d'un réseau de neurones, il est possible d'appliquer des techniques d'élagage [LDS90 ; HP89 ; HS93]. Le processus d'élagage s'effectue en général au prix d'une baisse de performance du réseau, celle-ci correspondant simplement à une mesure de sa justesse sur un ensemble de test. L'élagage consiste à déterminer, à l'aide de critères précis, les parties du réseau, généralement les poids, qui contribuent le moins à la performance finale, de sorte à les éliminer. La figure 7.2 montre le principe d'un processus d'élagage itératif. La quantité de poids à élaguer est habituellement un paramètre défini manuellement. Ces techniques étaient initialement appliquées sur des réseaux entraînés au préalable [HS93 ; LDS90], puis suivies d'un fine-tuning. L'application d'un processus d'élagage itératif est une proposition récente [CI18], où les poids sont élagués à chaque fois que suffisamment d'epochs d'entraînement ont été effectuées.

Les récentes avancées sur la question de l'élagage ont proposé des points de vue innovant,

comme le fait d'élaguer un réseau directement à l'initialisation de ses poids, et l'entraîner pour la première fois seulement après coup [Hay+20]. Les possibilités sont nombreuses :

— [Han+15] réussi à élaguer un réseau de neurone sans baisser sa performance ;
— [FC18] montre l'existence d'un sous-réseau qui, lorsqu'il ait entraîné seul, peut atteindre une performance identique au réseau complet ;
— [Zho+19 ; Ram+19] parviennent à trouver, dans un réseau sur-dimensionné et initialisé aléatoirement, des sous-réseaux possédant déjà une bonne performance même sans avoir été entraîné.

Il existe principalement trois sortes de critères utilisées dans le cadre de ces techniques d'élagage.

— Le critère de magnitude : les poids sont triés par rapport à leur valeur absolue $|w|$. Ce critère est dit indépendant des données, dans le sens où il est possible de le mettre en œuvre sans avoir besoin d'utiliser les instances d'entraînement.
— Le critère de sensibilité [LAT18] : les poids sont ordonnés selon la valeur $|w\frac{\partial \mathcal{L}}{\partial w}|$, où $\mathcal{L}$ est la fonction de perte du réseau. On considère que ce critère dépend des données : il est nécessaire d'utiliser les instances d'entraînement pour le calcul de $\mathcal{L}$.
— Le critère hessien [WZG20] : l'importance des poids est quantifiée en utilisant la matrice hessienne de la fonction de perte. Ce critère est une version évoluée du précédent, et est donc aussi dépendant des données.

Dans nos expériences, nous sommes intéressés par deux aspects du processus d'élagage : le critère utilisé, et la stratégie d'application du processus. En ce qui concerne les critères, nous expérimentons avec le critère indépendant des données, i.e. le critère de magnitude, ainsi qu'un critère dépendant des données, le critère de sensibilité. Nous considérons ensuite trois stratégies d'élagage :

— une stratégie rapide correspondant à élaguer une seule fois le réseau, après quelques epochs d'entraînement ;
— une stratégie tardive correspondant à élaguer une seule fois le réseau, après un grand nombre d'epochs d'entraînement ;
— une stratégie itérative, correspondant à élaguer petit à petit le réseau au cours de l'entraînement, en ponctuant régulièrement ce dernier par une étape d'élagage.

7.3 Les Fabriques de Réseaux Convolutifs

La conception d'un réseau de neurones de structure adaptée à la résolution d'un problème précis est une tâche ardue qui implique souvent un grand nombre d'essais infructueux, étant donnée la quantité d'hyperparamètres à choisir. En ce qui concerne le traitement d'images, Verbeek et. al. [SV16] proposent un modèle en lien avec les RNC : la Fabrique de Réseaux Convolutifs. Une FRC est une structure multidimensionnelle, en opposition à un RNC classique où les couches se suivent de manière linéaire. La structure contient un nombre exponentielle de RNC différents, organisés en treillis, et partageant massivement leur poids, ce qui permet de les entraîner aisément en même temps. Cela simplifie donc le processus de sélection du meilleur modèle pour un problème donné. Une FRC contient un nombre important de poids, et par conséquent, elle peut largement bénéficier des techniques d'élagage de réseau, pour réduire considérablement le temps d'entraînement, ainsi que l'espace mémoire nécessaire pour la stocker.

La structure d'un RNC classique est déterminée par de nombreux hyperparamètres :

— le nombre de couches de convolution $Conv$;

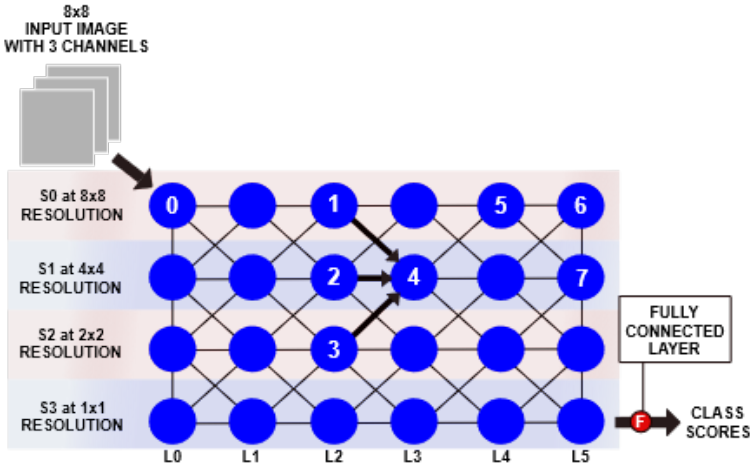


FIGURE 7.3 – Une FRC avec 6 couches et 4 niveaux de résolution. L'information est passée au nœud d'entrée (en haut à gauche) avec une première opération de convolution, puis se propage aux autres échelles de la première couche par des convolutions de sous-échantillonnage. La propagation continue vers les couches suivantes selon les liens de la structure, et converge vers le nœud de sortie (en bas à droite), pour être finalement transmise à une couche entièrement connectée.

— le nombre de chaînes, ainsi que la taille du filtre et le pas de décalage pour chacune des couches ;
— le nombre de couches de Pooling *Pool*, le type d'opérateur de ces couches et sa région d'application ;
— l'ordre d'application entre les deux types de couches *Conv* et *Pool* ;
— les fonctions d'activation à appliquer à chaque couche.

L'exploration de l'espace défini par cet ensemble d'hyperparamètres ne peut être réalisée de manière exhaustive, et la découverte d'une bonne structure de RNC en employant des heuristiques particulières reste un processus chronophage, lorsqu'on doit entraîner puis tester de nombreux réseaux successifs.

Cependant, l'ensemble des réseaux correspondant à différents choix de ces hyperparamètres peut être organisé de manière compacte au sein d'une unique structure en treillis (Figure 7.3), de sorte à pouvoir être entraîné en une seule fois. C'est ce qu'on appelle une FRC.

Une FRC traite l'information, i.e. l'instance reçue en entrée, selon trois axes :
— l'axe des couches, de taille $\mathcal{L}$;
— l'axe des échelles de résolution, de taille $\mathcal{S}$;
— l'axe des chaînes, de taille $\mathcal{C}$.

L'image d'entrée est vue comme un tenseur à 3 dimensions. Les deux premières dimensions correspondent à sa longueur et sa largeur. La troisième dimension correspond précisément à l'axe des chaînes. Chaque chaîne correspond donc à une matrice rectangulaire de pixels, et l'image d'entrée contient trois chaînes représentant le système de codage RGB. Cette image est d'abord transformée de sorte à atteindre le nombre de chaînes voulu au sein de la FRC $\mathcal{C}$, à l'aide d'une première opération de convolution (Figure 7.4.A). Le résultat de l'opération est le tenseur d'activation du nœud d'entrée de position $(couche = 0, échelle = 0)$, en haut à gauche de la structure (Figure 7.3). Ce tenseur subit ensuite de nombreuses transformations, suivant les liens de la FRC, jusqu'à ce que le tenseur d'activation du nœud de sortie $(\mathcal{L}-1, \mathcal{S}-1)$ soit finalement

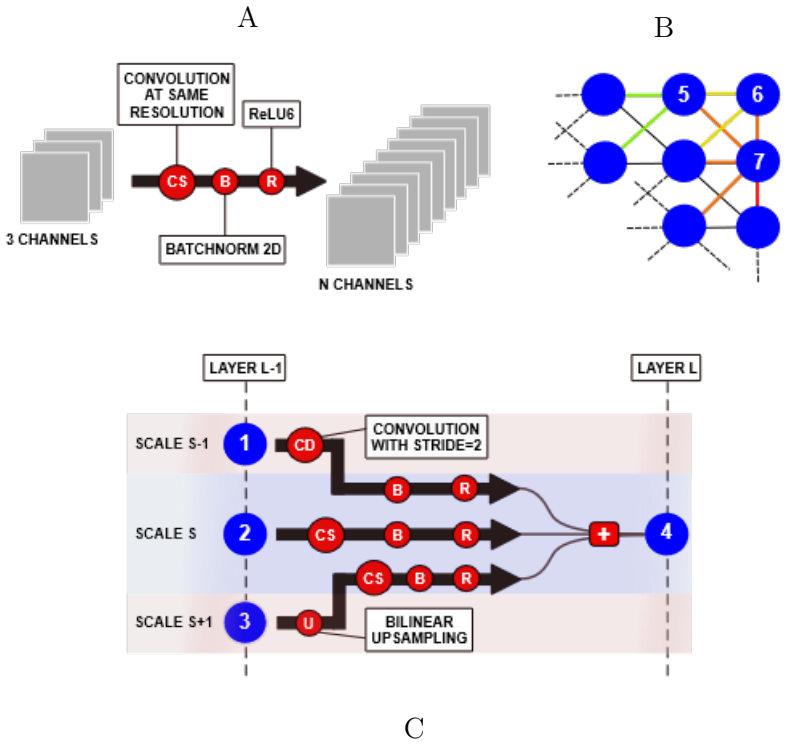


FIGURE 7.4 – Les figures A et C montrent respectivement comment le tenseur d'activation du nœud 0 est obtenu à partir de l'image reçue en entrée, et comment le tenseur d'activation du nœud 4 est calculé à partir des nœuds 1, 2 et 3. Le passage d'une échelle de résolution vers l'échelle inférieure est effectué par une opération de convolution de sous-échantillonnage, possédant un pas de décalage égal à 2. L'opposé est effectué en ajoutant une opération de sur-échantillonnage bilinéaire avant une opération de convolution de pas de décalage égal à 1. Enfin, la figure B montre comment des parties de la FRC peuvent devenir obsolètes (liens vert, jaune et orange) quand des liens précis sont élagués (lien rouge).

calculé. Dans la FRC de la figure 7.3, chaque nœud représente donc un tenseur d'activation de taille $(\mathcal{C}, W, H)$, où (W, H) est la résolution locale. La résolution à l'échelle 0 est la résolution de l'image d'entrée, et elle diminue de moitié à chaque passage au niveau de résolution inférieur. Les nœuds sont inter-connectés à l'aide des liens de la structure, en respectant un schéma de connexion précis :

— pour $(l, s) \in [0, \mathcal{L}-1] \times [0, \mathcal{S}-1]$, le nœud (l, s) est connecté aux trois nœuds les plus proches dans la couche précédente, i.e. aux positions $(l-1, s-1 \rightarrow s+1)$, ainsi qu'aux trois nœuds les plus proches dans la couche suivante, aux positions $(l+1, s-1 \rightarrow s+1)$, là où c'est possible ;

— les nœuds des premières et dernières couches propagent aussi l'information aux échelles inférieurs, et par conséquent, pour $l \in [0, \mathcal{L}-1]$ et $s \in [0, \mathcal{S}-1]$, le nœud (l, s) est connecté au nœud de l'échelle supérieure $(l, s-1)$, ainsi qu'au nœud de l'échelle inférieure $(l, s+1)$, lorsque c'est possible ;

— les liens ont un sens d'application, allant en règle générale de gauche à droite et de haut en bas.

Comme montré dans la figure 7.4.C, chaque lien applique un ensemble d'opérations au tenseur

d'activation de son nœud de départ, et transmet le résultat au nœud d'arrivée. Ainsi, chaque nœud rassemble le résultat des calculs des liens qui le précèdent, et les somme pour former son tenseur d'activation. Par exemple, le tenseur d'activation T d'un nœud (l, s) situé au milieu de la FRC est calculé par $T_{l,s} = \sum_{i\in\{-1,0,1\}} T_{l-1,s+i}$.

Les opérations typiques appliquées par un lien sont successivement :

— une convolution, avec un noyau de taille 3×3, éventuellement accompagnée d'un sur-échantillonnage ou d'un sous-échantillonnage selon la direction du lien (haut ou bas) si ce dernier n'est pas horizontal ;

— une normalisation de lot, ou *batch normalization* en anglais ;

— une fonction d'activation, qui dans notre cas correspond à une unité de rectification linéaire majorée par 6, abrégée en ReLU6, introduite pour la première fois dans [How+17] : $f(x) = min(max(x, 0), 6)$.

Verbeek et. al. argumentent dans [SV16] que les seuls hyperparamètres déterminants pour une FRC sont le nombre de chaînes et de couches, et ils montrent que le choix de la valeur de ces hyperparamètres est moins critique s'ils sont suffisamment grands, typiquement 8 pour le nombre de couches et 64 pour le nombre de chaînes. Le nombre d'échelles est en effet automatiquement déterminé selon la taille des images reçues en entrée, de sorte à ce que la résolution de l'image soit divisée par 2 à chaque niveau de résolution inférieur, et que la résolution soit à 1×1 au plus bas niveau $\mathcal{S} - 1$.

Structure des FRC de l'expérience

Dans le cadre de notre expérience, les tenseurs d'activation au sein d'une même FRC contiennent tous le même nombre de chaînes, et les chaînes de chaque couche sont entièrement connectées aux chaînes de la couche suivante. Cela signifie que, de manière similaire à un perceptron multicouches, où chaque neurone est entièrement connecté aux neurones de la couche précédente (Figure 7.1), le calcul d'une chaîne d'un tenseur d'activation A fait usage de l'ensemble des chaînes des tenseurs B_i le précédant et lui étant directement connectés par un lien de la FRC. Par conséquent, il n'est pas nécessaire de visualiser une FRC selon l'axe des chaînes. Dans la suite, nous considérons donc uniquement des plans 2D de nos FRC, i.e. selon l'axe horizontal des couches, et l'axe vertical des échelles de résolution.

7.4 Algorithme d'élagage

Notre processus d'élagage survient en deux étapes. Lors de la première étape, nous élaguons des liens complet de la FRC, i.e. les arcs sur le schéma de la figure 7.3, ce qui retire toutes les opérations qu'ils représentent du flux d'information ayant lieu dans la FRC. Par exemple, dans la figure 7.3, si le lien entre les nœuds 3 et 4 est supprimé, le nœud 4 ne reçoit plus que deux tenseurs à sommer au lieu de trois pour calculer son activation. De cette manière, nous déterminons en premier lieu, et à grande échelle, une sous-structure optimal au sein de la FRC, correspondant aux chemins les plus intéressants pour la résolution du problème d'apprentissage. En second lieu, nous appliquons un processus d'élagage classique aux poids des filtres de convolution de chaque lien restant dans la sous-structure obtenue. Nous n'élaguons cependant pas les poids de la couche entièrement connecté survenant après le nœud de sortie de la FRC, ainsi que les poids du lien d'entrée de la FRC, qui transforme l'image d'entrée pour atteindre le nombre de chaînes adéquat.

Rappelons que nous étudions trois stratégies différentes d'élagage de nos réseaux :

1. une stratégie **tardive**, correspondant à une étape d'élagage unique, effectuée plutôt tar-

divement lors de l'entraînement, suivie d'un fine-tuning du réseau ;

2. une stratégie **rapide**, similaire à la précédente excepté que l'étape d'élagage est appliquée très tôt lors de l'entraînement (moins d'une dizaine d'epochs en général) ;
3. une stratégie **itérative**, i.e. le réseau subit des élagages de manière régulière au cours de son entraînement.

Pour chaque processus, les critères d'élagage que nous avons considéré sont au nombre de deux :

— un critère indépendant des données : le critère de magnitude ;
— un critère dépendant des données : le critère de sensibilité.

Nous présentons en détails nos algorithmes d'élagage (cf. algorithme 3) dans ce qui suit. Nous dénotons par $\chi(e)$ la valeur du critère d'élagage appliqué à un élément e de la FRC. Cette valeur représente le score d'importance de e pour la performance finale du modèle.

Par exemple, e peut se référer au lien entre le noeud 1 et le noeud 4 de la FRC en figure 7.2, et $\chi(e)$ est un nombre réel positif. Plus celui-ci est proche de 0, plus on estime que l'importance du lien pour la tâche d'apprentissage est faible. L'élément e peut aussi correspondre à un seul poids du réseau, par exemple un des poids de l'opération de convolution appliquée par le lien précédent, auquel cas $\chi(e)$ mesure uniquement l'importance de ce poids, et non du lien en entier.

7.4.1 Élagage des liens

Pendant la première étape de notre processus d'élagage, la suppression d'un lien de la FRC peut rendre certaines autres opérations inutiles. Par exemple, dans la figure 7.4.B, la suppression du lien de couleur rouge empêche le nœud 7 de transmettre son tenseur d'activation aux nœuds suivants, étant donné que son seul lien de sortie disparait. Cela implique que toutes les opérations servant à déterminer ce tenseur ne sont plus utiles. Par conséquent, si l'on supprime le lien rouge, nous supprimons également les liens d'entrées du nœud 7 de couleur orange. Une suppression en chaîne survient alors : retirer les liens oranges crée une situation similaire pour le nœud 6, dont les liens d'entrées de couleur jaune doivent également être supprimés, suivis des liens verts, étant donné que le nœud 5 devient également obsolète.

Lors de ces suppressions, nous devons également nous assurer que nous ne retirons pas de lien qui couperait entièrement le flux d'information entre les nœud d'entrée et de sortie de la FRC. Pour éviter cette situation, nous faisons en sorte que l'ensemble $\mathcal{P}_L$ des liens que l'on s'apprête à supprimer satisfait la condition qui nous intéresse, à savoir que les liens restants permettent effectivement le flux d'information entre l'entrée et la sortie de la FRC. Nous assurons cette condition en classant tout d'abord les liens par leurs scores d'importance, définis par le critère d'élagage considéré, puis en ajoutant un à un les liens les moins important à $\mathcal{P}_L$, de sorte à vérifier après chaque ajout s'il existe toujours un chemin allant de l'entrée à la sortie si l'on retire les liens de $\mathcal{P}_L$ à la FRC. Si l'ajout d'un lien invalide la condition, nous le retirons de $\mathcal{P}_L$. Nous arrêtons la recherche des liens à supprimer lorsqu'on en a trouvé suffisamment, selon la quantité que l'on désire élaguer, ou alors lorsque tous les liens restant de la FRC ont été passé en revue. Les critères d'élagage utilisés pour les liens sont directement adaptés de ceux s'appliquant sur les poids : pour un lien l, et la matrice de poids W^l_{conv} du filtre de convolution de ce lien, le critère $\chi(l)$ est la norme euclidienne de la matrice obtenue en appliquant χ à chaque élément de W^l_{conv} :

$$\chi(l) = \left\| \chi(W^l_{conv}) \right\| .$$

Nous nous assurons enfin que le nombre de liens restants post-élagage est supérieur au nombre

de liens du plus grand chemin dans la FRC. Cela nous permet de laisser en théorie une chance d'être maintenu à n'importe quel chemin de la FRC, y compris le plus grand.

7.4.2 Élagage des poids

La deuxième étape du processus d'élagage est similaire à la première, à l'exception près que nous élaguons les poids des filtres de convolution des liens restants dans la FRC. En pratique, pour supprimer un poids w d'une matrice de poids W^l_{conv}, nous appliquons un masque binaire $\mathcal{M}_{W^l_{conv}}$, i.e. une matrice de 0 et de 1 de la même taille que W^l_{conv}, contenant un 0 à la même position que w, de sorte à annuler n'importe quelle opération future impliquant ce poids.

De manière analogue à l'élagage des liens, nous appliquons une condition aux poids pour déterminer si l'on peut les élaguer : un poids w peut être élagué seulement si la quantité de 0 dans $\mathcal{M}_{W^l_{conv}}$ n'en arrive pas à dépasser un seuil τ prédéfini manuellement. Dans nos expériences, τ est typiquement fixé à 90%. Nous faisons cela pour éviter que l'étape d'élagage des poids n'efface totalement les opérations de convolutions restantes dans la FRC, étant donné que l'équivalent est déjà réalisé lors de l'étape d'élagage des liens.

Algorithme 3 : Algorithme d'élagage

entrées : $\mathcal{E}$: l'ensemble des éléments de la FRC pouvant être élagués ; cet ensemble peut contenir soit des liens, soit les poids des filtres de convolutions ;
n : le nombre d'éléments à élaguer ;
χ : le critère utilisé pour classer les éléments par ordre d'importance ;
$\mathcal{C}$: une condition s'appliquant sur un élément et permettant de s'assurer que l'on est autorisé à l'élaguer.
sorties : un ensemble $\mathcal{P}_E$ d'éléments à élaguer
L est une liste vide destinée à contenir les éléments classés dans l'ordre ;
for *e in* $\mathcal{E}$ **do**
  Calculer le score $\chi(e)$ de e ;
  Insérer e dans L en respectant l'ordre défini par $\chi(e)$;
end
$\mathcal{P}_E$ est initialement vide ;
for *e in* L **do**
  if $\mathcal{C}(e)$ **then**
    Ajouter e à $\mathcal{P}_E$;
  end
end

7.5 Détails d'implantation

Pour l'implantation de nos expériences, nous avons utilisé la bibliothèque python PyTorch [Pas+17]. Le diagramme de classe en figure 7.5 montre comment nous avons implémenté une FRC. Une instance de la classe FRC contient de multiples instances de type nœud et Lien. Chaque instance de nœud possède des références aux instances de type Lien qui lui sont connectées, en général 3 en entrées et 3 en sortie. De même, chaque instance de type Lien possède des références vers les instances correspondant aux nœuds d'entrée et de sortie. La classe Lien représente de manière abstraite l'ensemble des opérations qui s'appliquent au tenseur d'activation T du nœud d'entrée (méthode *forward*). Les classes filles LienHoriz, LienHaut et LienBas

correspondent aux trois directions possibles pour un lien dans la FRC. La classe ConvMasque représente une opération de convolution, appliquée exactement une fois par lien, et contient également un tenseur M de même taille que son tenseur de poids, qui correspond au masque binaire permettant d'effacer la contribution des poids de l'opération de convolution ayant été élagué, en les multipliant par 0.

Implanter un réseau de neurones en PyTorch est facilement réalisé en regroupant l'ensemble des opérations du réseau dans une liste particulière (de type ListModule dans la documentation officielle de PyTorch). La manière dont ces opérations sont ensuite utilisées et combinées pour former la réponse du réseau est sauvegardée dans un graphe, lequel est ensuite utilisé pour calculer le gradient de la fonction de coût qui correspond exactement aux opérations venant d'être effectuées. Cela permet d'avoir un contrôle précis des opérations que l'on veut réaliser, sans devoir prévoir à l'avance l'ensemble des possibilités : ce que l'on calcule est précisément ce que l'on dérive. Optimiser les poids de toutes les opérations contenues dans la liste initiale se fait donc simplement par un unique appel de fonction, après avoir calculé la fonction de coût.

Cependant, cette liberté que permet Pytorch par rapport à la possibilité de faire varier le flux des calculs effectués au fur et à mesure de l'entraînement n'est pas exploitée à son plein potentiel lorsqu'on regroupe simplement les différentes opérations dans une structure de liste, sans sémantique supplémentaire sur la connexion entre ces opérations. C'est pourquoi nous avons choisi d'implanter la connexion entre les nœuds et les liens de nos FRC via le patron de conception « Observateur ». Ce patron de conception est principalement défini par l'utilisation de trois méthodes :

— *subscribe*(*objet*) : permet à un objet de *s'abonner* à un autre objet, de sorte à pouvoir être prévenu d'un évènement précis ;
— $notify()$: permet de prévenir tous les objets abonnés que l'évènement qui les intéresse est survenu ;
— *unsubscribe*(*objet*) : permet de se *désabonner* d'un objet pour ne plus être prévenu de l'évènement d'intérêt.

Dans notre cas, un nœud peut s'abonner à un lien, et inversement, un lien peut s'abonner à un nœud. La connexion entre lien et nœud est ainsi établie par la méthode *subscribe*, ce qui définit le flux de l'information ayant lieu au sein de la FRC. En pratique, lorsqu'une image est fournie en entrée de la FRC, le calcul de la sortie se fait de la manière suivante :

— chaque lien observe le nœud qui le précède, et attend que son tenseur d'activation soit mis à jour :
— une fois que le tenseur d'activation d'un nœud est mis à jour, le lien le récupère et applique les transformations de sa méthode *forward*, pour former un nouveau tenseur ;
— chaque nœud observe également les liens qui le précèdent (au maximum 4), et attend qu'ils aient tous terminés leurs calculs ;
— une fois que tous les liens précédents un nœud ont calculé leur tenseurs respectifs, ce nœud les récupère et les somme pour mettre à jour son tenseur d'activation.

Cette implantation nous a permis de réaliser de manière naturelle notre procédure d'élagage de liens. En effet, pour retirer un lien du flux des calculs, il suffit simplement de le désabonner du nœud qui le précède, et de désabonner le nœud qui le suit dudit lien. Bien sûr, il est tout de même nécessaire de s'assurer de la contrainte dont nous avons parlée en section 7.4.1, i.e. si un nœud devient obsolète, il doit également être retiré du flux des calculs avec les liens encore actifs qui lui sont connectés.

Dans la bibliothèque PyTorch, la manière usuelle d'implanter l'élagage consiste à utiliser les masques binaires, que nous avons introduit dans la section 7.4.2 sur l'élagage des poids. Cependant, une telle implantation ne permet pas de rendre plus rapide les calculs effectués. Elle

permet simplement de simuler la réponse du réseau, et le déroulement de son entraînement après élagage. En effet, ces poids étant mis à 0, leur effet sur le résultat des calculs est le même que s'ils avaient été réellement retirés. Mais ces poids sont tout de même encore présent dans les tenseurs, et restent impliqués dans les opérations effectuées. En revanche, notre implantation spécifique de l'élagage de lien permet de retirer concrètement du flux des calculs les opérations associées, ce qui a l'avantage d'améliorer nettement le temps que prennent ces calculs. En somme, notre élagage des poids, réalisé de manière conventionnelle, n'est pas optimisé pour le temps de calcul, tandis que l'élagage de liens que nous avons développé l'est.

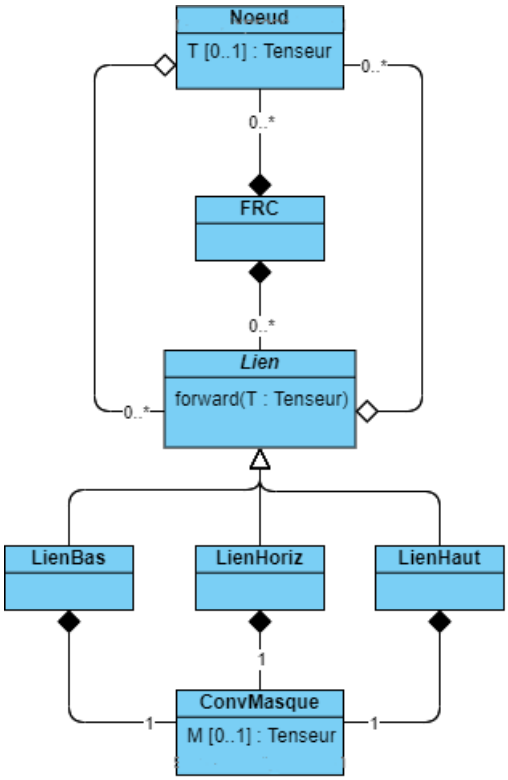


FIGURE 7.5 – Diagramme de classe d'une FRC.

7.6 Organisation des expériences

Pour observer le fonctionnement de nos algorithmes d'élagage, nous comparons la performance d'une FRC de référence, non élaguée, avec différentes FRC élaguées. Les modèles qui sont comparés sont bien sûr entraînés sur les mêmes ensembles. Nous utilisons pour cela 4 ensembles de données différents, que nous présentons en section 7.6.1. Nous entraînons ainsi une FRC de référence par ensemble, ce qui nous en donne 4 différentes. Nous construisons également pour chaque ensemble 18 FRC élaguées, obtenues pour différents paramètres de notre processus d'élagage. Ces expériences sont réalisées dans un premier temps sur nos 4 ensembles (résultats en section 7.7), puis dans un second temps, nous répétons les mêmes expériences, mais en introduisant un bruit d'annotation artificiel dans chaque ensemble (résultats en section 7.8). De cette manière, nous pouvons également observer l'influence d'instances mal annotées sur les processus d'élagage que nous appliquons, en comparant d'un côté les performances des réseaux obtenus par

élagage sur les données initiales, et de l'autre ceux obtenus par élagage sur des données bruités. Les performances de nos réseaux sont toujours calculées sur l'ensemble de données non bruité.

Les expériences ont été implantées à l'aide de la bibliothèque PyTorch, et les modèles ont été entraînés sur la plateforme Grid5000 [Bal+13], *en utilisant uniquement une GPU par réseau entraîné*. Nous n'avions en effet pas le luxe d'utiliser des centaines de GPU pour chaque entraînement, et nos processus d'élagage ont par conséquent été choisi de sorte à demander peu de temps de calcul, au détriment de la qualité d'estimation des meilleurs éléments à élaguer. Avec une puissance de calcul supérieure, des techniques d'analyse de sensibilité plus poussées, e.g. fondées sur l'estimation d'indices de Shapley [Sha53], pourraient être utilisées pour appliquer des critères d'élagage plus qualitatifs, et obtenir des structures élaguées plus performantes.

7.6.1 Les ensembles d'instances

Nous avons considéré 4 ensembles d'instances différents pour nos expériences :

— L'ensemble CIFAR10 [KH+09] : 60000 32×32 images en couleur, étiquetées selon 10 classes, chacune contenant 6000 images, parmi lesquelles 5000 sont utilisées pour l'entraînement, et 1000 sont gardées pour le test. Cela donne donc un total de 50000 instances d'entraînement, et 10000 instances de test.
— L'ensemble CIFAR100 [KH+09] : similaire à CIFAR10, cet ensemble contient par contre des images pouvant appartenir à 100 classes différentes. Chacune de ces classes représentent 500 images pour l'entraînement, et 100 images pour le test.
— L'ensemble SVHN [Net+11] : 600000 32×32 images couleur montrant des chiffres. Nous n'utilisons que les 73257 images contenues dans leur ensemble d'entraînement principal, ainsi que les 26032 images de leur ensemble de test, sans toucher aux 531131 images de l'ensemble d'entraînement secondaire.
— L'ensemble de classification VOC2012, provenant du challenge PASCAL VOC [Eve+] : 11530 images couleur de 500 pixels de largeur, et de hauteur variable, pour un total de 27450 annotations de régions contenant des objets particuliers. Ces objets peuvent appartenir à 20 classes différentes, et chaque image peut en contenir plusieurs. Les annotations contiennent plus d'informations que simplement la classe de l'objet annoté, comme par exemple les coordonnées du rectangle délimitant la position de l'objet. Cela s'explique par le fait que le challenge PASCAL VOC pose un problème de classification multi-étiquettes, et de détection de la position de l'objet dans les images. Comme nous nous intéressons à de la classification simple, nous appliquons au préalable une transformation sur l'ensemble VOC2012, pour récolter uniquement les images qui sont adéquates pour notre tâche. De cette manière, nous obtenons un ensemble de 9340 images, chacune associée à une seule étiquette.

La transformation que nous appliquons à chaque image d de l'ensemble VOC2012 est :

— si d ne contient qu'un seul objet, ou plusieurs objets de la même classe, nous gardons d en l'associant à l'étiquette correspondante ;
— sinon, nous calculons les surfaces des rectangles contenant les objets : si la surface la plus grande est au moins deux fois supérieure à la deuxième plus grande surface, nous gardons d en l'associant à l'étiquette correspondante au plus grand objet ;
— sinon, nous ne gardons pas l'image d.

7.6.2 Les instances d'entraînement

Les instances d'entraînement utilisées consistent en seulement 90% des ensembles d'entraînement initialement définis dans [KH+09 ; Net+11] pour CIFAR10, CIFAR100 et SVHN. Avec les 10% restants, nous formons un ensemble de validation, utilisé lors de l'application du critère d'élagage de sensibilité. En effet, ce critère a besoin du calcul du gradient de la fonction de perte, et nous effectuons donc ce calcul sur l'ensemble de validation. Dans le cas où nous appliquons l'autre critère, nous faisons tout de même la séparation entre ensemble d'entraînement et ensemble de validation, de sorte à ce que tous nos modèles construits pour un ensemble donné aient été entraînés sur le même nombre d'instances, indépendamment du critère d'élagage utilisé. Pour l'ensemble VOC2012, étant donné qu'aucun ensemble de test est fourni, nous appliquons la séparation suivante : 70% des instances sont utilisées pour l'entraînement, 20% pour le test, et les 10% restants forment l'ensemble de validation. Lorsque nous séparons un ensemble, nous nous assurons de maintenir la même proportion de classes dans les sous-ensembles obtenus. En ce qui concerne l'augmentation artificielle des instances, nous appliquons une fonction de normalisation, suivie éventuellement par un réajustement de la taille de l'image, puis nous appliquons des fonctions aléatoire de rognage et d'inversion horizontale (cf. figure A.1 en annexe).

7.6.3 Structure standard des FRC

Les FRC utilisées dans nos expériences contiennent un nombre de 8 couches et 64 chaînes. Nous avons en effet déterminé que ces valeurs sont optimales pour notre problème car nous obtenions de cette manière les FRC les plus performantes en un temps d'entraînement acceptable, i.e. de l'ordre de quelques heures. Nous notons cependant une exception pour l'ensemble SVHN, pour lequel nous avons réduit le nombre de chaînes à 32, car nous n'avons observé aucune différence de performance dans ce cas précis. Pour l'entraînement des FRC, la fonction de perte que nous utilisons est celle de l'entropie croisée, et nous appliquons un algorithme de descente de gradient stochastique, pendant un maximum de 200 epochs. Le pas d'apprentissage vaut 0.1 avant l'epoch 80, et nous le divisons par 10 entre les epochs 80 et 120, puis à nouveau par 10 après l'epoch 120.

7.6.4 L'algorithme d'élagage

L'algorithme d'élagage est appliqué avec trois hyperparamètres que nous faisons varier :

1. Tout d'abord, nous définissons trois différentes valeurs pour la fraction F à élaguer dans une FRC : 95%, 97% et 99%. Élaguer 95% d'une FRC correspond donc à élaguer 95% des liens et 95% des poids.
2. Nous définissons également trois stratégies d'élagage :
   — élagage rapide : dès l'epoch 5, l'élagage de lien est appliqué, suivi de l'élagage de poids, pour élaguer directement F% de la FRC ;
   — élagage tardif : idem que pour l'élagage rapide, mais à l'epoch 75 ;
   — élagage itératif : après l'epoch 5, nous appliquons l'élagage de liens, suivi par l'élagage de poids, et nous répétons cela toutes les 10 epochs jusqu'à atteindre l'epoch 75 ; le nombre de liens et de poids élagués à chaque itération est calculé selon un profil linéaire de sorte à ce que, arrivé à l'epoch 75, une fraction F% de la FRC ait été élaguée.
3. Le dernier hyperparamètre de l'algorithme d'élagage est simplement le critère considéré, pouvant être soit le critère de magnitude, soit le critère de sensibilité. Nous laissons de côté le critère hessien pour des travaux futurs.

Cela donne donc un total de 18 configurations différentes pour l'algorithme d'élagage, et ce pour chacun des 4 ensembles d'instances considérés dans nos expériences.

CIFAR10

Model	**Param.**	**Test Err. (%)**
FRC-B	>4M	6.44
FRC-P	0.14M	8.03
Maxout [Goo+13]	>5M	9.38
DSN [Lee+15]	0.97M	7.97
RCNN-96 [LH15]	0.67M	7.37
RCNN-160 [LH15]	1.86M	7.09

CIFAR100

Model	**Param.**	**Test Err. (%)**
FRC-B	>4M	27.54
FRC-P	0.24M	33.26
Maxout [Goo+13]	>5M	38.57
DSN [Lee+15]	0.97M	34.57
RCNN-96 [LH15]	0.67M	34.18
RCNN-160 [LH15]	1.86M	31.75

SVHN

Model	**Param.**	**Test Err. (%)**
FRC-B	0.29M	3.56
FRC-P	15K	4.43
Maxout [Goo+13]	>5M	2.47
DSN [Lee+15]	0.97M	1.92
RCNN-160 [LH15]	0.67M	1.80
RCNN-192 [LH15]	1.86M	1.77

FIGURE 7.6 – Comparaison des FRC obtenues sur CIFAR10, CIFAR100 et SVHN avec différentes modèles de référence. FRC-B représente la FRC non élaguée, FRC-P est la FRC élaguée ayant obtenue la meilleure performance finale.

7.7 Résultats sur les ensembles initiaux

Les résultats obtenus pour les FRC élaguées sur les ensembles non bruité montrent que :

— l'élagage de FRC permet d'obtenir un modèle dont la performance est acceptable pour une tâche de classification d'image, bien qu'il contienne très peu de poids par rapport à la norme 7.7.1 ;

— le critère MAGN est globalement préférable au critère SBD, essentiellement pour sa simplicité d'utilisation, car son efficacité semble être au moins équivalente à SBD 7.7.2 ;

— la stratégie d'élagage itérative forme un bon compromis entre gain en temps d'entraînement et performance finale 7.7.2 ;
— l'étape d'élagage de lien est bénéfique pour les FRC, permettant d'obtenir des modèles au moins équivalents à ceux obtenus uniquement en élaguant les poids, et plus performants lorsque la stratégie d'élagage rapide est appliquée 7.7.3.

7.7.1 État de l'art

Nous montrons en figure 7.6 une comparaison entre les performances que nous avons pu obtenir avec des FRC, élaguées ou non, et les performances de différents modèles dans la littérature. Globalement, on remarque que le fait d'élaguer grandement nos FRC ne les rend pas trop mauvaises par rapport aux autres modèles. On voit par exemple que sur CIFAR10, notre FRC non élaguée, notée FRC-B, obtient un taux d'erreur un peu plus faible que les autres modèles, mais contient plus de $4M$ de poids. En revanche, l'une de nos FRC élaguée ne contient que $0.14M$ de poids, i.e. largement moins que les autres modèles, et présente un taux d'erreur à peine plus élevé, si ce n'est meilleur, quand on compare avec le modèle Maxout. L'élagage de FRC peut donc constituer un bon compromis lorsque l'on désire un modèle de très petite taille possédant une performance acceptable sur un problème de classification d'image.

En ce qui concerne l'ensemble PASCALVOC, il est difficile de fournir une comparaison de la sorte, car nous avons utilisé les données de cet ensemble d'une manière détournée, à des fins de simple classification, tandis que dans la littérature, il sert à de la détection localisée d'objets. Nous aurions en effet besoin d'utiliser la « Mean Average Precision », une mesure dont le calcul nécessite d'utiliser les régions prédites par le modèle pour chaque objet détecté, et nos FRC n'ont pas été conçues pour cela. Nous pensons malgré tout que PASCALVOC permet de montrer comment nos FRC élaguées se comportent en comparaison à la FRC entière sur un problème de classification d'images différent de ceux posés par les trois autres ensembles.

7.7.2 Paramètres de l'algorithme d'élagage

Pour la comparaison des différents paramètres de notre algorithme d'élagage, nous nous basons principalement sur les FRC que nous avons élaguées à 95% et 97%. Celles ayant subies 99% d'élagage présentent en effet des performances sensiblement dégradées, et trop variables pour que l'on puisse les considérer. Les performances des FRC sont listées en détails en annexe B.

En termes de performance des FRC élaguées, nous avons obtenu les meilleurs résultats en appliquant la stratégie d'élagage tardif. Cela s'explique simplement par le fait que lorsque la FRC est suffisamment entraînée, il est plus facile de détecter les parties moins importantes du réseau, et de les retirer sans trop diminuer la performance finale. Cependant, la stratégie d'élagage itératif a également donné des résultats souvent équivalents à l'élagage tardif.

Nous remarquons également que le critère de magnitude (MAGN) permet d'obtenir des FRC élaguées légèrement plus performantes que lorsque le critère de sensibilité (SBD) est appliqué, comme le montre la figure 7.7. Les pourcentages représentent le nombre de fois qu'une FRC élaguée avec l'un des deux critères surpasse la FRC élaguée avec l'autre critère, lorsque toutes les autres variables, i.e. l'ensemble d'instances considérés, la stratégie d'élagage, et la fraction à élaguer, sont fixes. Il est cependant possible que le manque d'efficacité observé pour le critère de sensibilité soit principalement dû à la petite taille de l'ensemble de validation utilisé pour le calculer. En effet, des estimations de basse qualité pour le gradient de la fonction de perte peuvent influer sur la détermination des parties non importantes du réseau, et par conséquent

rendre moins efficace ledit critère.

	MAGN	SBD
Ensembles sans bruit d'annotation		
CIFAR10	100%	0%
CIFAR100	83%	17%
SVHN	33%	67%
PASCALVOC	33%	67%
OVERALL	62.5%	37.5%

	MAGN	SBD
Ensembles avec bruit d'annotation		
CIFAR10	33%	67%
CIFAR100	50%	50%
SVHN	83%	17%
PASCALVOC	42%	58%
OVERALL	52%	48%

FIGURE 7.7 – Comparaison de l'efficacité des deux critères d'élagage MAGN et SBD. E.g, Pour CIFAR10, sur les ensembles sans bruit d'annotation, le critère MAGN donne les meilleurs FRC 100% du temps.

En ce qui concerne les temps d'entraînement des FRC, l'idéal est naturellement d'élaguer le plus tôt possible. Étant donné que la stratégie itérative donne des FRC avec une performance finale en général plus élevée que la stratégie rapide, tout en débutant l'élagage très tôt lors de l'entraînement, elle semble constituer un bon compromis entre perte de performance finale et gain en temps d'entraînement, ainsi qu'en consommation mémoire. Malheureusement, étant donné l'implantation de l'élagage en PyTorch, nous n'avons pas accès à des estimations correctes de ce gain temporel. En effet, l'élagage des poids d'une opération de convolution s'effectue en appliquant un masque binaire sur la matrice de poids de l'opération. Cela ne permet pas de diminuer le temps de calcul, mais simplement d'effacer l'impact des poids élagués sur les calculs effectués au sein de la FRC. En revanche, notre implantation de l'élagage de lien permet d'améliorer le temps de calcul, car nous modifions concrètement le flux de l'information dans la FRC. Par conséquent :

— entre une FRC non élaguée et une FRC ayant subi un élagage de ses *poids*, i.e. implanté par l'application de masque binaire sur les opérations de convolution, aucun gain en temps de calcul ne peut être observé ;
— si la FRC élaguée a également subie un élagage de ses liens, i.e. avec notre implantation de l'élagage de lien, il est possible d'observer un gain en temps de calcul.

En somme, le gain de temps dû à l'élagage des poids n'est pas observable, tandis que celui dû à l'élagage de liens peut être observé. Étant donné que nos FRC sont élaguées par un mélange des deux types d'élagages, les relevés de leur temps d'entraînement ne sont pas corrects. Les temps d'entraînement obtenus sont plus longs que ce qui aurait été atteignable avec une implantation efficace de l'élagage des poids. Malgré tout, nos relevés des temps d'entraînement moyens permettent de voir partiellement le gain que l'on peut attendre d'une structure élaguée :

— 4 heures pour les FRC obtenues par élagage rapide ;
— 6 heures pour les FRC obtenues par élagage itératif ;
— 8 heures pour les FRC obtenues par élagage tardif ;
— et entre 8 et 14 heures pour les FRC non élaguées.

7.7.3 L'élagage de lien

L'élagage de lien nous permet d'observer simplement les sous-structures préférables pour le problème de classification (figure 7.8). Par exemple, l'idéal pour les images de CIFAR10 et CIFAR100 semble être de les manipuler à différentes échelles de résolution en même temps, au moins lors des couches initiales pour CIFAR10, et jusqu'aux dernières couches pour CIFAR100.

CIFAR10 CIFAR100

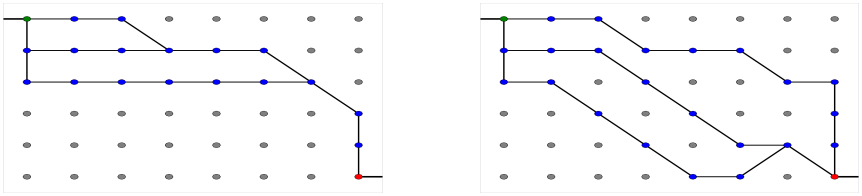


FIGURE 7.8 – Exemples de sous-structures performantes pour CIFAR10 et CIFAR100, obtenue par élagage itératif à 95% avec le critère MAGN.

Lors de nos expériences, nous avons également tenté d'appliquer notre algorithme d'élagage en retirant uniquement des poids, comme cela est fait de manière classique. Lorsque nous confrontons nos modèles obtenus avec élagage de liens, et ceux obtenus sans élagage de liens (figure 7.9), nous remarquons que :

— lorsqu'on applique un élagage rapide, inclure l'étape d'élagage de liens donne toujours les meilleurs résultats ;
— en ce qui concerne l'élagage itératif ou tardif, les modèles obtenus avec ou sans élagage de liens sont globalement similaires en terme de performance.

	MAGN		**SBD**	
	LP+WP	WP	LP+WP	WP
RAPIDE	0.56	0.0	0.44	0.0
ITÉRATIF	0.22	0.33	0.33	0.12
TARDIF	0.45	0.45	0.10	0.0

FIGURE 7.9 – Comparaison des modèles obtenus avec élagage de liens (LP+WP) et sans élagage de liens (WP). E.g pour l'élagage rapide, l'application de l'élagage de lien est toujours préférable. En effet, dans ce cas, les modèles les plus performants sont toujours obtenus avec élagage de liens, 56% du temps avec le critère MAGN, et 44% du temps avec le critère SBD.

Cela indique que lorsqu'il est nécessaire d'élaguer rapidement la FRC, e.g. l'entraînement doit consommer peu de ressources, il est préférable d'appliquer un mélange entre élagage de liens et de poids pour obtenir une FRC élaguée avec de meilleurs performances.

7.8 Répétition des expériences avec bruit d'annotation

Nous voulons maintenant observer l'impact du bruit d'annotation sur le processus d'élagage. En principe, le critère de magnitude MAGN étant indépendant des données, il devrait être insensible à la présence d'instances mal annotées, contrairement au critère de sensibilité SBD qui en a l'usage. Il semble donc naturel de s'attendre à ce que le critère de magnitude soit le plus efficace, lorsque les ensembles d'instances contiennent du bruit d'annotation. Nous avons donc répété les mêmes expériences, sur les mêmes ensembles, à la différence que nous y avons introduit au préalable un bruit d'annotation artificiel. Cela nous permet d'observer et de comparer l'efficacité de ces deux critères quand appliqués dans une situation où les instances sont mal annotées.

7.8.1 Introduction du bruit d'annotation

Il nous faut en premier lieu introduire artificiellement du bruit d'annotation dans nos quatre ensembles, de sorte à pouvoir comparer les FRC élaguées par l'un ou l'autre des critères d'élagage. Pour la validité de nos conclusions, nous voulons également que la structure du bruit d'annotation introduit ait une complexité aussi réaliste que possible. Nous excluons donc l'utilisation d'un modèle de bruit uniforme. Typiquement, une structure de bruit réaliste signifie pour nous que le modèle entraîné est capable d'apprendre ce bruit, et de le reproduire dans ces prédictions futures. Par conséquent, nous pouvons évaluer si le bruit d'annotation introduit est satisfaisant en regardant les valeurs de l'indicateur F_n (Cf. partie 6.3) mesurées sur les instances *de test* pour les modèles entraînés sur les ensembles bruités. En effet, plus F_n est haut, plus le modèle est capable de prédire l'étiquette des instances mal annotées, donc de généraliser la structure du bruit d'annotation à de nouvelles instances.

Pour introduire du bruit d'annotation, nous avons adopté une approche plus simple que celle mise en œuvre dans la partie 6.5.2 (cf. figure 7.10). En effet, d'un point de vue chronologique, nous avons mené les expériences du chapitre 6 après celles-ci, pour lesquelles notre algorithme d'introduction de bruit n'était par conséquent pas encore totalement développé. Nous avons donc entraîné plusieurs FRC de dimensions variées sur chaque ensemble, de sorte à obtenir des modèles obtenant différents taux d'erreurs, allant de 1% à 40%. Ces modèles sont entraînés sur la totalité des instances, i.e. les instances d'entraînement et de test, et la justesse est ensuite mesurée sur les mêmes instances ayant servies pour l'entraînement (Cf. mesure de re-substitution en section 2.3.3).

Pour introduire 10% de bruit dans un ensemble, il suffit ensuite de choisir un modèle possédant une erreur de re-substitution de 10% sur le même ensemble, et de remplacer les annotations par les prédictions de ce modèle. De cette manière, le bruit d'annotation obtenu est dépendant des caractéristiques des instances, bien que cette méthode soit bien moins sophistiquée que celle que nous avons appliquée en section 6.5.2. Nous avons néanmoins préféré cette méthode à celles existant dans la littérature. En effet, ces dernières appliquent en général un modèle de bruit par classe, ou alors choisissent l'option la plus simple et introduisent un bruit d'annotation de manière uniforme. Après de nombreux essais, nous avons empiriquement observé que les valeurs de F_n étaient plus élevées pour les modèles ayant appris sur les ensembles bruités avec notre méthode d'introduction de bruit, que ceux ayant appris sur des ensembles bruités selon un modèle de bruit par classe ou uniforme, ce qui explique notre choix.

Nous avons ainsi obtenu des versions bruitées à 10% et 20% pour CIFAR10, CIFAR100, SVHN et PASCALVOC.

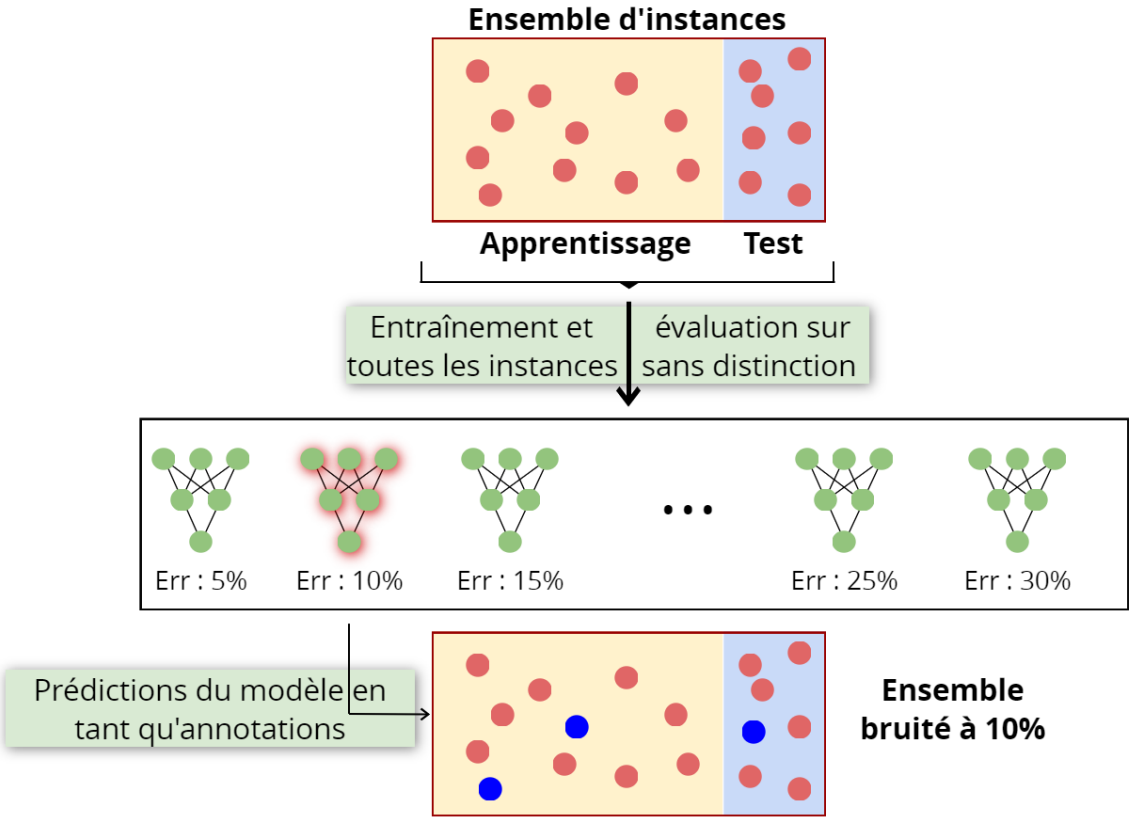


FIGURE 7.10 – Schéma explicatif de l'obtention d'une version bruité à x% d'un ensemble d'instances.

7.8.2 Résultats

Nous avons répété les expériences menées précédemment, à la différence que les ensembles d'instances sont à présent bruités. Pour tester les FRC, nous avons mesuré leurs justesses sur les ensembles de test *non bruités*, ainsi que les valeurs de F_n et F_c pour avoir une idée de l'impact du bruit d'annotation sur les prédictions des modèles. La figure 7.11 montre les valeurs de F_n et F_c obtenues *en moyenne* sur toutes les FRC, pour chaque ensemble d'instance. On peut voir par exemple que les FRC entraînées sur SVHN sont les plus biaisées par le bruit introduit ($F_n > 40\%$), tandis que celles entraînées sur CIFAR100 sont les moins biaisées ($F_n < 25\%$).

Dans un premier temps, selon les valeurs de F_n, il apparaît que les FRC élaguées sont biaisées de manière équivalente, si ce n'est plus, que les FRC non élaguées. Cela contredit l'intuition qu'on pourrait avoir : un modèle plus petit a moins de chances d'apprendre une structure complexe, i.e. la structure du bruit d'annotation. De plus, ce sont les modèles élagués par le critère de magnitude qui présentent en général des valeurs de F_n légèrement plus grandes que les autres, malgré que ce critère soit considéré comme indépendant des instances et de leur annotation.

De manière surprenante, l'efficacité du critère de sensibilité est meilleure que prévue, malgré sa dépendance aux instances. Inversement, le critère de magnitude présente une efficacité inférieure à ce qui était espéré. En effet, les résultats présentés en section 7.7 ont montré une légère supériorité de MAGN sur SBD, et nous nous attendions à ce que cette supériorité s'affirme davantage en expérimentant avec du bruit d'annotation. Nous avons néanmoins remarqué la tendance inverse,

ce qui apparaît sur la figure 7.7. Nous pensons qu'une explication de ce phénomène est lié au fait que plus les prédictions d'un modèle sont biaisées par le bruit d'annotation, plus cela est encodé dans ses poids, ce qui implique un impact négatif sur l'efficacité du critère MAGN.

Nous notons une exception pour l'ensemble SVHN, pour lequel le critère SBD est clairement devenu moins efficace lors des expériences avec bruit d'annotation. Étant donné la haute valeur de F_n pour les modèles entraînés sur cet ensemble, le nombre de prédictions biaisées (i.e. prédictions des étiquettes bruitées) sur l'ensemble de validation SVHN est plus élevé que dans le cas des autres ensembles. Une hypothèse possible est que cela pourrait avoir trop impacté la qualité de l'estimation de SBD pour SVHN, ce qui expliquerait la baisse en efficacité de ce critère dans ce cas particulier.

Pour résumer, on peut poser les conclusions suivantes :

— élaguer un modèle de réseau de neurones, et donc le rendre moins complexe, ne permet pas forcément d'être moins sensible au bruit d'annotation dans l'ensemble d'apprentissage ;
— de façon contre-intuitive, le critère indépendant des données, MAGN, perd en efficacité en présence de bruit d'annotation, jusqu'à devenir moins efficace que le critère dépendant des données, SBD.

	CIFAR10	CIFAR100	SVHN	PASCALVOC
F_c	$> 80\%$	$> 60\%$	$\approx 97\%$	$\approx 50\%$
F_n	$\approx 35\%$	15%-25%	40%-50%	10%-30%

FIGURE 7.11 – Valeurs de F_n et F_c pour les FRC entraînées sur les versions bruités de chaque ensemble. Les valeurs sont données ici sous la forme d'intervalles, comprenant toutes les FRC, i.e. élaguées ou non, et obtenues avec MAGN ou SBD, sans distinction. Néanmoins, en règle général, les valeurs de F_n pour les FRC non élaguées correspondent aux bornes inférieures des intervalles, et les valeurs de F_c correspondent aux bornes supérieures. Par exemple, pour CIFAR10, les valeurs de F_c de toutes les FRC sont supérieures à 80%, avec la valeur maximale obtenue par la FRC non élaguée, et les valeurs de F_n gravitent autour de 35%, avec la valeur minimale également obtenue par la FRC non élaguée.

7.9 Application de méthodes d'analyse de sensibilité pour l'élagage de réseaux de neurones

Nous avons également mené de nombreuses expériences pour concevoir un autre critère d'élagage de nos FRC, fondé sur des notions d'analyse de sensibilité. Le critère de sensibilité, que nous avons utilisé dans nos expériences, et le critère hessien que nous avons présenté en section 7.2, se fondent déjà sur ces principes. Nous voulions cependant pousser les possibilités du champs de l'analyse de sensibilité un peu plus loin, en mettant en œuvre dans nos processus d'élagage des indices de sensibilité tels que ceux de Sobol ou de Shapley. Dans ce qui suit, après avoir présenté le principe de l'analyse de sensibilité, nous expliquons nos différentes expérimentations sur le sujet, concernant plus précisément la méthode de *screening*, et l'estimation des indices de Shapley.

7.9.1 L'analyse de sensibilité

L'analyse de sensibilité regroupe un grand nombre de méthode permettant le calcul d'indices de sensibilité d'un modèle quelconque. Étant donné un modèle disposant de plusieurs variables d'entrée, et d'une variable de sortie, les indices de sensibilité permettent de quantifier à quel point la variabilité de la sortie peut être expliqué par chaque entrée du modèle, ou par des combinaisons de ces entrées [Mor91 ; CCS07 ; Sob01 ; GZ20 ; BPP17]. Les indices de Sobol et de Shapley [HS96 ; Sha53] sont un exemple d'indices populaires dans le domaine. La difficulté d'application de ces méthodes provient souvent du fait que le nombre de variables d'entrée du modèle est beaucoup trop grand, car une tâche habituelle de ce genre de méthodes est l'échantillonnage de l'espace mathématique dans lequel vivent ces entrées. Des techniques ont cependant récemment été développées pour proposer des solutions à l'analyse de sensibilité de modèle en grande dimension [BPP17 ; GZ20].

7.9.2 Critères d'élagage fondés sur des techniques d'analyse de sensibilité

Pour la conception d'un critère d'élagage, il nous faut un moyen de quantifier l'importance d'un élément de la FRC. Selon le principe de l'analyse de sensibilité, l'importance d'un élément au sein d'un modèle se quantifie par la variation mesurée de la sortie de ce modèle. Dans notre cas, le modèle correspond à la FRC, et la sortie que l'on considère est la norme euclidienne du tenseur d'activation du nœud de sortie. Les entrées du modèle, i.e. les éléments dont on veut quantifier l'importance, sont les matrices de poids des liens de la FRC.

La méthode du *screening*

Nous avons pour cela tenter en premier lieu de mettre en place une méthode de *screening*. Le *screening* correspond à perturber une à la fois chaque variable d'entrée selon une direction et une amplitude particulière, et à évaluer l'écart répercuté sur la variable de sortie. Répéter le processus plusieurs fois permet de relever plusieurs écarts liés à la même variable d'entrée, que l'on peut agencer dans un vecteur. Cela nous permet, pour chaque variable d'entrée, de calculer la moyenne μ et la variance σ du vecteur correspondant, de sorte à pouvoir disposer μ et σ dans un graphe en deux dimensions (figure 7.12).

Dans ce graphe, plus une variable est située à droite, plus elle a une influence linéaire sur la sortie du modèle. On peut parler d'influence d'ordre 1. De même, plus elle est située en haut, plus il y a de risques qu'elle ait une influences non-linéaire, ou alors que son influence dépende de la valeur d'autres variables. On parle alors d'influence d'ordre 2 ou plus. Par exemple, dans le graphe en figure 7.12, la variable 1 a une influence essentiellement linéaire assez élevée. La variable 3 est encore plus influente, et possède probablement des effets non-linéaire ou inter-dépendants avec certaines autres variables. Si l'on devait choisir une variable parmi les quatre à ignorer dans ce modèle, la variable 4 serait le choix idéal, étant donné son niveau d'influence négligeable.

Pour estimer l'importance des matrices de poids de chaque lien de la FRC avec une méthode de screening, nous avons mis en place le processus suivant :

- choisir une image aléatoirement dans l'ensemble de validation, et la donner en entrée à la FRC ;
- mesurer la norme du tenseur d'activation du nœud de sortie $\|T_s\|$;
- pour un lien i, perturber sa matrice de poids W_i par une matrice de perturbation E_i choisie aléatoirement, dont les éléments sont $\pm\lambda$, avec λ le pas d'apprentissage utilisé lors de l'entraînement ;

— évaluer à nouveau la sortie de la FRC pour la même image, et mesurer l'écart entre $\|T_s\|$ et la nouvelle norme du tenseur d'activation du nœud de sortie ;
— répéter l'opération k fois, pour pouvoir mesurer la moyenne des écarts μ_i, ainsi que leur variance σ_i ;
— passer au lien suivant, et répéter le processus ;
— Les liens sont ensuite classés par ordre d'importance selon les valeurs de μ, puis selon les valeurs de σ.

Nous avons cependant rencontré des difficultés à mettre en place ce critère d'élagage, étant donné la dimension de nos matrices de poids. En effet, chaque convolution, ayant lieu dans un lien d'une FRC dont le nombre de chaîne est de 64, contient de l'ordre de 10^6 poids. De plus, un nombre de couches de 8 équivaut à plus d'une centaine de lien. Pour obtenir des estimations convenables, il était nécessaire que la valeur du nombre de répétition k, dans la procédure ci-dessus, soit suffisamment grande. Pour ces raisons, le temps de calcul était trop grand, et nous n'avons pas pu obtenir de résultats satisfaisant.

Indices de Shapley

Nous avons ensuite expérimenté avec les indices de Shapley, indices de sensibilité tenant compte de relations de dépendance d'ordre supérieure à 2. En comparaison, le critère de sensibilité tient seulement compte des relation d'ordre 1, étant donné qu'il fait usage de la dérivée de la fonction de perte du réseau, tandis que le critère hessien prend uniquement en compte les relations d'ordre 2, en considérant les dérivés secondes de la fonction de perte. L'intérêt des indices de Shapley pour le processus d'élagage est donc qu'ils permettent de tenir compte des dépendances entre variables d'ordre supérieure à 2.

De façon similaire à ce qui précède, le principe est de déterminer les indices de Shapley de chaque lien de la FRC, pour avoir une mesure de leur importance relative par rapport à la sortie du modèle. De cette manière, les liens les moins importants peuvent simplement être élagués. Le critère peut également s'appliquer sur les poids du réseau, en estimant les indices de Shapley pour chaque poids d'une convolution.

Pour le calcul des indices de Shapley, nous avons appliqué l'algorithme de Ghorbani et. al. dans [GZ20]. Cet algorithme est conçu de sorte à pouvoir calculer les indices de Shapley d'éléments arbitraire au sein d'un réseau de neurones, il a donc l'avantage de pouvoir s'appliquer aussi bien aux poids d'un réseau qu'à des structures plus complexes, e.g. les liens de nos FRC. Cependant, nous ne sommes pas encore parvenus à obtenir de bons résultats avec ce critère. En effet, les FRC élaguées de cette manière montrent une performance finale trop faible par rapport aux autres. Nous soupçonnons que la qualité d'estimation des indices de Shapley ne soit beaucoup trop basse avec notre configuration. En particulier, le nombre d'itérations de l'algorithme de Ghorbani et. al. que nous sommes en mesure d'effectuer avec une seule GPU est très faible, i.e. de l'ordre d'une dizaine. Nous pensons néanmoins que l'application de ce processus pendant suffisamment d'itérations pourrait conduire à des résultats satisfaisant pour les FRC élaguées.

7.10 Limites et perspectives

Dans le cadre de ces expériences, nous avons mis au point un algorithme d'élagage adapté à la structure d'une CNF, en combinant une étape d'élagage de liens à l'élagage habituel des poids. Nous avons montré l'efficacité de coupler les deux types d'élagage lorsque le réseau est

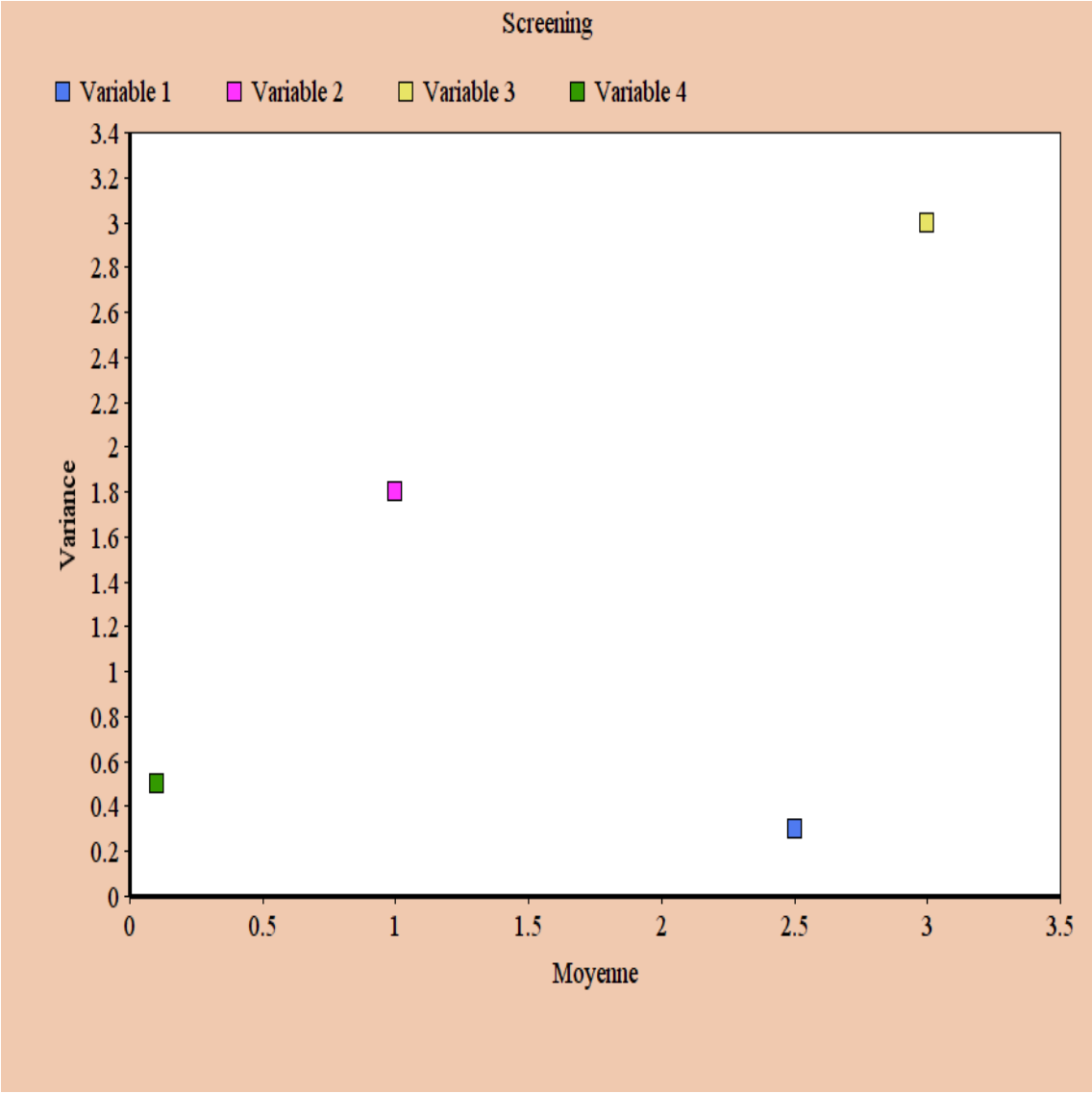


FIGURE 7.12 – Exemple de *screening* réalisé pour un modèle à 4 variables d'entrée.

élagué très tôt pendant son entraînement. Les expériences ont été menées sur quatre ensembles d'instances différents.

Nos expérimentations ont montré l'impact du bruit d'annotation sur deux indicateurs, l'un dépendants des données annotées (le critère de sensibilité), l'autre non (le critère de magnitude), dans le cadre applicatif de l'élagage de réseaux de neurones. Le bruit d'annotation impacte négativement les deux critères, mais de façon contre-intuitive, le critère de magnitude est celui pour lequel nous avons observé la plus grande baisse en efficacité. Par conséquent, que cela concerne l'élagage de réseaux de neurones ou leur évaluation, concevoir un indicateur indépendant des données dans l'objectif de le rendre robuste au bruit d'annotation n'est pas chose aisée. En effet, à partir du moment où le réseau apprend sur ces mêmes données bruitées, il devient conditionné par la présence du bruit d'annotation, ce qui se retrouve dans ses poids.

Nous pouvons cependant noter plusieurs perspectives d'amélioration du travail accompli. Premièrement, l'un des avantages des réseaux de neurones élagués est la réduction du temps

d'entraînement et d'inférence. Pour des raisons d'implantation, nous n'avons pu nous concentrer véritablement sur cet aspect, et présenter les gains en temps de calculs de nos FRC élaguées. En effet, même si une partie de notre élagage, i.e. l'élagage de liens, est implantée de sorte à améliorer le temps de calcul, l'implantation de l'étape d'élagage des poids ne l'améliore quant à elle en rien, du fait de contraintes techniques. Par conséquent, les estimations des temps d'entraînement de nos modèles sont très différentes de ce qui serait atteignable en réalité, bien que la diminution du temps de calcul reste visible en partie, grâce à notre élagage de lien. La prochaine étape serait donc d'implanter l'élagage de poids de façon efficace, pour pouvoir mesurer les véritables temps de calcul de nos FRC élaguées.

De plus, nos conclusions sont tirées des résultats obtenus par l'élagage de 18 FRC par ensemble d'instances, toutes avec des paramètres différents pour l'algorithme d'élagage. Étant donné le temps nécessaire pour entraîner et élaguer ces FRC, nous n'avons pu considérer d'autres paramétrages de notre algorithme, ou l'appliquer sur des FRC dimensionnées différemment, ou encore sur d'autres ensembles de données. Il serait donc intéressant de valider les conclusions tirées précédemment en élargissant le contexte de ces expériences.

Enfin, il serait pertinent d'ajouter au moins un troisième critère d'élagage à ces expériences : le critère aléatoire. De cette manière, nous pourrions établir une analyse plus complète, en nous intéressant au bénéfice apporté par l'application des critères de magnitude et de sensibilité, par rapport à un choix aléatoire des éléments à élaguer. L'inclusion d'autres critères d'élagage ajoutant un nombre non négligeable de FRC à entraîner et élaguer, et donc nécessitant du temps, nous pensons le mettre en place par la suite.

Conclusion générale

Durant cette thèse, nous avons eu pour objectif de mieux comprendre le problème posé par la présence d'instances mal annotées lors d'une procédure de test en classification supervisée, et d'identifier des solutions permettant de limiter leur impact sur la qualité de l'évaluation portée sur les classifieurs.

L'évaluation empirique fait partie intégrante du processus de construction d'un classifieur. L'estimation empirique de la performance d'un classifieur est effectuée via des procédures spécifiques, comme le *holdout* ou la validation croisée. Dans notre étude, nous nous sommes restreints au contexte de l'évaluation d'un classifieur binaire ou multi-classes par *holdout*. De plus, notre étude implique également une notion de confiance au sujet du résultat d'une évaluation. En apprentissage automatique, cette confiance est habituellement quantifiée par un intervalle de confiance, outil statistique sur lequel nous nous sommes par conséquent concentrés. Nous référons le lecteur au chapitre 2 pour plus de précisions.

Le bruit d'annotation (cf. chapitre 3) se réfère aux erreurs pouvant être commises par les annotateurs de données destinées à être utilisées en apprentissage automatique. Ces erreurs peuvent arriver pour différentes raisons : fatigue, inexpérience, ambiguïté... Trois différents modèles sont utilisés dans la littérature pour les représenter : bruit uniforme, bruit par classes, ou bruit par caractéristiques. Il existe de nombreuses études sur l'impact négatif du bruit d'annotation sur l'apprentissage d'un classifieur, et sur les techniques pour le limiter. Nous avons cependant remarqué que :

— la définition habituelle du bruit d'annotation se restreint au cas où, parmi les différentes interprétations possibles pour une donnée, une seule est vraie, ce qui entre en conflit avec la subjectivité inhérente à un grand nombre de problème de classification ;
— les travaux au sujet des conséquences négatives qu'il implique lors des procédures de test sont rares.

Le premier point présente le risque que le problème du bruit d'annotation soit minimisé, étant donné qu'on ne peut le considérer que de façon situationnelle, et non dans tout type d'application. En réponse à cela, le chapitre 4 couvre des problématiques éthiques et légales, et établit un lien entre le bruit d'annotation et le cadre législatif d'application d'un classifieur. Nous y avons par ailleurs défini les termes suivants, qui nous permettent de formuler de façon duale les conséquences du bruit d'annotation sur un classifieur :

— le biais de prédiction, i.e. comment le bruit d'annotation modifie le concept qu'apprend un classifieur, et par conséquent, comment il influence ses prédictions ;
— le biais d'évaluation , i.e. comment le bruit d'annotation modifie le jugement que nous portons sur les performances d'un classifieur, jugement qui se fonde sur les résultats de nos procédures de test.

Notre chapitre 5 compile différentes techniques établies dans la littérature, ou alors en développement, qui, à notre connaissance, peuvent aider à limiter l'apparition de ces deux types de biais. On peut tout d'abord citer plusieurs mesures préventives :

— mettre en place certaines bonnes pratiques lors de la phase de récolte et d'annotation des instances (cf. section 5.1.1), e.g. des annotateurs pour l'ensemble d'entraînement différents de ceux de l'ensemble de test ;
— appliquer au préalable des procédures permettant de nettoyer les données et éventuellement d'évaluer les annotateurs eux-mêmes (cf. section 5.1.4 ;
— choisir des algorithmes d'apprentissage connus pour leur robustesse face au bruit d'annotation.

On peut ensuite appliquer des procédures de test modifiées pour tenir compte de la présence de données mal annotées, ou alors changer radicalement notre façon d'aborder l'évaluation de classifieurs (cf. section 5.2 :

— évaluation pseudo-supervisée : estimer les performances d'un classifieur à partir d'une vérité-terrain de meilleure qualité, e.g. modifiée par un algorithme de nettoyage ;
— évaluation non supervisée : utilisant uniquement des données non annotées pour l'évaluation de classifieurs, à l'image de B. Raj [RSB11] présentant un test statistique pour comparer deux classifieurs sans avoir besoin de données annotées ;
— explicabilité : champ de recherche mettant en œuvre des techniques d'analyse de sensibilité, permettant de caractériser plus précisément le comportement d'un classifieur, en analysant quelles caractéristiques l'amènent à faire chaque prédiction ;
— preuve formelle : champ de recherche encore peu développé pour les systèmes construit par apprentissage automatique, promettant d'assurer d'un point de vue formelle que le système respecte certaines contraintes.

Décrire l'impact du bruit d'évaluation sur l'évaluation de classifieurs d'un point de vue formel est également essentiel. Nous en avons par conséquent fait l'objet du chapitre 6.

— Cet impact prend la forme d'un biais mathématique pour les estimateurs de mesures usuelles, et nous en avons obtenu l'expression en fonction des caractéristiques du bruit d'annotation pour la justesse, la précision, le rappel et la f-mesure.
— En ce qui concerne la confiance envers l'estimation faite, nous avons obtenu que, de façon surprenante, la présence d'un bruit d'annotation n'avait pour effet que le décalage des intervalles de confiance, et non l'augmentation de leur taille.
— Suite à cette étude théorique, nous avons mis au point un protocole expérimental contrôlable et reproductible permettant d'observer la différence d'efficacité entre une procédure d'évaluation ne supposant pas la présence de bruit d'annotation d'un côté, et de l'autre, des procédures qui au contraire la prennent en compte.
— Dans le cadre de ces expériences, nous avons développé un algorithme pour introduire de mauvaises annotations dans un ensemble d'instances tout en respectant la dépendance entre les caractéristiques de celles-ci et le bruit d'annotation résultant.

En dernier lieu, nous avons réalisé des travaux expérimentaux sur le sujet de l'élagage de réseaux de neurones en présence de bruit d'annotation, dans le but d'appuyer nos réflexions sur l'avantage d'indicateurs conçus pour être indépendants des annotations des instances. Ces expériences nous ont permis de conclure sur la difficulté de mettre au point de tels indicateurs, le danger étant que l'information sur laquelle ils se fondent peut aisément avoir un lien, même indirect, avec les étiquettes des instances. Dans notre expérience, par exemple, nous avons comparé deux indicateurs d'élagage documentés dans la littérature :

— le premier est dit indépendant des données, et se fonde uniquement sur les poids du réseaux ;
— le deuxième est dépendant des données, car il utilise leurs annotations.

Cependant, les poids du réseau étant progressivement déterminé par les données et leurs annotations au cours de l'entraînement, la présence d'un bruit d'annotation a négativement impacté

l'efficacité des deux indicateurs, sans distinction.

Annexe A

Augmentation de données

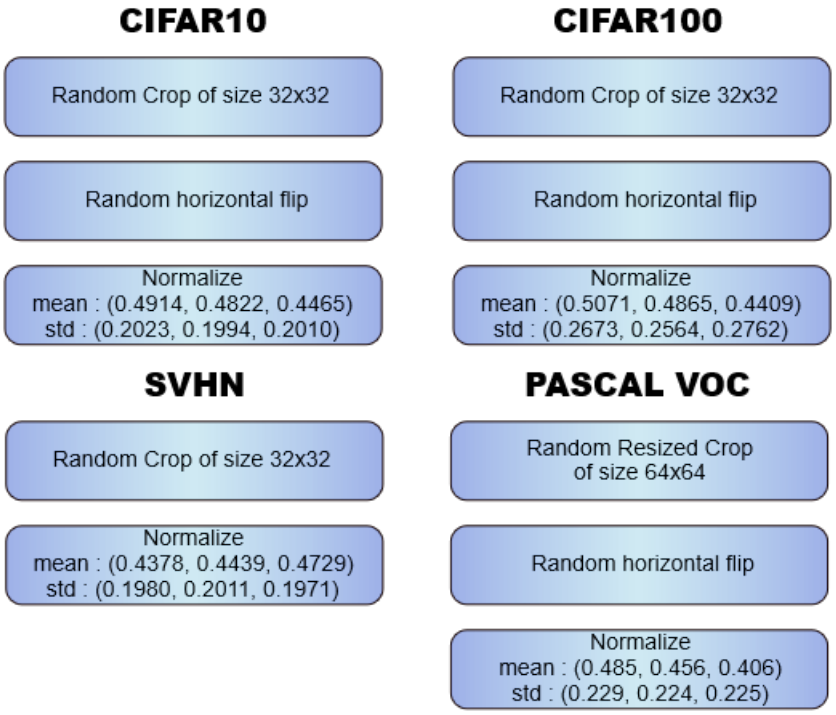


FIGURE A.1 – Transformations appliquées aux 4 ensembles de données utilisés dans le chapitre 7. Les noms des transformations, en anglais, correspondent aux noms de fonctions dans la bibliothèque PyTorch.

Annexe B

Liste des performances des FRC élaguées

Les résultats obtenus par les FRC élaguées sur les 4 ensembles de données sont listés dans ce qui suit, pour le cas non bruité et le cas bruité. Les noms attribués aux modèles représentent les paramètres de l'algorithme d'élagage qui leur a été appliqué :

- MAGN ou SBD pour le critère de magnitude ou de sensibilité ;
- RAP, ITE ou TAR pour une stratégie d'élagage rapide, itérative ou tardive ;
- le nombre correspond à la fraction élaguée de la FRC, en pourcentage ;
- REFERENCE désigne la FRC de référence pour un ensemble de donnée, i.e. celle qui n'a pas été élaguée.

En ce qui concerne les expériences d'élagage sur les ensembles bruités, la performance des FRC est tout de même mesurée sur les versions non bruitées des ensembles correspondant. Enfin, le tableau des résultats des FRC élaguées sur les ensembles non bruités présente deux performances pour chaque modèle (sauf ceux obtenus sur PASCALVOC). La performance à gauche correspond à la FRC élaguée avec une étape d'élagage de lien suivie d'un élagage des poids, i.e. la particularité de notre algorithme d'élagage, et celle à droite correspond à la FRC obtenue uniquement par des étapes d'élagage de poids, i.e. la manière conventionnelle d'élaguer un réseau.

CIFAR10 non bruité

Modèle	**Nb. Param.**	**Taux d'err. avec | sans LP (%)**
REFERENCE	4523402	6.44
MAGN-RAP-95	228611	09.55 | 10.93
SBD-RAP-95	-	09.94 | 10.53
MAGN-ITE-95	-	08.03 | 07.35
SBD-ITE-95	-	08.99 | 07.67
MAGN-TAR-95	-	08.56 | 07.32
SBD-TAR-95	-	08.69 | 08.57
MAGN-RAP-97	138194	10.49 | 12.78
SBD-RAP-97	-	11.75 | 11.31
MAGN-ITE-97	-	09.97 | 08.79
SBD-ITE-97	-	10.31 | 06.39
MAGN-TAR-97	-	08.39 | 08.25
SBD-TAR-97	-	09.31 | 10.40
MAGN-RAP-99	47778	15.33 | 26.56
SBD-RAP-99	-	14.32 | 20.33
MAGN-ITE-99	-	13.40 | 17.11
SBD-ITE-99	-	16.20 | 19.84
MAGN-TAR-99	-	12.02 | 17.83
SBD-TAR-99	-	14.48 | 21.35

CIFAR100 non bruité

Modèle	**Nb. Param.**	**Taux d'err. avec | sans LP (%)**
REFERENCE	4529252	27.54
MAGN-RAP-95	234461	36.41 | 38.82
SBD-RAP-95	-	37.08 | 37.27
MAGN-ITE-95	-	34.72 | 31.62
SBD-ITE-95	-	35.21 | 31.77
MAGN-TAR-95	-	33.26 | 29.95
SBD-TAR-95	-	33.73 | 33.07
MAGN-RAP-97	144044	40.98 | 42.52
SBD-RAP-97	-	39.03 | 41.56
MAGN-ITE-97	-	34.76 | 33.78
SBD-ITE-97	-	37.75 | 35.77
MAGN-TAR-97	-	34.81 | 34.09
SBD-TAR-97	-	36.39 | 37.00
MAGN-RAP-99	53628	48.71 | 57.56
SBD-RAP-99	-	47.96 | 54.64
MAGN-ITE-99	-	42.99 | 50.66
SBD-ITE-99	-	46.55 | 54.13
MAGN-TAR-99	-	42.03 | 49.09
SBD-TAR-99	-	45.92 | 53.01

SVHN non bruité

Modèle	**Nb. Param.**	**Taux d'err. avec | sans LP (%)**
REFERENCE	287594	03.56
MAGN-RAP-95	14997	05.01/07.90
SBD-RAP-95	-	05.20/06.40
MAGN-ITE-95	-	04.77/04.77
SBD-ITE-95	-	04.43/05.33
MAGN-TAR-95	-	05.00/04.82
SBD-TAR-95	-	04.47/06.03
MAGN-RAP-97	9258	06.71/18.94
SBD-RAP-97	-	05.74/14.25
MAGN-ITE-97	-	06.49/12.73
SBD-ITE-97	-	06.46/13.07
MAGN-TAR-97	-	04.90/11.46
SBD-TAR-97	-	05.90/20.14
MAGN-RAP-99	3519	14.45/80.41
SBD-RAP-99	-	17.77/80.41
MAGN-ITE-99	-	80.41/80.41
SBD-ITE-99	-	14.54/80.41
MAGN-TAR-99	-	17.94/80.41
SBD-TAR-99	-	19.88/80.41

PASCALVOC non bruité

Modèle	**Nb. Param.**	**Taux d'err. (%)**
REFERENCE	5376340	48.78
MAGN-RAP-95	271876	53.44
SBD-RAP-95	-	52.33
MAGN-ITE-95	-	51.22
SBD-ITE-95	-	51.69
MAGN-TAR-95	-	51.69
SBD-TAR-95	-	51.64
MAGN-RAP-97	164413	52.70
SBD-RAP-97	-	56.41
MAGN-ITE-97	-	55.46
SBD-ITE-97	-	50.85
MAGN-TAR-97	-	53.07
SBD-TAR-97	-	52.70
MAGN-RAP-99	56951	59.27
SBD-RAP-99	-	84.16
MAGN-ITE-99	-	53.92
SBD-ITE-99	-	54.71
MAGN-TAR-99	-	54.24
SBD-TAR-99	-	55.14

CIFAR10 bruité à 10%

Modèle	**Nb. Param.**	**Taux d'err. (%)**
REFERENCE	4523402	13.1
MAGN-RAP-95	228611	14.37
SBD-RAP-95	-	13.81
MAGN-ITE-95	-	13.49
SBD-ITE-95	-	14.24
MAGN-TAR-95	-	13.58
SBD-TAR-95	-	13.77
MAGN-RAP-97	138194	14.53
SBD-RAP-97	-	14.59
MAGN-ITE-97	-	14.05
SBD-ITE-97	-	13.66
MAGN-TAR-97	-	13.45
SBD-TAR-97	-	13.75
MAGN-RAP-99	47778	17.35
SBD-RAP-99	-	16.99
MAGN-ITE-99	-	16.02
SBD-ITE-99	-	17.96
MAGN-TAR-99	-	15.42
SBD-TAR-99	-	17

CIFAR100 bruité à 10%

Modèle	**Nb. Param.**	**Taux d'err. (%)**
REFERENCE	4529252	30.68
MAGN-RAP-95	234461	39.24
SBD-RAP-95	-	39
MAGN-ITE-95	-	36.20
SBD-ITE-95	-	35.96
MAGN-TAR-95	-	35.14
SBD-TAR-95	-	35.27
MAGN-RAP-97	144044	41.33
SBD-RAP-97	-	40.84
MAGN-ITE-97	-	37.79
SBD-ITE-97	-	37.56
MAGN-TAR-97	-	36.37
SBD-TAR-97	-	37.81
MAGN-RAP-99	53628	48.17
SBD-RAP-99	-	47.98
MAGN-ITE-99	-	41.66
SBD-ITE-99	-	44.72
MAGN-TAR-99	-	43.68
SBD-TAR-99	-	47.43

SVHN bruité à 10%

Modèle	**Nb. Param.**	**Taux d'err. (%)**
REFERENCE	287594	7.91
MAGN-RAP-95	14997	7.79
SBD-RAP-95	-	8.1
MAGN-ITE-95	-	7.81
SBD-ITE-95	-	7.65
MAGN-TAR-95	-	7.58
SBD-TAR-95	-	7.59
MAGN-RAP-97	9258	8.59
SBD-RAP-97	-	8.41
MAGN-ITE-97	-	8.50
SBD-ITE-97	-	9.34
MAGN-TAR-97	-	8.1
SBD-TAR-97	-	8.47
MAGN-RAP-99	3519	16.22
SBD-RAP-99	-	15.9
MAGN-ITE-99	-	52.38
SBD-ITE-99	-	80.41
MAGN-TAR-99	-	14.13
SBD-TAR-99	-	19.53

PASCALVOC bruité à 10%

Modèle	**Nb. Param.**	**Taux d'err. (%)**
REFERENCE	5376340	52.07
MAGN-RAP-95	271876	52.86
SBD-RAP-95	-	54.13
MAGN-ITE-95	-	50.79
SBD-ITE-95	-	50.26
MAGN-TAR-95	-	52.44
SBD-TAR-95	-	51.75
MAGN-RAP-97	164413	54.13
SBD-RAP-97	-	54.08
MAGN-ITE-97	-	51.11
SBD-ITE-97	-	52.44
MAGN-TAR-97	-	52.28
SBD-TAR-97	-	52.49
MAGN-RAP-99	56951	80.83
SBD-RAP-99	-	56.73
MAGN-ITE-99	-	53.44
SBD-ITE-99	-	55.35
MAGN-TAR-99	-	53.65
SBD-TAR-99	-	53.07

Bibliographie

[20a] *Label Errors in ML*. 8 avr. 2020. URL : `https://labelerrors.com/` (visité le 03/08/2021).

[20b] *PROPOSITION DE LOI CONSTITUTIONNELLE relative à la Charte de l'intelligence artificielle et des algorithmes*. 15 jan. 2020. URL : `http://www.assemblee-nationale.fr/dyn/15/textes/l15b2585_proposition-loi` (visité le 10/11/2020).

[78] *Loi n° 78-17 du 6 janvier 1978 relative à l'informatique, aux fichiers et aux libertés*. 6 jan. 1978. URL : `https://www.legifrance.gouv.fr/loda/id/LEGITEXT000006068624/2020-11-14/` (visité le 14/11/2020).

[AA99] Robert Audi et Paul Audi. *The Cambridge dictionary of philosophy*. T. 584. Cambridge university press Cambridge, 1999.

[ABN11] Michel Anteby, Philippe Bertreau et Charlotte Newman. « ProPublica ». In : *Harvard Business School Organizational Behavior Unit Case* 410-140 (2011).

[AL13] Wenjuan An et Mangui Liang. « Fuzzy support vector machine based on within-class scatter for classification problems with outliers or noises ». In : *Neurocomputing* 110 (2013), p. 101-110.

[AL88] Dana Angluin et Philip Laird. « Learning from noisy examples ». In : *Machine Learning* 2.4 (1988), p. 343-370.

[AM10] Joaquín Abellán et Andrés R Masegosa. « Bagging decision trees on data sets with classification noise ». In : *International Symposium on Foundations of Information and Knowledge Systems*. Springer. 2010, p. 248-265.

[Arr+16] Leila Arras et al. « Explaining predictions of non-linear classifiers in nlp ». In : *arXiv preprint arXiv :1606.07298* (2016).

[Arr+17] Leila Arras et al. « " What is relevant in a text document ?" : An interpretable machine learning approach ». In : *PloS one* 12.8 (2017), e0181142.

[Art17] Ron Artstein. « Inter-annotator agreement ». In : *Handbook of linguistic annotation*. Springer, 2017, p. 297-313.

[Ary+19] Vijay Arya et al. « One explanation does not fit all : A toolkit and taxonomy of ai explainability techniques ». In : *arXiv preprint arXiv :1909.03012* (2019).

[Asi04] Isaac Asimov. *I, robot*. Spectra, 2004.

[Bac+15] Sebastian Bach et al. « On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation ». In : *PloS one* 10.7 (2015), e0130140.

[Bae+10] David Baehrens et al. « How to explain individual classification decisions ». In : *The Journal of Machine Learning Research* 11 (2010), p. 1803-1831.

[Bal+13] Daniel Balouek et al. « Adding Virtualization Capabilities to the Grid'5000 Testbed ». In : *Cloud Computing and Services Science*. Sous la dir. d'Ivan I. Ivanov et al. T. 367. Communications in Computer and Information Science. Springer International Publishing, 2013, p. 3-20.

[BF+96] Carla E Brodley, Mark A Friedl et al. « Identifying and eliminating mislabeled training instances ». In : *Proceedings of the National Conference on Artificial Intelligence*. Citeseer. 1996, p. 799-805.

[BF96] Carla E Brodley et Mark A Friedl. « Improving automated land cover mapping by identifying and eliminating mislabeled observations from training data ». In : *IGARSS'96. 1996 International Geoscience and Remote Sensing Symposium*. T. 2. IEEE. 1996, p. 1379-1381.

[BF99] Carla E Brodley et Mark A Friedl. « Identifying mislabeled training data ». In : *Journal of artificial intelligence research* 11 (1999), p. 131-167.

[BGO09] Charles Bouveyron, Stephane Girard et Madalina Olteanu. « Supervised classification of categorical data with uncertain labels for DNA barcoding. » In : *ESANN*. Citeseer. 2009.

[BH02] Richard J Bolton et David J Hand. « Statistical fraud detection : A review ». In : *Statistical science* (2002), p. 235-249.

[Bir+20] Sarah Bird et al. « Fairlearn : A toolkit for assessing and improving fairness in AI ». In : *Microsoft, Tech. Rep. MSR-TR-2020-32* (2020).

[BJ10] Yingtao Bi et Daniel R Jeske. « The efficiency of logistic regression compared to normal discriminant analysis under class-conditional classification noise ». In : *Journal of Multivariate Analysis* 101.7 (2010), p. 1622-1637.

[BK09] Eyal Beigman et Beata Beigman Klebanov. « Learning with annotation noise ». In : *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*. 2009, p. 280-287.

[Bou+07] Sylvain Bouix et al. « On evaluating brain tissue classifiers without a ground truth ». In : *Neuroimage* 36.4 (2007), p. 1207-1224.

[BPP17] Rafael Ballester-Ripoll, Enrique G Paredes et Renato Pajarola. « Tensor Approximation of Advanced Metrics for Sensitivity Analysis ». In : *arXiv preprint arXiv :1712.01633* (2017).

[Bra00] Thorsten Brants. « Inter-annotator Agreement for a German Newspaper Corpus. » In : *LREC*. 2000.

[Bru18] Matthew Adam Bruckner. « The promise and perils of algorithmic lenders' use of big data ». In : *Chi.-Kent L. Rev.* 93 (2018), p. 3.

[Car08] Bob Carpenter. « Multilevel bayesian models of categorical data annotation ». In : *Unpublished manuscript* 17.122 (2008), p. 45-50.

[CCS07] Francesca Campolongo, Jessica Cariboni et Andrea Saltelli. « An effective screening design for sensitivity analysis of large models ». In : *Environmental modelling & software* 22.10 (2007), p. 1509-1518.

[CH79] David Roxbee Cox et David Victor Hinkley. *Theoretical statistics*. CRC Press, 1979.

[Cha12] Michel Chauvière. *L'intelligence sociale en danger : chemins de résistance et propositions*. La Découverte, 2012.

[CHS07] Sébastien Cuendet, Dilek Hakkani-Tür et Elizabeth Shriberg. « Automatic labeling inconsistencies detection and correction for sentence unit segmentation in conversational speech ». In : *International Workshop on Machine Learning for Multimodal Interaction*. Springer. 2007, p. 144-155.

[CI18] Miguel Carreira-Perpinan et Yerlan Idelbayev. « "Learning-Compression" Algorithms for Neural Net Pruning ». In : juin 2018, p. 8532-8541.

[CK09] Gordon V Cormack et Aleksander Kolcz. « Spam filter evaluation with imprecise ground truth ». In : *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*. 2009, p. 604-611.

[CM00] M Chavance et R Manfredi. « Modélisation d'observations incomplètes ». In : *Revue d'épidémiologie et de santé publique* 48.4 (2000), p. 389-400.

[CM84] Raj S Chhikara et Jim McKeon. « Linear discriminant analysis with misallocation in training samples ». In : *Journal of the American Statistical Association* 79.388 (1984), p. 899-906.

[DC04] Sarah Jane Delany et Pádraig Cunningham. « An analysis of case-base editing in a spam filtering system ». In : *European Conference on Case-Based Reasoning*. Springer. 2004, p. 128-141.

[Den00] Thierry Denoeux. « A neural network classifier based on Dempster-Shafer theory ». In : *IEEE Transactions on Systems, Man, and Cybernetics-Part A : Systems and Humans* 30.2 (2000), p. 131-150.

[Den08] Thierry Denoeux. « A k-nearest neighbor classification rule based on Dempster-Shafer theory ». In : *Classic works of the Dempster-Shafer theory of belief functions*. Springer, 2008, p. 737-760.

[Die00] Thomas G Dietterich. « An experimental comparison of three methods for constructing ensembles of decision trees : Bagging, boosting, and randomization ». In : *Machine learning* 40.2 (2000), p. 139-157.

[Die95] Tom Dietterich. « Overfitting and undercomputing in machine learning ». In : *ACM computing surveys (CSUR)* 27.3 (1995), p. 326-327.

[DS79] Alexander Philip Dawid et Allan M Skene. « Maximum likelihood estimation of observer error-rates using the EM algorithm ». In : *Journal of the Royal Statistical Society : Series C (Applied Statistics)* 28.1 (1979), p. 20-28.

[DSB17] Derek Doran, Sarah Schulz et Tarek R Besold. « What does explainable AI really mean ? A new conceptualization of perspectives ». In : *arXiv preprint arXiv :1710.00794* (2017).

[DU09] Weiwei Du et Kiichi Urahama. « Error-correcting semi-supervised learning with mode-filter on graphs ». In : *2009 IEEE 12th International Conference on Computer Vision Workshops, ICCV Workshops*. IEEE. 2009, p. 2095-2100.

[Efr82] Bradley Efron. *The jackknife, the bootstrap and other resampling plans*. SIAM, 1982.

[Esk00] Eleazar Eskin. « Detecting errors within a corpus using anomaly detection ». In : *1st Meeting of the North American Chapter of the Association for Computational Linguistics*. 2000.

[ESP06] Neamat El Gayar, Friedhelm Schwenker et Günther Palm. « A study of the robustness of KNN classifiers trained using soft labels ». In : *IAPR Workshop on Artificial Neural Networks in Pattern Recognition*. Springer. 2006, p. 67-80.

[ETS01] Cigdem Eroglu Erdem, A Murat Tekalp et Bülent Sankur. « Metrics for performance evaluation of video object segmentation and tracking without ground-truth ». In : *Proceedings 2001 International Conference on Image Processing (Cat. No. 01CH37205)*. T. 2. IEEE. 2001, p. 69-72.

[Eve+] M. Everingham et al. *The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results*. http ://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html.

[FC18] Jonathan Frankle et Michael Carbin. « The lottery ticket hypothesis : Finding sparse, trainable neural networks ». In : *arXiv preprint arXiv :1803.03635* (2018).

[Fel+14] Paul Felt et al. « Momresp : A Bayesian Model for Multi-Annotator Document Labeling. » In : *LREC*. 2014, p. 3704-3711.

[FJR15] Matt Fredrikson, Somesh Jha et Thomas Ristenpart. « Model inversion attacks that exploit confidence information and basic countermeasures ». In : *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*. 2015, p. 1322-1333.

[FL17a] Maksym Fedorchuk et Bart Lamiroy. « Binary classifier evaluation without ground truth ». In : *2017 Ninth International Conference on Advances in Pattern Recognition (ICAPR)*. IEEE. 2017, p. 1-6.

[FL17b] Maksym Fedorchuk et Bart Lamiroy. « Statistic metrics for evaluation of binary classifiers without ground-truth ». In : *2017 IEEE First Ukraine Conference on Electrical and Computer Engineering (UKRCON)*. IEEE. 2017, p. 1066-1071.

[FMN10] Annalisa Franco, Davide Maltoni et Loris Nanni. « Data pre-processing through reward–punishment editing ». In : *Pattern Analysis and Applications* 13.4 (2010), p. 367-381.

[FNR12] Karën Fort, Adeline Nazarenko et Sophie Rosset. « Modeling the complexity of manual annotation tasks : a grid of analysis ». In : *International Conference on Computational Linguistics*. 2012, p. 895-910.

[FR21] Michael Felderer et Rudolf Ramler. *Quality Assurance for AI-based Systems : Overview and Challenges*. 2021. arXiv : `2102.05351 [cs.SE]`.

[FV14] Benoît Frénay et Michel Verleysen. « Classification in the Presence of Label Noise : A Survey ». In : *Neural Networks and Learning Systems, IEEE Transactions on* 25 (mai 2014), p. 845-869.

[Gat72] Geoffrey Gates. « The reduced nearest neighbor rule (corresp.) » In : *IEEE transactions on information theory* 18.3 (1972), p. 431-433.

[GG05] Cyril Goutte et Eric Gaussier. « A probabilistic interpretation of precision, recall and F-score, with implication for evaluation ». In : *European conference on information retrieval*. Springer. 2005, p. 345-359.

[GGB19] Mor Geva, Yoav Goldberg et Jonathan Berant. « Are we modeling the task or the annotator ? an investigation of annotator bias in natural language understanding datasets ». In : *arXiv preprint arXiv :1908.07898* (2019).

[GLD00] Dragan Gamberger, Nada Lavrac et Saso Dzeroski. « Noise detection and elimination in data preprocessing : experiments in medical domains ». In : *Applied Artificial Intelligence* 14.2 (2000), p. 205-223.

[GLD96] Dragan Gamberger, Nada Lavrač et Sašo Džeroski. « Noise elimination in inductive concept learning : A case study in medical diagnosis ». In : *International Workshop on Algorithmic Learning Theory*. Springer. 1996, p. 199-212.

[GLG99] Dragan Gamberger, Nada Lavrac et Ciril Groselj. « Experiments with noise filtering in a medical domain ». In : *ICML*. T. 99. 1999, p. 143-151.

[GM11] Sandrine Garcia et Sabine Montagne. « Pour une sociologie critique des dispositifs d'évaluation ». In : *Actes de la recherche en sciences sociales* 4 (2011), p. 4-15.

[Goe+18] Randy Goebel et al. « Explainable ai : the new 42 ? » In : *International cross-domain conference for machine learning and knowledge extraction*. Springer. 2018, p. 295-303.

[Goo+13] Ian J Goodfellow et al. « Maxout networks ». In : *arXiv preprint arXiv :1302.4389* (2013).

[Gur+19] Karthik S Gurumoorthy et al. « Efficient Data Representation by Selecting Prototypes with Importance Weights ». In : *2019 IEEE International Conference on Data Mining (ICDM)*. IEEE. 2019, p. 260-269.

[GW92] Anil Gaba et Robert L Winkler. « Implications of errors in survey data : a Bayesian model ». In : *Management Science* 38.7 (1992), p. 913-925.

[GZ09] Shesen Guo et Ganzhou Zhang. « Robot rights ». In : *Science* 323.5916 (2009), p. 876.

[GZ20] Amirata Ghorbani et James Zou. *Neuron Shapley : Discovering the Responsible Neurons*. 2020. arXiv : `2002.09815 [stat.ML]`.

[Hai96] Y Haitovsky. « Bayesian analysis of binary data subject to misclassification ». In : *Bayesian analysis in statistics and econometrics : Essays in honor of Arnold Zellner* 309 (1996), p. 67.

[Hal18] Jeff Hale. *7 Data Types : A Better Way to Think about Data Types for Machine Learning*. 2018. URL : `https://towardsdatascience.com/7-data-types-a-better-way-to-think-about-data-types-for-machine-learning-939fae99a689` (visité le 07/10/2020).

[Han+15] Song Han et al. « Learning both weights and connections for efficient neural network ». In : *Advances in neural information processing systems*. 2015, p. 1135-1143.

[Har68] Peter Hart. « The condensed nearest neighbor rule (corresp.) » In : *IEEE transactions on information theory* 14.3 (1968), p. 515-516.

[Haw04] Douglas M Hawkins. « The problem of overfitting ». In : *Journal of chemical information and computer sciences* 44.1 (2004), p. 1-12.

[Hay+20] Soufiane Hayou et al. *Pruning untrained neural networks : Principles and Analysis*. 2020. arXiv : `2002.08797 [stat.ML]`.

[Hen+16] Lisa Anne Hendricks et al. « Generating visual explanations ». In : *European Conference on Computer Vision*. Springer. 2016, p. 3-19.

[Hen13] Kagermann Henning. « Recommendations for implementing the strategic initiative INDUSTRIE 4.0 ». In : (2013).

[Hes00] Tom Heskes. « The use of being stubborn and introspective ». In : *Prerational Intelligence : Adaptive Behavior and Intelligent Systems Without Symbols and Logic, Volume 1, Volume 2 Prerational Intelligence : Interdisciplinary Perspectives on the Behavior of Natural and Artificial Systems, Volume 3*. Springer, 2000, p. 1184-1200.

[Hes94] Tom Heskes. « The use of being stubborn and introspective ». In : (1994).

[HH02] Ardo van den Hout et Peter GM van der Heijden. « Randomized response, statistical disclosure control and misclassificatio : a review ». In : *International Statistical Review* 70.2 (2002), p. 269-288.

[HM82] James A Hanley et Barbara J McNeil. « The meaning and use of the area under a receiver operating characteristic (ROC) curve. » In : *Radiology* 143.1 (1982), p. 29-36.

[Hov+13] Dirk Hovy et al. « Learning whom to trust with MACE ». In : *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics : Human Language Technologies*. 2013, p. 1120-1130.

[How+17] Andrew G Howard et al. « Mobilenets : Efficient convolutional neural networks for mobile vision applications ». In : *arXiv preprint arXiv :1704.04861* (2017).

[HP89] Stephen José Hanson et Lorien Y Pratt. « Comparing biases for minimal network construction with back-propagation ». In : *Advances in neural information processing systems*. 1989, p. 177-185.

[HRT04] Nicholas P Hughes, Stephen J Roberts et Lionel Tarassenko. « Semi-supervised learning of probabilistic models for ECG segmentation ». In : *The 26th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*. T. 1. IEEE. 2004, p. 434-437.

[HS93] Babak Hassibi et David G Stork. « Second order derivatives for network pruning : Optimal brain surgeon ». In : *Advances in neural information processing systems*. 1993, p. 164-171.

[HS96] Toshimitsu Homma et Andrea Saltelli. « Importance measures in global sensitivity analysis of nonlinear models ». In : *Reliability Engineering & System Safety* 52.1 (1996), p. 1-17.

[JGC95] Lawrence Joseph, Theresa W Gyorkos et Louis Coupal. « Bayesian estimation of disease prevalence and the parameters of diagnostic tests in the absence of a gold standard ». In : *American journal of epidemiology* 141.3 (1995), p. 263-272.

[JLM20] Odest Chadwicke Jenkins, Daniel Lopresti et Melanie Mitchell. « Next Wave Artificial Intelligence : Robust, Explainable, Adaptable, Ethical, and Accountable ». In : *arXiv preprint arXiv :2012.06058* (2020).

[Kat+17] Guy Katz et al. *Reluplex : An Efficient SMT Solver for Verifying Deep Neural Networks*. 2017. arXiv : `1702.01135 [cs.AI]`.

[KH+09] Alex Krizhevsky, Geoffrey Hinton et al. « Learning multiple layers of features from tiny images ». In : (2009).

[KH05] Immanuel Kant et Arthur Hannequin. *Critique de la raison pure*. F. Alcan, 1905.

[Kim09] Ji-Hyun Kim. « Estimating classification error rate : Repeated cross-validation, repeated hold-out and bootstrap ». In : *Computational statistics & data analysis* 53.11 (2009), p. 3735-3745.

[KKK16] Been Kim, Rajiv Khanna et Oluwasanmi O Koyejo. « Examples are not enough, learn to criticize ! criticism for interpretability ». In : *Advances in neural information processing systems*. 2016, p. 2280-2288.

[Kle+18] Jon Kleinberg et al. « Algorithmic fairness ». In : *Aea papers and proceedings*. T. 108. 2018, p. 22-27.

[Koh+95] Ron Kohavi et al. « A study of cross-validation and bootstrap for accuracy estimation and model selection ». In : *Ijcai*. T. 14. 2. Montreal, Canada. 1995, p. 1137-1145.

[Koh20] G. Kohs. *AlphaGo*. https ://www.youtube.com/watch ?v=WXuK6gekU1Y. 2020.

[Kow+19] Kamran Kowsari et al. « Text classification algorithms : A survey ». In : *Information* 10.4 (2019), p. 150.

[KPG11] Mirjam J. Knol, Wiebe R. Pestman et Diederick E. Grobbee. « The (mis)use of overlap of confidence intervals to assess effect modification ». In : *European Journal of Epidemiology* 26.4 (2011), p. 253-254.

[Kri+18] Krishnamurthy et al. *A Dual Approach to Scalable Verification of Deep Networks*. 2018. arXiv : `1803.06567 [cs.LG]`.

[KZ94] Yu Kharin et Eugene Zhuk. « Robustness in statistical pattern recognition under" contaminations" of training samples ». In : *Proceedings of the 12th IAPR International Conference on Pattern Recognition, Vol. 3-Conference C : Signal Processing (Cat. No. 94CH3440-5)*. T. 2. IEEE. 1994, p. 504-506.

[Lac66] Peter A Lachenbruch. « Discriminant analysis when the initial samples are misclassified ». In : *Technometrics* 8.4 (1966), p. 657-662.

[Lac79] Peter A Lachenbruch. « Note on initial misclassification effects on the quadratic discriminant function ». In : *Technometrics* 21.1 (1979), p. 129-132.

[Lam18] Lamine Diop and Jean Cupe. *Explainable AI : The data scientists' new challenge*. url : `https://towardsdatascience.com/explainable-ai-the-data-scientists-new-challenge-f7cac935a5b4`. Juin 2018.

[Lap+16] Sebastian Lapuschkin et al. « Analyzing classifiers : Fisher vectors and deep neural networks ». In : *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2016, p. 2912-2920.

[Lar+98] Jan Larsen et al. « Design of robust neural network classifiers ». In : *Proceedings of the 1998 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP'98 (Cat. No. 98CH36181)*. T. 2. IEEE. 1998, p. 1205-1208.

[LAT18] Namhoon Lee, Thalaiyasingam Ajanthan et Philip HS Torr. « Snip : Single-shot network pruning based on connection sensitivity ». In : *arXiv preprint arXiv :1810.02340* (2018).

[LC12] Vincent Labatut et Hocine Cherifi. « Accuracy measures for the comparison of classifiers ». In : *arXiv preprint arXiv :1207.3790* (2012).

[LDS90] Yann LeCun, John S Denker et Sara A Solla. « Optimal brain damage ». In : *Advances in neural information processing systems*. 1990, p. 598-605.

[Lee+15] Chen-Yu Lee et al. « Deeply-supervised nets ». In : *Artificial intelligence and statistics*. 2015, p. 562-570.

[LH15] Ming Liang et Xiaolin Hu. « Recurrent convolutional neural network for object recognition ». In : *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015, p. 3367-3375.

[Lin+04] Chun-fu Lin et al. « Training algorithms for fuzzy support vector machines with noisy data ». In : *Pattern recognition letters* 25.14 (2004), p. 1647-1656.

[LL17] Scott M Lundberg et Su-In Lee. « A unified approach to interpreting model predictions ». In : *Advances in neural information processing systems*. 2017, p. 4765-4774.

[LMW12] Christopher H Lin, Mausam Mausam et Daniel S Weld. « Dynamically switching between synergistic workflows for crowdsourcing ». In : *Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence*. 2012, p. 87-93.

[LMZ02] Stéphane Lallich, Fabrice Muhlenbach et Djamel A Zighed. « Improving classification by removing or relabeling mislabeled instances ». In : *International Symposium on Methodologies for Intelligent Systems*. Springer. 2002, p. 5-15.

[LS01] N Lawrence et Bernhard Schölkopf. « Estimating a kernel fisher discriminant in the presence of label noise ». In : *18th International Conference on Machine Learning (ICML 2001)*. Morgan Kaufmann. 2001, p. 306-306.

[LS03] Chuck P Lam et David G Stork. « Evaluating classifiers by means of test data with noisy labels ». In : *IJCAI*. T. 3. 2003, p. 513-518.

[LS05] Chuck P Lam et David G Stork. « Toward Optimal Labeling Strategy under Multiple Unreliable Labelers. » In : *AAAI Spring Symposium : Knowledge Collection from Volunteer Contributors*. 2005, p. 42-47.

[Lur+20] Michal Luria et al. « Medieval Robots : The Role of Historical Automata in the Design of Future Robots ». In : *Companion Publication of the 2020 ACM Designing Interactive Systems Conference*. 2020, p. 191-195.

[Mad+20] Michael A Madaio et al. « Co-designing checklists to understand organizational challenges and opportunities around fairness in ai ». In : *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*. 2020, p. 1-14.

[Mal17] Stephane Mallat. *Science des données*. `https://www.college-de-france.fr/site/stephane-mallat/inaugural-lecture-2018-01-11-18h00.htm`. 2017.

[Mas10] Gaston Maspero. « RELIGIOUS ARCHITECTURE ». In : *Manual of Egyptian Archaeology and Guide to the Study of Antiquities in Egypt : For the Use of Students and Travellers*. Sous la dir. d'Amelia B.Translator Edwards. Cambridge Library Collection - Egyptology. Cambridge University Press, 2010, p. 47-110.

[MBN06] Andrea Malossini, Enrico Blanzieri et Raymond T Ng. « Detecting potential labeling errors in microarrays by data perturbation ». In : *Bioinformatics* 22.17 (2006), p. 2114-2121.

[McL72] GJ McLachlan. « Asymptotic results for discriminant analysis when the initial samples are misclassified ». In : *Technometrics* 14.2 (1972), p. 415-422.

[McN47] Quinn McNemar. « Note on the sampling error of the difference between correlated proportions or percentages ». In : *Psychometrika* 12.2 (1947), p. 153-157.

[Mir+09] André LB Miranda et al. « Use of classification algorithms in noise detection and elimination ». In : *International Conference on Hybrid Artificial Intelligence Systems*. Springer. 2009, p. 417-424.

[Mor91] Max D Morris. « Factorial sampling plans for preliminary computational experiments ». In : *Technometrics* 33.2 (1991), p. 161-174.

[MS05] Gonzalo Martínez-Muñoz et Alberto Suárez. « Switching class labels to generate classification ensembles ». In : *Pattern Recognition* 38.10 (2005), p. 1483-1494.

[MS13] Naresh Manwani et PS Sastry. « Noise tolerance under risk minimization ». In : *IEEE transactions on cybernetics* 43.3 (2013), p. 1146-1151.

[MW09] Robin Murphy et David D Woods. « Beyond Asimov : the three laws of responsible robotics ». In : *IEEE intelligent systems* 24.4 (2009), p. 14-20.

[NAM21] Curtis G. Northcutt, Anish Athalye et Jonas Mueller. *Pervasive Label Errors in Test Sets Destabilize Machine Learning Benchmarks*. 2021. arXiv : `2103.14749 [stat.ML]`.

[Net+11] Yuval Netzer et al. « Reading digits in natural images with unsupervised feature learning ». In : (2011).

[Nis09] Helen Nissenbaum. *Privacy in context : Technology, policy, and the integrity of social life*. Stanford University Press, 2009.

[NLF99] Hwee Tou Ng, Chung Yong Lim et Shou King Foo. « A case study on inter-annotator agreement for word sense disambiguation ». In : *SIGLEX99 : Standardizing Lexical Resources*. 1999.

[NOF10] David F Nettleton, Albert Orriols-Puig et Albert Fornells. « A study of the effect of different types of noise on the precision of supervised learning techniques ». In : *Artificial intelligence review* 33.4 (2010), p. 275-306.

[NR10] Stefanie Nowak et Stefan Rüger. « How reliable are annotations via crowdsourcing : a study about inter-annotator agreement for multi-label image annotation ». In : *Proceedings of the international conference on Multimedia information retrieval*. 2010, p. 557-566.

[OY97] Seishi Okamoto et Nobuhiro Yugami. « An average-case analysis of the k-nearest neighbor classifier for noisy domains ». In : *IJCAI (1)*. 1997, p. 238-245.

[Pap+16] Nicolas Papernot et al. « Towards the science of security and privacy in machine learning ». In : *arXiv preprint arXiv :1611.03814* (2016).

[Pas+17] Adam Paszke et al. « Automatic differentiation in PyTorch ». In : (2017).

[Pec+06] Mykola Pechenizkiy et al. « Class noise and supervised learning in medical domains : The effect of feature extraction ». In : *19th IEEE symposium on computer-based medical systems (CBMS'06)*. IEEE. 2006, p. 708-713.

[Pen+11] Yi Peng et al. « FAMCDM : A fusion approach of MCDM methods to rank multiclass classification algorithms ». In : *Omega* 39.6 (2011), p. 677-689.

[Per89] Philippe Perrenoud. « Vers une sociologie de l'évaluation ». In : *Bulletin de l'Association des enseignants et chercheurs en éducation* 6 (1989), p. 19-31.

[Pic11] Joseph P Pickett. « Empirical ». In : *The American Heritage Dictionary of the English Language* (2011).

[Pié54] Henri Piéron. « Vocabulaire de la Psychologie, collaboration de l'Association des Travailleurs scientifiques ». In : (1954).

[PR13] Jeff Pasternack et Dan Roth. « Latent credibility analysis ». In : *Proceedings of the 22nd international conference on World Wide Web*. 2013, p. 1009-1020.

[Qui86] J. Ross Quinlan. « Induction of decision trees ». In : *Machine learning* 1.1 (1986), p. 81-106.

[Rac18] Rachel Thomas. *Analyzing & Preventing Unconscious Bias in Machine Learning*. url : `https://www.infoq.com/presentations/unconscious-bias-machine-learning/`. Juin 2018.

[Ram+19] Vivek Ramanujan et al. « What's Hidden in a Randomly Weighted Neural Network ? » In : *arXiv preprint arXiv :1911.13299* (2019).

[RB07] Umaa Rebbapragada et Carla E Brodley. « Class noise mitigation through instance weighting ». In : *European conference on machine learning*. Springer. 2007, p. 708-715.

[RH05] Charles Ramsey et Alan Hewitt. « A Methodology for Assessing Sample Representativeness ». In : *Environmental Forensics - ENVIRON FORENSICS* 6 (mar. 2005), p. 71-75.

[Rin97] Thomas C Rindfleisch. « Privacy, information technology, and health care ». In : *Communications of the ACM* 40.8 (1997), p. 92-100.

[Ros58] Frank Rosenblatt. « The perceptron : a probabilistic model for information storage and organization in the brain. » In : *Psychological review* 65.6 (1958), p. 386.

[RR18] R. Ros et P. Runeson. « Continuous Experimentation and A/B Testing : A Mapping Study ». In : *2018 IEEE/ACM 4th International Workshop on Rapid Continuous Software Engineering (RCoSE)*. 2018, p. 35-41.

[RSB11] Bhiksha Raj, Rita Singh et James Baker. « A paired test for recognizer selection with untranscribed data ». In : *2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE. 2011, p. 5676-5679.

[RSG16] Marco Tulio Ribeiro, Sameer Singh et Carlos Guestrin. « " Why should I trust you ?" Explaining the predictions of any classifier ». In : *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*. 2016, p. 1135-1144.

[RY12] Vikas C Raykar et Shipeng Yu. « Eliminating spammers and ranking annotators for crowdsourced labeling tasks ». In : *The Journal of Machine Learning Research* 13.1 (2012), p. 491-518.

[Sak20] Yaser Sakkaf. *An Overview of Pruning Neural Networks using PyTorch*. `https://medium.com/@yasersakkaf123/pruning-neural-networks-using-pytorch-3bf03d16a76e`. 2020.

[Sam+16] Wojciech Samek et al. « Evaluating the visualization of what a deep neural network has learned ». In : *IEEE transactions on neural networks and learning systems* 28.11 (2016), p. 2660-2673.

[Sei+14] Chris Seiffert et al. « An empirical study of the classification performance of learners on imbalanced and noisy software quality data ». In : *Information Sciences* 259 (2014), p. 571-595.

[Sha53] Lloyd S Shapley. « A value for n-person games ». In : *Contributions to the Theory of Games* 2.28 (1953), p. 307-317.

[She18] Mary Wollstonecraft Shelley. *Frankenstein : The 1818 Text*. Penguin, 2018.

[Sho+17] Reza Shokri et al. « Membership inference attacks against machine learning models ». In : *2017 IEEE Symposium on Security and Privacy (SP)*. IEEE. 2017, p. 3-18.

[Sig+02] Sigurdur Sigurdsson et al. « Outlier estimation and detection application to skin lesion classification. » In : *IEEE INTERNATIONAL CONFERENCE ON ACOUSTICS SPEECH AND SIGNAL PROCESSING*. T. 1. IEEE ; 1999. 2002, p. I-1049.

[Sin+18] Gagandeep Singh et al. « Fast and Effective Robustness Certification ». In : *Advances in Neural Information Processing Systems*. Sous la dir. de S. Bengio et al. T. 31. Curran Associates, Inc., 2018, p. 10802-10813. url : https://proceedings.neurips.cc/paper/2018/file/f2f446980d8e971ef3da97af089481c3-Paper.pdf.

[SJS06] Marina Sokolova, Nathalie Japkowicz et Stan Szpakowicz. « Beyond accuracy, F-score and ROC : a family of discriminant measures for performance evaluation ». In : *Australasian joint conference on artificial intelligence*. Springer. 2006, p. 1015-1021.

[SL09] Marina Sokolova et Guy Lapalme. « A systematic analysis of performance measures for classification tasks ». In : *Information processing & management* 45.4 (2009), p. 427-437.

[Sla+20] Dylan Slack et al. « Fooling lime and shap : Adversarial attacks on post hoc explanation methods ». In : *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society*. 2020, p. 180-186.

[SLH11] José A Sáez, Julián Luengo et Francisco Herrera. « Fuzzy rule based classification systems versus crisp robust learners trained in presence of class noise's effects : a case of study ». In : *2011 11th International Conference on Intelligent Systems Design and Applications*. IEEE. 2011, p. 1229-1234.

[SLH16] José A Sáez, Julián Luengo et Francisco Herrera. « Evaluating the classifier behavior with noisy data considering performance and robustness : The equalized loss of accuracy measure ». In : *Neurocomputing* 176 (2016), p. 26-35.

[SM19] Jake Silberg et James Manyika. « Notes from the AI frontier : Tackling bias in AI (and in humans) ». In : *McKinsey Global Institute (June 2019)* (2019).

[Smy+95] Padhraic Smyth et al. « Inferring ground truth from subjective labelling of venus images ». In : *Advances in neural information processing systems*. 1995, p. 1085-1092.

[Smy96] Padhraic Smyth. « Bounds on the mean classification error rate of multiple experts ». In : *Pattern Recognition Letters* 17.12 (1996), p. 1253-1257.

[Sob01] Ilya M Sobol. « Global sensitivity indices for nonlinear mathematical models and their Monte Carlo estimates ». In : *Mathematics and computers in simulation* 55.1-3 (2001), p. 271-280.

[SPF97] José Salvador Sánchez, Filiberto Pla et Francesc J Ferri. « Prototype selection for the nearest neighbour rule through proximity graphs ». In : *Pattern Recognition Letters* 18.6 (1997), p. 507-513.

[SR15] Takaya Saito et Marc Rehmsmeier. « The precision-recall plot is more informative than the ROC plot when evaluating binary classifiers on imbalanced datasets ». In : *PloS one* 10.3 (2015), e0118432.

[Sun+07] Jiang-wen Sun et al. « Identifying and correcting mislabeled training instances ». In : *Future generation communication and networking (FGCN 2007)*. T. 1. IEEE. 2007, p. 244-250.

[SV16] Shreyas Saxena et Jakob Verbeek. « Convolutional neural fabrics ». In : *Advances in Neural Information Processing Systems*. 2016, p. 4053-4061.

[SVZ13] Karen Simonyan, Andrea Vedaldi et Andrew Zisserman. « Deep inside convolutional networks : Visualising image classification models and saliency maps ». In : *arXiv preprint arXiv :1312.6034* (2013).

[Swa+04] Tim B Swartz et al. « Bayesian identifiability and misclassification in multinomial data ». In : *Canadian Journal of Statistics* 32.3 (2004), p. 285-302.

[SWM17] Wojciech Samek, Thomas Wiegand et Klaus-Robert Müller. « Explainable artificial intelligence : Understanding, visualizing and interpreting deep learning models ». In : *arXiv preprint arXiv :1708.08296* (2017).

[Ten00] Choh Man Teng. « Evaluating noise correction ». In : *Pacific Rim International Conference on Artificial Intelligence*. Springer. 2000, p. 188-198.

[Ten01] Choh-Man Teng. « A Comparison of Noise Handling Techniques. » In : *FLAIRS Conference*. 2001, p. 269-273.

[Ter+18] Milagro Teruel et al. « Increasing argument annotation reproducibility by using inter-annotator agreement to improve guidelines ». In : *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*. 2018.

[TGE11] Mahdi Tabassian, Reza Ghaderi et Reza Ebrahimpour. « Knitted fabric defect classification for uncertain labels based on Dempster–Shafer theory of evidence ». In : *Expert Systems with Applications* 38.5 (2011), p. 5259-5267.

[Tho+08] Jaree Thongkam et al. « Support vector machine for outlier detection in breast cancer survivability prediction ». In : *Asia-Pacific Web Conference*. Springer. 2008, p. 99-109.

[Tru15] Elly Rachel Truitt. *Medieval robots : mechanism, magic, nature, and art*. University of Pennsylvania Press, 2015.

[Tur09] Alan M Turing. « Computing machinery and intelligence ». In : *Parsing the turing test*. Springer, 2009, p. 23-65.

[Vér98] Jean Véronis. « A study of polysemy judgements and inter-annotator agreement ». In : *Programme and advanced papers of the Senseval workshop*. Citeseer. 1998, p. 2-4.

[Vil+18] Cédric Villani et al. *Donner un sens à l'intelligence artificielle : pour une stratégie nationale et européenne*. Conseil national du numérique, 2018. Chap. 5.

[VV17] Paul Voigt et Axel Von dem Bussche. « The eu general data protection regulation (gdpr) ». In : *A Practical Guide, 1st Ed., Cham : Springer International Publishing* (2017).

[Wak18] Daisuke Wakabayashi. « Self-driving Uber car kills pedestrian in Arizona, where robots roam ». In : *The New York Times* 19 (2018).

[WD+11] Daniel S Weld, Peng Dai et al. « Human intelligence needs artificial intelligence ». In : *Workshops at the Twenty-Fifth AAAI Conference on Artificial Intelligence*. Citeseer. 2011.

[Wen+18] Tsui-Wei Weng et al. *Towards Fast Computation of Certified Robustness for ReLU Networks*. 2018. arXiv : `1804.09699 [stat.ML]`.

[WH02] Rory Wolfe et James Hanley. « If we're so different, why do we keep overlapping ? When 1 plus 1 doesn't make 2 ». In : *Cmaj* 166.1 (2002), p. 65-66.

[Whi+09] Jacob Whitehill et al. « Whose vote should count more : Optimal integration of labels from labelers of unknown expertise ». In : *Advances in neural information processing systems* 22 (2009), p. 2035-2043.

[Wit+16a] Ian H Witten et al. *Data Mining : Practical machine learning tools and techniques*. Morgan Kaufmann, 2016. Chap. 1.

[Wit+16b] Ian H Witten et al. *Data Mining : Practical machine learning tools and techniques*. Morgan Kaufmann, 2016. Chap. 5.

[WM00] D Randall Wilson et Tony R Martinez. « Reduction techniques for instance-based learning algorithms ». In : *Machine learning* 38.3 (2000), p. 257-286.

[WY19] Tzu-Tsung Wong et Po-Yang Yeh. « Reliable accuracy estimates from k-fold cross validation ». In : *IEEE Transactions on Knowledge and Data Engineering* (2019).

[WZG20] Chaoqi Wang, Guodong Zhang et Roger Grosse. « Picking winning tickets before training by preserving gradient flow ». In : *arXiv preprint arXiv :2002.07376* (2020).

[YCC10] Ainur Yessenalina, Yejin Choi et Claire Cardie. « Automatically generating annotator rationales to improve sentiment classification ». In : *Proceedings of the ACL 2010 Conference Short Papers*. 2010, p. 336-341.

[YS16] Sanjay Yadav et Sanyam Shukla. « Analysis of k-fold cross-validation over hold-out validation on colossal datasets for quality classification ». In : *2016 IEEE 6th International conference on advanced computing (IACC)*. IEEE. 2016, p. 78-83.

[Zha+09] Chen Zhang et al. « Methods for labeling error detection in microarrays based on the effect of data perturbation on the regression model ». In : *Bioinformatics* 25.20 (2009), p. 2708-2714.

[Zho+12] Dengyong Zhou et al. « Learning from the wisdom of crowds by minimax entropy ». In : *Advances in neural information processing systems* 25 (2012), p. 2195-2203.

[Zho+19] Hattie Zhou et al. « Deconstructing lottery tickets : Zeros, signs, and the supermask ». In : *arXiv preprint arXiv :1905.01067* (2019).

[ZL15] Reza Zafarani et Huan Liu. « Evaluation without ground truth in social media research ». In : *Communications of the ACM* 58.6 (2015), p. 54-60.

[ZRB06] Wensheng Zhang, Romdhane Rekaya et Keith Bertrand. « A method for predicting disease subtypes in presence of misclassification among training samples using gene expression : application to human breast cancer ». In : *Bioinformatics* 22.3 (2006), p. 317-325.

[ZWC03] Xingquan Zhu, Xindong Wu et Qijun Chen. « Eliminating class noise in large datasets ». In : *Proceedings of the 20th International Conference on Machine Learning (ICML-03)*. 2003, p. 920-927.

[ZWC06] Xingquan Zhu, Xindong Wu et Qijun Chen. « Bridging local and global data cleansing : Identifying class noise in large, distributed data datasets ». In : *Data mining and Knowledge discovery* 12.2 (2006), p. 275-308.

Résumé

Les récents progrès de l'intelligence artificielle ont permis de construire des systèmes autonomes dans presque tous les secteurs de la société humaine. Des voitures autonomes jusqu'à la police prédictive, des domaines pouvant impliquer des conséquences critiques commencent à mettre en application cette technologie. Devant cette précipitation, des questionnements éthiques et légaux apparaissent dans la communauté. Quelles sont les potentielles dérives d'une mauvaise conception d'une IA ? Comment s'en protéger ? En effet, celle-ci apprend une tâche en regardant comment nous, humains, l'accomplissons. Cela signifie que les erreurs que nous commettons peuvent influencer leur apprentissage. Concrètement, nous annotons manuellement des données, exemplifiant la tâche que nous voulons que la machine apprenne, et nous les lui présentons dans l'objectif qu'elle comprenne par elle-même la nature de cette tâche. Ces données contiennent cependant de potentielles erreurs commises lors de l'annotation : on parle de bruit d'annotation. Non seulement ces erreurs perturbent l'apprentissage de la machine, mais la situation est en réalité plus grave. Ces données erronées sont également utilisées pour vérifier si l'apprentissage s'est bien déroulé : l'évaluation du système est ainsi biaisée, et n'indique pas sa véritable performance. Dans cette thèse, nous nous concentrons sur l'étude de l'évaluation de systèmes de classification sur un ensemble de test contenant du bruit d'annotation. Nous modélisons le problème d'un point de vue théorique, et nous montrons en pratique l'impact négatif du bruit d'annotation lors d'une procédure de test, et les compromis à accepter pour s'en défaire.

Mots-clés: évaluation, bruit d'annotation, classifieur

Abstract

The recent progress of artificial intelligence allowed to build autonomous systems in almost every sector of human society. From autonomous cars to predictive policing, many critical domains are now welcoming machine learning technology. This has risen ethic and legal concern among the scientific community with respect to the danger of misconceptions in an AI. What are the risks ? How can we reduce them ? It is all the more concerning that recent examples of AI misuse can be found in the news, such as an autonomous car from Uber company that killed a pedestrian, or a face recognition systems published online by Google, that made racist predictions. Indeed, an AI cannot be perfect : it directly learns from us doing the task, and that includes our mistakes, which can impact its learning quality. Concretely, we manually label data that show how the task must be performed, and we give them to the machine for it to understand what is the nature of that task. These data contain however potential errors that we made during the labelling : this is named label noise. Label noise degrades the quality of the learning. The problem is even more serious when the testing data we use to make sure the machine learned correctly is also noisy. In that case, the evaluation is biased, and we do not see the true performance of the AI. This thesis focuses on studying the evaluation of classification systems on a test set containing label noise. We modelise the problem in a theoretical framework, and we show in practice the negative impact of label noise during classifier testing, together with the compromises we must make in order to dispose of it.

Keywords: evaluation, label noise, classifier