CHAPITRE 1

Classification et réseaux de neurones artificiels

Introduction

Ce chapitre présente le domaine de la classification [1] par réseaux de neurones artificiels. Les notions connexionnistes nécessaires à la bonne compréhension de cette thèse sont introduites et détaillées dans l'annexe A. Le modèle connexionniste que nous avons étudié et parallélisé est décrit (section 1.1) et évalué au travers de deux applications : la reconnaissance de formes manuscrites (section 1.2) et le problème des formes d'ondes (section 1.3).

La classification

Un classifieur est un système permettant de discriminer un ensemble en classes. L'homme utilisait des classifieurs bien avant l'avènement de l'ère informatique, notamment grâce à des systèmes mécaniques parfois très ingénieux ; du tamis du chercheur d'or au spectrographe de masse du physicien, en passant par la centrifugeuse du chimiste, les exemples ne manquent pas. L'ordinateur permet de faire plus que de séparer de petits cailloux du sable, l'informatique étend le champ de la classification à l'information. Dans le cadre de cette thèse, un classifieur désigne un algorithme capable d'associer des classes à des formes. Lors du processus de reconnaissance, des exemples sont présentés au classifieur et le système répond par la classe correspondante. Le classifieur peut rejeter la forme et ne pas donner de réponse en cas de doute.

Exemple des chiffres manuscrits

On peut, par exemple, utiliser un classifieur pour reconnaître des chiffres manuscrits. Lors du processus de reconnaissance, le classifieur reçoit l'image d'un chiffre et renvoie un entier entre 0 et 9. L'intérêt du refus de donner une réponse dépend de l'application. Par exemple, si le classifieur reconnaît les codes postaux, dans un

1. La littérature consacrée à la Reconnaissance De Formes (RDF) nomme « classement » le processus de reconnaissance [BB92], c'est-à-dire la recherche de la classe associée à une forme. Le terme « classification » fait alors référence à la création de nouvelles classes à partir d'un ensemble d'apprentissage dépourvu d'étiquettes. Nous utiliserons la terminologie connexionniste bien que le verbe « classer » respecte mieux la langue française que le verbe « classifier ».

centre de tri, une erreur ne peut entraîner qu'un léger retard dans l'acheminement du courrier. En revanche, si les chiffres sont lus sur des chèques, le classifieur se doit de donner une réponse sûre. Les chiffres jugés peu lisibles seront rejetés par le classifieur, et traités par un opérateur.

Classification et réseaux de neurones

Nous nous intéressons à la classification par réseaux de neurones artificiels. La classification est un domaine riche du connexionnisme. Nombre d'applications performantes ont vu le jour dans les laboratoires, ainsi que dans l'industrie. Ce transfert technologique est remarquable pour des travaux relevant de l'informatique fondamentale. Nous pouvons déjà donner une description fonctionnelle d'un classifieur neuronal (la notion de réseau de neurones artificiels est précisée en annexe). Une différence fondamentale entre un réseau de neurones artificiels et un algorithme traditionnel est le besoin d'un apprentissage.

Apprentissage supervisé

En effet, avant de pouvoir reconnaître une forme, un réseau de neurones doit passer par une phase d'apprentissage. Les exemples sont présentés au classifieur avec une information, une étiquette, qui précise à quelle classe l'exemple appartient. Le réseau apprend à associer exemples et étiquettes. Ce type d'apprentissage est dit « supervisé ».

Exemple

À titre d'exemple, pour apprendre à un enfant à reconnaître les chiffres manuscrits, un « superviseur » dessine des chiffres (des exemples) en précisant pour chacun le nom du chiffre (l'étiquette) : « ceci est un 1 », « ceci est un 2 », *etc.*

Apprentissage non supervisé

Un classifieur neuronal peut de plus apprendre de façon non supervisée. Cette fois, les exemples sont présentés seuls, c'est-à-dire sans étiquette. Le classifieur regroupe les exemples proches (au sens d'une mesure donnée). Chaque groupe est une classe qui est étiquetée *a posteriori*, par exemple par un expert humain.

Exemple (suite)

Si nous poursuivons notre exemple, un enfant qui découvre pour la première fois des chiens et des chats peut créer deux classes qui seront étiquetées ultérieurement. Les classes émergentes d'un apprentissage non supervisé dépendent de l'algorithme du classifieur et de ses paramètres, ainsi que des exemples présentés. Si l'enfant découvre des chiens de plusieurs races et un gros chat, le résultat peut être surprenant.

Processus de généralisation

À la suite d'un apprentissage (supervisé ou non), le classifieur est capable de reconnaître des formes qu'il n'a jamais rencontrées. Les exemples présentés étant nouveaux, le réseau de neurones n'utilise pas une association apprise « par cœur » pour répondre. Aussi, le processus de reconnaissance porte également le nom de « généralisation ».

Processus de test

Les exemples utilisés pour l'apprentissage et la généralisation composent respectivement, « l'ensemble d'apprentissage » et « l'ensemble de généralisation ». Le processus

de généralisation est représentatif des performances du modèle connexionniste testé si les deux ensembles sont totalement disjoints. Aussi, l'ensemble de généralisation est parfois appelé « ensemble de test ». Dans les expérimentations que nous avons réalisées, le classifieur généralise toujours sur de nouveaux exemples, aussi nous utilisons indifféremment les termes de « test » et de « généralisation ».

Processus de validaton

Notons que certains algorithmes d'apprentissage considèrent que le modèle a appris lorsque le taux de succès en généralisation atteint une valeur donnée. Il est alors nécessaire d'utiliser un nouvel ensemble d'exemples, disjoint des deux autres, pour évaluer les performances du modèle ; on parle alors de « validation ».

Le connexionnisme

Le connexionnisme est maintenant suffisamment développé pour supposer que le lecteur possède les notions de base. Aussi, la suite de ce chapitre traite avant tout du modèle mis en œuvre dans cette thèse. Cependant, afin de permettre à tous une bonne lecture, l'annexe A présente les bases du connexionnisme tout en restant centrée sur la classification. De plus, l'annexe présente exclusivement les notions nécessaires à la compréhension de la thèse :

- le neurone formel de McCulloch et Pitts (section A.1) ;
- le perceptron de Rosenblatt (section A.2) ;
- la notion de réseau de neurones (section A.3) et les trois grandes familles de réseaux (les réseaux à couches, les réseaux à connexions latérales, et les réseaux récurrents) ;
- la nécessité d'un apprentissage (section A.4) ;
- la notion de prototype (section A.5) ;
- l'apprentissage par compétition (section A.6), et plus particulièrement les cartes de Kohonen ainsi que l'apprentissage LVQ.

Le lecteur désirant en savoir plus peut se référer à l'article de Hush et Horne [HH93] qui introduit les principaux modèles. Enfin, une présentation rigoureuse du connexionnisme est donnée, par exemple, par les livres de Hassoun [Has95] et de Bishop [Bis95].

1.1 Classifieur neuronal incrémental

Cette thèse s'intéresse à un modèle de classifieur développé en 1991 par Azcarraga et Giacometti au Laboratoire de l'Informatique Fondamentale et d'Intelligence Artificielle (LIFIA) de l'Institut National Polytechnique de Grenoble (INPG). Un réseau incrémental est capable d'accroître le nombre de ses cellules de façon autonome durant le processus d'apprentissage. Ce modèle résout avec élégance le difficile

problème du dimensionnement du réseau (bien que le problème soit en fait partiellement déplacé sur le choix de bons paramètres). Ce modèle a été utilisé avec succès, notamment pour la classification de dessins au trait, et plus particulièrement pour la reconnaissance de chiffres manuscrits [AA92, Azc93].

1.1.1 Réseaux incrémentaux et littérature connexionniste

Le modèle NLS

La notion de réseau connexionniste incrémental n'est pas nouvelle. Un des premiers modèles présenté comme étant incrémental est le « *Nestor Learning System* » (NLS) de Reilly, Cooper et Elbaum [RCE82]. Les prototypes d'un réseau NLS sont créés si aucun prototype n'est assez proche de l'exemple à apprendre. L'exemple en question devient un nouveau prototype. En revanche, si des prototypes sont assez proches de l'exemple à apprendre mais que les plus proches ne sont pas de la classe de l'exemple (erreur de classification), la zone d'influence des prototypes gagnant à tord est réduite jusqu'à exclusion de l'exemple.

Notons que l'algorithme d'apprentissage ne modifie que la zone d'influence des prototypes (i.e. le rayon de l'hypersphère), en aucun cas les poids des prototypes ne sont modifiés. Bien sûr, si une zone d'influence est réduite, il est indispensable de présenter à nouveau la base d'exemples pour s'assurer que le prototype modifié représente encore les exemples déjà appris.

Le modèle GAL

Un deuxième modèle incrémental célèbre est le réseau « *Grow And Learn* » (GAL) de Alpaydin [Alp90a, Alp90b, Alp91]. Un réseau GAL crée un nouveau prototype à partir de l'exemple en entrée à chaque erreur de classification. Le zone d'influence des prototypes n'est jamais modifiée. L'utilisateur peut faire passer le réseau dans une phase dite « de sommeil ». Durant cette phase, un prototype déjà existant (choisi aléatoirement) est présenté au réseau. Ce prototype est ensuite retiré du réseau et présenté à nouveau. Si la réponse du réseau est peu affectée, le prototype est définitivement retiré.

Notons que cette technique « d'élagage » impose de présenter à nouveau la base d'exemples pour s'assurer que tous les exemples sont encore correctement classifiés. Aussi, plusieurs cycles d'apprentissage et de sommeil sont souvent nécessaires.

ART, cartes de Kohonen et LVQ

Enfin, il existe des versions incrémentales du modèle ART [CG87, CG88] (Neo-ART de Yin, Lengellé et Gaillard [YLG90]), des cartes de Kohonen [Koh82, Koh84] (voir [Xu90, XO90]) et de l'apprentissage LVQ [Koh90b] (DVQ de Poirier et Ferrieux [PF91, TP93]).

1.1.2 Architecture du classifieur incrémental

Le classifieur incrémental développé à l'INPG est un réseau de type *feedforward* à trois couches, le nombre de couches, le nombre de cellules par couche, et l'interconnexion des cellules, sont pleinement déterminés par le problème traité. La figure 1.1 donne un exemple de topologie possible.

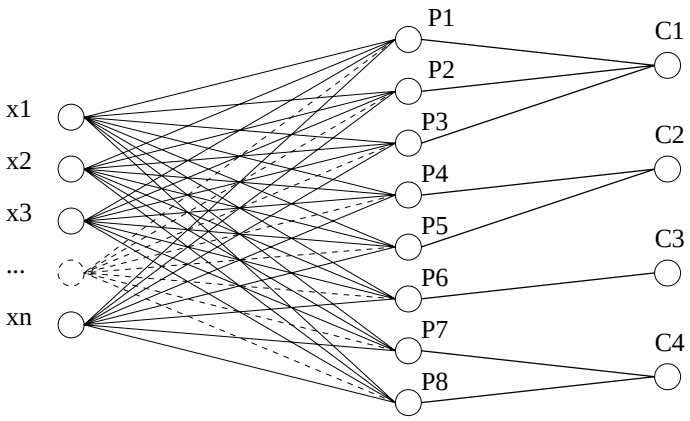

FIG. 1.1 – *Architecture du classifieur neuronal incrémental*

La couche d'entrée

La couche d'entrée est activée par un exemple à apprendre ou à reconnaître. L'exemple est modélisé par un vecteur d'entrée X qui le caractérise. Le nombre de cellules qui composent la couche d'entrée correspond à la dimension de l'espace d'entrée.

Exemple

Considérons, par exemple, un problème de reconnaissance de caractères dactylographiés discrétisées en $n \times n$ points. Le vecteur X peut être l'ensemble des n^2 points, en noir sur blanc (entrées binaires) ou en niveaux de gris (entrées réelles, $X \in \mathbb{R}^{n^2}$). Une autre possibilité est de considérer le nombre de points de l'image non blancs de chaque colonne ($X \in \mathbb{R}^n$). En fait, il « suffit » de trouver un ensemble de caractéristiques permettant de discriminer les classes à reconnaître.

La couche intermédiaire

La deuxième couche est composée de prototypes. Chaque cellule est totalement connectée avec la première couche, mais connectée à une seule cellule de la couche suivante. La taille de la deuxième couche varie durant l'apprentissage. Le réseau ajoute et connecte de nouvelles cellules s'il éprouve des difficultés à apprendre (nous préciserons cette stratégie par une définition algorithmique).

La couche de sortie

La couche de sortie stocke la « manière de classifier ». Une cellule de la troisième couche est connectée à un petit nombre de cellules de la deuxième couche. Durant un apprentissage supervisé, la dernière couche comprend autant de cellules qu'il existe de classes à apprendre ; en non supervisé, les cellules sont créées au fur et à mesure de l'émergence de nouvelles classes.

Réponse du classifieur

La réponse du classifieur est déterminée par le prototype le plus fortement activé, c'est-à-dire le prototype $P_{meilleur}$ le plus proche de l'entrée (au sens de la mesure choisie).

Schéma fonctionnel

La figure 1.2 schématise le classifieur incrémental par un module fonctionnel : le réseau reçoit des exemples accompagnés d'une étiquette (apprentissage supervisé) ou seuls (apprentissage non supervisé). Lors du processus de généralisation, le module reçoit des exemples et retourne leur classe (ou rejette l'exemple s'il n'est pas reconnu). Le module contient des ensembles disjoints de prototypes associés à des classes.

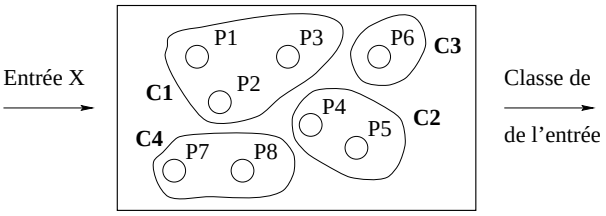

FIG. 1.2 – *Schéma fonctionnel du classifieur incrémental*

Recherche du prototype gagnant

Si l'on présente au classifieur un exemple à apprendre ou à reconnaître, l'influx se propage et les prototypes s'activent. L'apprentissage du classifieur suit un algorithme par compétition.

- L'algorithme cherche le prototype $P_{meilleur}$ le plus proche de l'entrée (au sens de la mesure choisie pour le problème traité).
- Le classifieur cherche alors le second meilleur prototype (P_{second}) appartenant à une autre classe que $P_{meilleur}$.
- L'algorithme s'assure que $P_{meilleur}$ est suffisamment proche de la forme en entrée, et suffisamment éloigné de P_{second}. Si ces deux conditions sont vérifiées, le prototype $P_{meilleur}$ est élu prototype gagnant et noté $P_{gagnant}$.

Algorithme d'apprentissage supervisé

Durant un apprentissage supervisé, les exemples à apprendre sont présentés accompagnés d'une étiquette indiquant leur classe.
Si la classe du prototype gagnant correspond à l'étiquette de l'exemple en entrée, le prototype est rapproché de l'exemple. Dans l'éventualité d'une absence de gagnant, ou si le gagnant n'est pas de la classe de l'exemple à apprendre, le classifieur crée un nouveau prototype à partir de l'exemple en entrée (les w_i sont initialisés par les valeurs des x_i). Le nouveau prototype est ajouté à l'ensemble des prototypes chargés de reconnaître la classe de l'exemple à apprendre.

Algorithme d'apprentissage non supervisé

Peu de modèles de réseaux de neurones artificiels sont capables d'apprendre de façon non supervisée. Le classifieur incrémental a cette capacité.
Les exemples sont présentés sans étiquette. De même que pour un apprentissage supervisé, le classifieur recherche un prototype gagnant. Le prototype trouvé est rapproché de l'exemple en entrée. En l'absence d'un gagnant, le classifieur crée un nouveau prototype dans une nouvelle classe. Les classes qui émergent de ce processus peuvent être identifiées *a posteriori* par un expert humain.

Algorithme « winner takes all »

Notons que l'algorithme d'apprentissage du classifieur ne modifie que le prototype gagnant. Ce type d'algorithme est appelé « algorithme *winner takes all* ». Ce comportement diverge des algorithmes classiques, apprenant par compétition, présentés en annexe (section A.6) :

- dans les cartes de Kohonen, les voisins de $P_{gagnant}$ sont également approchés de la forme en entrée, de plus quelques prototypes sont éventuellement repoussés (selon l'étendue du voisinage).
- pour la méthode LVQ, les prototypes gagnant par erreur (mauvaise classe) sont repoussés de la forme en entrée ;

1.1.3 Variante de l'algorithme incrémental

Il existe plusieurs variantes de l'algorithme du classifieur que nous venons d'étudier.

- *Mécanisme d'élagage des prototypes inutiles*

  Un mécanisme d'élagage permet de réduire de nombre de prototypes. Contrairement au modèle GAL, la suppression des prototypes surnuméraires n'impose pas une longue phase de sommeil. Durant l'apprentissage, chaque prototype évalue son utilité en comptant ses victoires ($P_{gagnant}$). Les prototypes gagnant le moins souvent finissent par se retirer d'eux-mêmes du classifieur. Ce mécanisme d'élagage est efficace mais potentiellement dangereux car un représentant atypique d'une classe risque de disparaître.

  Exemple

  À titre d'exemple, on peut écrire le chiffre « 7 » avec ou sans barre horizontale. Une base d'apprentissage européenne comprendra probablement une grande majorité de « 7 barrés ». Le processus d'élagage risque de supprimer le prototype « 7 non barré » du classifieur alors que ce représentant est indispensable à une bonne reconnaissance de la classe des « 7 ».

- *Ajustement de la zone d'influence*

  Une autre variante du classifieur modifie la zone d'influence des prototypes à la manière du modèle NLS [AG91, Mal96]. Lorsque le prototype gagnant n'est pas de la bonne classe, au lieu de créer un nouveau prototype pour représenter l'exemple à apprendre, le classifieur réduit la zone d'influence du prototype gagnant jusqu'à exclure l'exemple à apprendre. Comme pour le réseau NLS, la base d'exemples doit être présentée à nouveau pour s'assurer que tous les exemples sont bien classifiés.

Parallélisation des variantes

Les parallélisations présentées dans cette thèse se basent sur l'algorithme originel mis en œuvre par Azcarraga dans sa thèse, sur une tâche de reconnaissance de formes que nous détaillons dans la prochaine section. Cependant, les deux variantes décrites ci-dessus peuvent facilement être mises en œuvre dans chacune des parallélisations que nous allons étudier.

1.2 Évaluation du classifieur en reconnaissance de formes

Cette section traite de l'utilisation du classifieur incrémental pour reconnaître des dessins au trait, et plus particulièrement des chiffres manuscrits. Le connexionnisme s'est souvent avéré être la méthode la plus efficace pour ce type de problème dit « d'OCR » (*Optical Character Recognition*).

1.2.1 Un prétraitement adapté aux données

La figure 1.3 présente des exemples (parmi les plus lisibles) de chaque classe des chiffres que nous allons apprendre au classifieur.

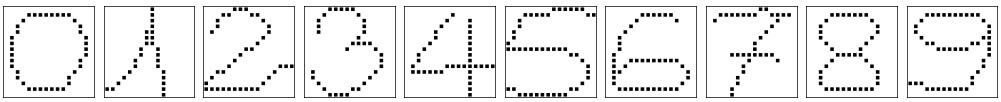

FIG. 1.3 – *Exemples de chiffres manuscrits à apprendre et à reconnaître*

Le classifieur incrémental est indépendant du type de formes à reconnaître, cependant l'utilisation directe des matrices de pixels comme exemples à apprendre donnerait de mauvais résultats. En effet, un prétraitement adapté au domaine d'application facilite fortement la tâche du classifieur.

« *COLD* »

Dans le cadre de sa thèse, Azcarraga [Azc93] présente un module de prétraitement neuronal efficace nommé COLD (*Configuration of Oriented Line Detectors*) dédié à la reconnaissance de dessins au trait [AA92]. Le schéma 1.4 présente le couplage du module de prétraitement et du classifieur.

Détail du prétraitement

Le prétraitement sera détaillé à l'occasion d'une des parallélisations (chapitre 3). L'entrée du classifieur est le résultat de l'activation de cellules, dites « à champ récepteur [Har40] », chargées de détecter la position et l'orientation de petits segments dans l'image. Nous verrons notamment que ce prétraitement offre une bonne tolérance : (i) à un bruit aléatoirement distribué dans l'image ; (ii) aux petites variations en translation et aux petites déformations.

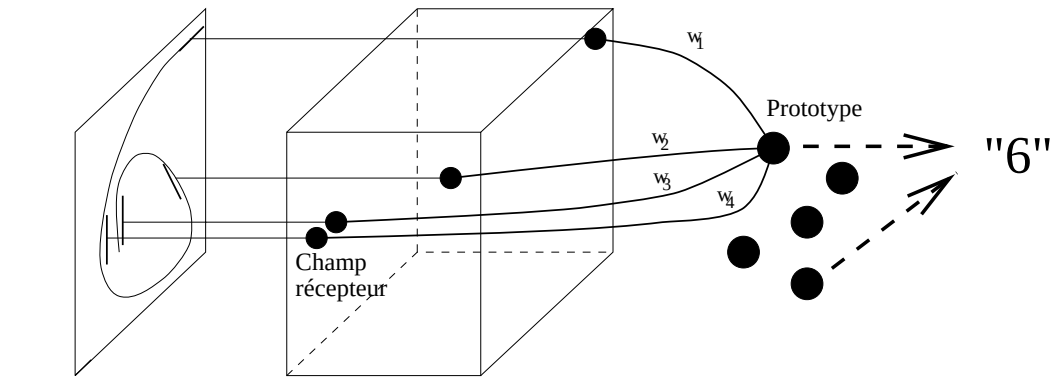

FIG. 1.4 – *Couplage du module de prétraitement et du classifieur incrémental*

1.2.2 Algorithme d'apprentissage supervisé adapté à l'OCR

Mesure de similarité

Il est important d'utiliser une mesure adaptée au problème pour l'activation d'un prototype P_i du classifieur. C'est au sens de cette mesure de « similarité » qu'un des prototypes est élu gagnant. Une mesure efficace pour le problème de la reconnaissance de dessins au trait [Azc93] est donné par l'équation 1.1. Les entrées x_j sont les sorties des n cellules à champ récepteur activées, connectées au prototype P_i. Les w_{ij} sont les poids valuant les connexions : w_{ij} est la composante j du prototype P_i. Le numérateur est l'intersection du prototype considéré et de la forme en entrée. Le dénominateur normalise les résultats en prenant en compte la taille de l'entrée.

$$s(X, P_i) = \frac{2 \sum_{j=1}^{n} w_{ij} x_j}{\left(\sum_{j=1}^{n} w_{ij}^2\right) + \left(\sum_{j=1}^{n} x_j^2\right)} \tag{1.1}$$

Le processus d'apprentissage supervisé peut être présenté sous la forme d'un algorithme. Dans cette section, $C(P_i)$ représente la classe du prototype P_i. L'apprentissage supervisé se décompose en 4 étapes. Chaque exemple est présenté une seule fois. Les poids des cellules de la deuxième couche (c'est-à-dire les prototypes) sont modifiés à la dernière étape de l'algorithme.

1. Présenter un exemple à apprendre X, avec son étiquette E.
2. Rechercher le meilleur prototype $P_{meilleur}$, avec la plus grande similarité $s(X, P_i)$:

$$s(X, P_{meilleur}) = \max_i s(X, P_i) = \max_i \frac{2 \sum_{j=1}^{n} w_{ij} x_j}{\left(\sum_{j=1}^{n} w_{ij}^2\right) + \left(\sum_{j=1}^{n} x_j^2\right)} \tag{1.2}$$

3. Rechercher le second meilleur prototype P_{second}, d'une autre classe que $P_{meilleur}$:

$$s(X, P_{second}) = \max_{\substack{i \\ C(P_i) \neq C(P_{meilleur})}} s(X, P_i) \tag{1.3}$$

4. **Si** $P_{meilleur}$ est suffisamment proche de X, suffisamment éloigné de P_{second}, et si la classe de $P_{meilleur}$ correspond à l'étiquette de X :

$$\begin{cases} s(X, P_{meilleur}) \Leftrightarrow s(X, P_{second}) > \Theta_{confusion} \\ s(X, P_{meilleur}) > \Theta_{influence} \\ C(P_{meilleur}) = E \end{cases} \tag{1.4}$$

Alors modifier $P_{meilleur}$ pour l'approcher de X :

$$Pour\ i\ \ tel\ que\ \ P_i = P_{meilleur} \qquad \forall j\ \ w_{ji} \leftarrow w_{ji} + \alpha(x_j \Leftrightarrow w_{ji}) \tag{1.5}$$

Sinon X devient un nouveau prototype connecté à la classe E :

$$\begin{cases} P_{nouveau} \leftarrow X \\ C(P_{nouveau}) \leftarrow E \end{cases} \tag{1.6}$$

Initialisation de l'apprentissage

Il découle de cet algorithme que le premier exemple présenté au classifieur, ainsi que le premier exemple présenté pour chaque classe, deviennent directement des prototypes. En conséquence, et contrairement à beaucoup d'algorithmes connexionnistes, aucune initialisation particulière des poids du réseau n'est nécessaire.

1.2.3 Performances du classifieur

Les performances du classifieur dépendent de la pertinence des prototypes créés lors de l'apprentissage. La qualité des prototypes est testée en phase de généralisation, en présentant 500 nouveaux exemples. L'algorithme de généralisation suit le même principe que l'algorithme d'apprentissage mais les prototypes ne sont pas modifiés :

1. Présenter un exemple X, avec son étiquette E.
2. Rechercher le meilleur prototype $P_{meilleur}$, avec la plus grande similarité $s(X, P_i)$.
3. Rechercher le second meilleur prototype P_{second}, d'une autre classe que $P_{meilleur}$.
4. **Si** $P_{meilleur}$ est trop éloigné de X ou trop proche de P_{second} :

$$\begin{cases} s(X, P_{meilleur}) \Leftrightarrow s(X, P_{second}) \leq \Theta_{confusion} \\ s(X, P_{meilleur}) \leq \Theta_{influence} \end{cases} \tag{1.7}$$

Alors rejeter X (pas de réponse) ;

Sinon Si la classe de $P_{meilleur}$ correspond à l'étiquette de X :

$$C(P_{meilleur}) = E \tag{1.8}$$

Alors l'exemple X est reconnu (bonne réponse) ;

Sinon l'exemple X n'est pas reconnu (mauvaise réponse).

Qualité de l'apprentissage

La qualité de l'apprentissage séquentiel se mesure par le pourcentage de formes reconnues en généralisation. La table 1.1 donne le nombre de prototypes créés en apprentissage, ainsi que le meilleur pourcentage de chiffres reconnus en généralisation, une fois les meilleurs paramètres $\Theta_{confusion}$ et $\Theta_{influence}$ trouvés.

Nombre de prototypes créés	27
Chiffres correctement reconnus	99 %

TAB. 1.1 – *Qualité de l'apprentissage séquentiel sur la base de chiffres*

Rapidité d'exécution de l'apprentissage séquentiel

La table 1.2 résume les temps d'exécution mesurés. Le programme est testé sur un des processeurs de la machine parallèle Volvox (processeur *intel i*860). Ce choix de plate-forme de test sera motivé dans le prochain chapitre. Deux ensembles disjoints, chacun de 500 chiffres manuscrits, sont utilisés pour l'apprentissage supervisé et la généralisation.

Temps d'apprentissage (en secondes)	109.38
Temps de généralisation (en secondes)	102.84

TAB. 1.2 – *Temps d'exécution de l'apprentissage séquentiel sur la base de chiffres*

Comparaison des temps de prétraitement et de classification

La figure 1.5 suit l'évolution du temps de prétraitement (courbe supérieure) et du temps de classification (courbe inférieure) au fur et à mesure de l'apprentissage des 500 exemples. Le temps de classification comprend la recherche d'un éventuel prototype gagnant ($P_{gagnant}$), ainsi que les temps de création de nouveaux prototypes, ou de modification des prototypes existants. Le classifieur étant incrémental, le temps de recherche de $P_{gagnant}$ augmente avec le nombre de prototypes.

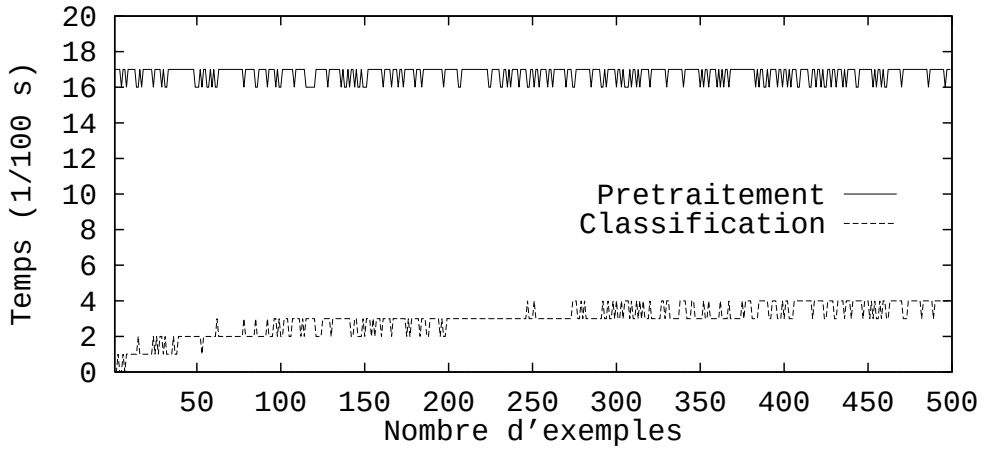

FIG. 1.5 – *Évolution des temps de traitement en apprentissage*

La table 1.3 reprend les temps de prétraitement et de classification, en fin d'apprentissage. On constate que le classifieur passe quatre fois plus de temps à prétraiter les exemples qu'à les apprendre.

Prétraitement (10^{-2} s)	16.5
Classification du dernier exemple (10^{-2} s)	4.0

TAB. 1.3 – *Temps de prétraitement et de classification en fin d'apprentissage*

1.2.4 Conclusion sur le problème d'OCR

Le classifieur neuronal incrémental est très efficace sur le problème de la reconnaissance de dessins au trait. De plus amples informations sur le fonctionnement du classifieur appliqué au problème d'OCR sont disponibles dans la thèse d'Azcarraga [Azc93]. Au-delà des taux d'erreurs, et pour revenir à notre objectif de parallélisation, il apparaît clairement que le temps de prétraitement n'est pas négligeable. Aussi, toute méthode de parallélisation efficace du classifieur, dans le cadre de cette application, se doit de paralléliser également le prétraitement.

Le classifieur a été conçu pour résoudre un problème d'OCR. Dans la suite de ce chapitre, le modèle est testé sur une autre tâche de classification afin de valider sa généricité.

1.3 Évaluation du classifieur sur les « formes d'ondes »

1.3.1 Construction des formes d'ondes

Cette section présente une évaluation du classifieur incrémental neuronal sur un problème de classification introduit par Breiman, Friedman, Olshen, et Stone en 1984 pour l'étude des arbres de décision [BFOS84]. L'objectif est de discriminer parmi trois classes de formes d'ondes notées h_1, h_2, et h_3 (figure 1.6). Chaque onde simule un phénomène chronologique étudié à 21 instants ($X \in \mathbb{R}^{21}$). Les classes sont générées par une combinaison de trois ondes de bases données par les équations 1.9, où U est une variable aléatoire de densité uniforme sur l'intervalle $[0, 1]$ et ϵ est un vecteur aléatoire gaussien (bruit). Les associations sont réalisées aléatoirement et altérées par le bruit gaussien.

$$\begin{cases} X = Uh_1 + (1 \Leftrightarrow U)h_2 + \epsilon & \Leftrightarrow \quad X \in Classe\ 1 \\ X = Uh_1 + (1 \Leftrightarrow U)h_3 + \epsilon & \Leftrightarrow \quad X \in Classe\ 2 \\ X = Uh_2 + (1 \Leftrightarrow U)h_3 + \epsilon & \Leftrightarrow \quad X \in Classe\ 3 \end{cases} \tag{1.9}$$

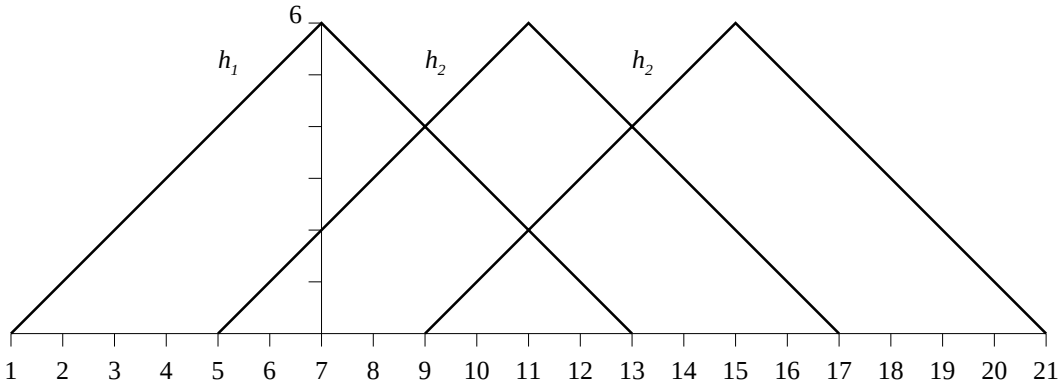

FIG. 1.6 – *Trois formes d'ondes h_1, h_2, et h_3, dont la combinaison engendre les trois classes du problème introduit par Breiman et al.*

1.3.2 Algorithmique du classifieur pour les formes d'ondes

Nous reprenons l'algorithme d'apprentissage supervisé donné dans la section 1.2.2 en l'adaptant à la classification des formes d'ondes. Il faut notamment changer la mesure utilisée pour l'activation des prototypes, ce qui change le sens des inéquations du test principal (équation 1.13). Aucun prétraitement spécifique n'étant nécessaire, prenons simplement la distance euclidienne. On note $d(X, P_i)$ la distance euclidienne entre l'exemple $X = (x_1 x_2 \ldots x_n)$ et le prototype $P_i = (w_{1i} w_{2i} \ldots w_{ni})$:

$$d(X, P_i) = \sqrt{\sum_{j=1}^{n} (x_j \Leftrightarrow w_{ji})^2} \tag{1.10}$$

Algorithme

1. Présenter un exemple à apprendre X, avec son étiquette E.

2. Rechercher le meilleur prototype $P_{meilleur}$, avec la plus faible distance $d(X, P_i)$:

$$d(X, P_{meilleur}) = \min_i d(X, P_i) = \min_i \sqrt{\sum_{j=1}^{n} (x_j \Leftrightarrow w_{ji})^2} \tag{1.11}$$

3. Rechercher le second meilleur prototype P_{second}, d'une autre classe que $P_{meilleur}$:

$$d(X, P_{second}) = \min_{\substack{i \\ C(P_i) \neq C(P_{meilleur})}} d(X, P_i) \tag{1.12}$$

4. Si $P_{meilleur}$ est suffisamment proche de X, suffisamment éloigné de P_{second}, et si la classe de $P_{meilleur}$ correspond à l'étiquette de X :

$$\begin{cases} d(X, P_{second}) \Leftrightarrow d(X, P_{meilleur}) > \Theta_{confusion} \\ d(X, P_{meilleur}) < \Theta_{influence} \\ C(P_{meilleur}) = E \end{cases} \tag{1.13}$$

Alors modifier $P_{meilleur}$ pour l'approcher de X :

$$Pour\ i\ \ tel\ que\ \ P_i = P_{meilleur} \qquad \forall j\ \ w_{ji} \leftarrow w_{ji} + \alpha(x_j \Leftrightarrow w_{ji}) \quad (1.14)$$

Sinon X devient un nouveau prototype connecté à la classe E :

$$\begin{cases} P_{nouveau} \leftarrow X \\ C(P_{nouveau}) \leftarrow E \end{cases} \quad (1.15)$$

1.3.3 Performances du classifieur

Base de Breiman

Une approche bayesienne conduit à une erreur minimale théorique en généralisation d'environ 14 % avec un ensemble d'apprentissage infini (voir [BFOS84]). La table 1.4 donne les performances du classifieur en généralisation, après un apprentissage sur les bases de Breiman *et al.*

Proto.	Succès	Erreur	(%)	Rejet
65	3865	1062	21	73

TAB. 1.4 – *Résultats du classifieur sur la base de Breiman*

Les ensembles d'apprentissage et de généralisation ne sont pas parfaitement équilibrés, la table 1.5 précise l'effectif de chaque classe. Les 300 exemples appris engendrent 65 prototypes. La généralisation sur 5000 exemples donne 21 % d'erreur.

	Apprentissage	Généralisation
Classe 1	103	1678
Classe 2	108	1643
Classe 3	89	1679
Total	300	5000

TAB. 1.5 – *Effectif de chacune des trois classes du problème de Breiman et al.*

Bases inter-PRC

Un grand nombre de méthodes de classification ont été mises en œuvre sur le problème de Breiman dans le cadre du projet inter-PRC [GG] « Méthodes Symbolique-Numériques de Discrimination ».
Ce projet a impliqué les laboratoires INRIA (Rocquencourt), IRISA (Rennes), LAFORIA (Paris), LIASC (Brest), LIM (Marseille), LIRMM (Montpellier), et LRI (Orsay). Le projet était animé par Gascuel (LIRMM) et Gallinari (LAFORIA). Nous allons comparer les résultats obtenus afin de valider le classifieur incrémental.

Protocole expérimental

Afin de profiter du travail déjà accompli, nous avons testé le classifieur sur les bases du projet inter-PRC et sur la base de Breiman, en respectant scrupuleusement les mêmes conditions d'expérimentation.

Les bases 1 à 10 présentées dans la table 1.6 sont issues du projet inter-PRC. L'ajout de la base de Breiman (numéro 0 dans la table) permet de calculer une réponse moyenne sur 11 bases. La lecture de ce tableau révèle une bonne stabilité du classifieur incrémental. En effet, l'écart-type sur les 11 expériences n'est que de 5 prototypes et de 40 erreurs de classification (sur les 5000 exemples).

Base	Proto.	Succès	Erreur	(%)	Rejet
0	65	3865	1062	21	73
1	76	3896	1049	21	55
2	72	3916	1059	21	25
3	72	3826	1106	22	68
4	70	3806	1141	23	53
5	76	3875	1069	21	56
6	79	3939	1009	20	52
7	67	3908	1030	21	62
8	71	3832	1123	22	45
9	61	3836	1099	22	65
10	74	3854	1065	21	81
Moyenne	71	3868	1074	21,5	58
Écart-type	5	42	40		15

TAB. 1.6 – *Résultats du classifieur sur les bases inter-PRC et sur la base de Breiman*

Matrice de confusion

Une matrice de confusion résume parfaitement les performances en généralisation, et ainsi la qualité de l'apprentissage. L'élément situé à la ligne i et à la colonne j est le nombre d'exemples de la classe i classifiés comme appartenant à la classe j.

- La matrice (1.16) est calculée après un apprentissage de la base de Breiman.
- La matrice de confusion (1.17) est calculée sur la moyenne des 11 apprentissages (c'est-à-dire avec chacune des bases).

Aucun déséquilibre n'est visible en fonction des trois classes.

$$\mathcal{C}_{conf\_0} = \begin{pmatrix} 1160 & 333 & 148 \\ 72 & 1384 & 179 \\ 124 & 132 & 1416 \end{pmatrix} = \begin{pmatrix} 23,20\,\% & 6,66\,\% & 2,96\,\% \\ 1,44\,\% & 27,68\,\% & 3,58\,\% \\ 2,48\,\% & 2,64\,\% & 28,32\,\% \end{pmatrix} \quad (1.16)$$

$$\mathcal{C}_{conf\_moy} = \begin{pmatrix} 1160 & 253 & 234 \\ 160 & 1330 & 141 \\ 157 & 129 & 1379 \end{pmatrix} = \begin{pmatrix} 23,20\,\% & 5,06\,\% & 4,58\,\% \\ 3,20\,\% & 26,60\,\% & 2,82\,\% \\ 3,14\,\% & 2,58\,\% & 27,58\,\% \end{pmatrix} \quad (1.17)$$

La matrice (1.18) donne l'écart-type des 11 matrices de confusion et permet de vérifier la forte stabilité de la méthode d'une exécution à une autre.

$$\mathcal{C}_{écart-type} = \begin{pmatrix} \mathbf{4} & 3 & 3 \\ 4 & \mathbf{5} & 3 \\ 3 & 3 & \mathbf{5} \end{pmatrix} \tag{1.18}$$

Taux d'erreurs du classifieur en généralisation

Les taux d'erreurs en généralisation sont bons en comparaison des méthodes testées par le projet inter-PRC. La table 1.7 regroupe les résultats présentés dans le rapport d'activité du projet.

Méthode mise en œuvre	Erreur	Classe 1	Classe 2	Classe 3
Classifieur incrémental	21,5 %	29,0 %	18,3 %	17,0 %
Discrimination linéaire	20,4 %	23,5 %	19,8 %	17,9 %
Discrimination quadratique	21,4 %	21,5 %	21,7 %	21,0 %
Noyaux de Parzen	24,0 %	24,9 %	23,2 %	24,0 %
Les k plus proches voisins	27,0 %	31,6 %	22,9 %	26,5 %
Réseau de neurones (gradient)	17,1 %	21,8 %	14,2 %	15,4 %
Arbres de décision flous	28,0 %	29,8 %	27,2 %	27,0 %

TAB. 1.7 – *Comparaison du classifieur avec les modèles du projet inter-PRC*

Méthodes statistiques La table présente deux méthodes statistiques [Fis36] non-paramétriques (discriminations linéaire et quadratique), et deux méthodes statistiques paramétriques (noyaux de Parzen [Par62] et les k plus proches voisins [CH67]).

Méthode connexionniste Le modèle connexionniste présenté dans la table est un perceptron multicouche. L'apprentissage est conduit par l'algorithme de rétro-propagation du gradient.

Méthode des sous-ensembles flous La dernière méthode présentée dans la table 1.7 est un classement par arbres de décision flous [Mar94], utilisant une mesure d'entropie de Shannon.

1.3.4 Conclusion sur le problème des formes d'ondes

Le classifieur neuronal incrémental affiche un bon taux d'erreurs en reconnaissance (21,5 %). Le seul résultat significativement meilleur est obtenu par un autre modèle connexionniste.

Inconvénients d'un MLP

Cependant, la supériorité du couple perceptron multicouche - algorithme de rétro-propagation du gradient est contrebalancée par la nécessité de rechercher la meilleure

architecture (nombre de couches et de cellules par couche), et par une bonne initialisation des poids. En effet, ces deux contraintes sont implicitement levées par l'apprentissage du classifieur incrémental.

Conclusion

Résumé du chapitre

Cette thèse s'intéresse à un modèle particulier de réseau de neurones artificiels : un « classifieur neuronal incrémental ». Ce modèle est capable de discriminer un ensemble de vecteurs en classes. Le réseau étant incrémental, des exemples caractéristiques nommés « prototypes » sont créés et ajustés durant l'apprentissage. L'algorithme mis en œuvre pour apprendre les exemples est dit « winner takes all », une compétition entre les prototypes permet d'élire un « gagnant » qui seul sera modifié. Si aucun gagnant viable n'est trouvé parmi les prototypes existants, l'exemple à apprendre devient lui-même un nouveau prototype. Contrairement à la plupart des modèles connexionnistes, le nombre de couches, le nombre de cellules par couches, et l'interconnexion des cellules, sont complètement déterminés par le problème à résoudre. Il n'est donc pas nécessaire de chercher empiriquement la topologie du réseau.

Problème d'OCR

Le classifieur incrémental neuronal a été mis au point sur un problème de reconnaissance de formes manuscrites. Les taux d'erreur sont excellents, notamment grâce à un prétraitement très efficace mais lourd en calculs. Cette constatation met en lumière l'importance de paralléliser aussi bien le prétraitement que la classification en elle-même.

Benchmark de Breiman et al.

Le modèle a été testé sur une nouvelle tâche afin de garantir la pérennité des parallélisations que nous allons étudier. Le problème choisi pour évaluer le classifieur, la classification de formes d'ondes, a été particulièrement étudié dans la littérature. Le respect pointu du protocole mis en œuvre avec d'autres méthodes permet de valider la qualité du classifieur incrémental.

Nous avons vu que si le classifieur du LIFIA se comporte bien, d'autres modèles obtiennent des résultats légèrement meilleurs. Cependant, ces différences ne sont pas très significatives. Le classifieur incrémental a de nombreuses qualités qui justifient amplement que l'on s'attache à le paralléliser.

L'informatique parallèle est un domaine de recherche aussi vaste que le connexionnisme. Aussi, avant de présenter les parallélisations du classifieur, il est indispensable d'introduire quelques notions et définitions de base. Cette introduction au parallélisme est l'objet du prochain chapitre.