REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
Ministre de l'enseignement supérieur et de la recherche scientifique

Université Mouloud Mammeri de Tizi-Ouzou
Faculté de Génie Electrique et d'informatique
Département d'Automatique.

Mémoire de fin d'étude
De MASTER ACCADEMIQUE
Spécialité : Commande des systèmes

Présenté par
Naima IFREK
Lahna BOUSSAID
Mémoire dirigé par Farida DORBANE

Thème

Etude et application du réseau ELM (Extreme Learning Machine) pour la classification de donnes

Mémoire soutenu publiquement le 12/07/2017 devant le jury composé de

M^me^ ALKAMA Sadia
MCA UMMTO Président

M^me^ DORBANE Farida
MAA UMMTO Encadreur

M^me^ AIT AIDER Malika
MAA UMMTO Examinateur

M^me^ CHILALI Ouardai
MAA UMMTO Examinateur

Remerciements

Nous tenons à remercier :

- Le bon Dieu qui nous a donné santé et la persévérance durant notre cursus.
- Nos parent est tous ceux qui nous ont aidé pour mener à terminer ce travail.
- Notre promotrice F. DORBANE
- Les membres de jury qui fera l'honneur d'examiner notre travail.
- Toutes les personnes qui ont participé aux enregistrements
- Et tous les lecteurs

Dédicaces

Avec les sentiments d'amour et de gratitude le plus sincères
Je dédie ce modeste ce travail.

- A ma très chère mère GHNIMA pour son amour, ses prières et ses sacrifices.
- A mon très cher père MESSAOUD qui a veille, tout au long de ma vie, à ce que je n'eusse besoin de rien

- A mes frères Rafik, Djaffar, Karim et ma très chère sœur BACERA que dieu les protège.
- A mon très cher ami Hocine
- A mes amis du cœur : Hocine, Kahina, Fatah, Lila Moh Ameziane et Ghiles.

- A mes enseignants qui ont contribues à ma formation depuis mon jeune âge

- A mon très cher binôme et sa famille

Lahna

Dédicaces

Je dédié ce modeste travail à :

- A ma très chère mère source de tendresse
- A mon très cher père, qui m'encourage dans les instants délicats
- Mon frère Samir
- Mes sœurs : Kahina, Lynda et Nassima
- A ma petite nièce Silyna
- A mon très chère ami Mohamad
- A la famille de mon binôme Lahna
- Tous mes amis qui ont partagé mes bons souvenirs

Naima

Sommaire

CHAPITRE II : Description et Fonctionnement du réseau ELM (*Extreme Learning Machine*)

CHAPITRE III : Evaluation des performances du réseau ELM en classification

Introduction générale

Plusieurs chercheurs tentent de mieux comprendre le fonctionnement du cerveau et s'intéressent aux interactions entre les neurones biologiques. Pour y arriver certains d'entre eux utilisent des réseaux de neurones artificiels dans le mécanisme de déclenchement des influx nerveux qui respecte la réalité des neurones biologiques.

Le réseau de neurones artificiels inspiré du système nerveux biologique, est un automate caractérisé par un nombre de fonctions mathématiques. Il traite un signal recueilli (signal d'entrée) à travers ses connexions entrantes pour fournir un signal en sortie, calculé par la fonction de transfert du réseau. En général, ils considèrent que chaque neurone (unité de calcul) fournit une information additive aux neurones qui lui sont connectés.

Dans la littérature, il existe plusieurs types de réseaux de neurones. En particulier, nous nous sommes intéressés aux réseaux de neurones multicouches de type *feed-forward*, qui sont les plus utilisés pour la classification de données. Ces réseaux de neurones multicouches fonctionnent comme des approximateurs de fonctions. Les algorithmes d'apprentissage conventionnels, pour la modélisation des réseaux de neurones, doivent toujours ajuster les paramètres du réseau, en particulier ceux de la couche cachée. Ce qui nécessite un temps d'apprentissage relativement long par rapport à la taille et la dimension des données utilisées. En plus, le choix de l'architecture du réseau est d'une grande importance pour la résolution du problème d'approximation.

Contrairement à ces considérations, un nouveau réseau a vu le jour, récemment, qui ne se soucie pas beaucoup de taille du réseau. En plus, d'un point de vue d'approximation de fonctions, les paramètres de la couche cachée sont générés de manière aléatoire et restent constants au lieu d'être ajustés. Seuls les poids de sortie sont calculés par une simple régression linéaire. Ce réseau, connu sous l'appellation réseau de neurones ELM (*Extrême Learning Machine*), a été proposée par **Guang-Bin Huang** en 2006[1]. Parmi les avantages de ce nouveau réseau nous citons : la vitesse d'apprentissage qui est estimée à 100 fois plus rapide et une bonne capacité de généralisation.

L'objectif de notre travail est d'étudier le réseau ELM, ainsi confirmer ces performances en classification de données. Pour cela, nous avons réparti notre travail comme suit :

Dans le chapitre I, nous présentons les éléments de base des réseaux de neurones artificiels, ainsi les différents types de réseau de neurones. A la fin nous exposons les avantages et inconvénients des réseaux de neurones artificiels.

Le chapitre II se porte sur la description du réseau ELM : architecture, fonctionnement et avantages.

Dans le chapitre III, nous nous sommes intéressés à l'évaluation des résultats d'application du réseau ELM en classification de données. Pour le tester, nous avons utilisés différentes bases de données(en terme de taille, nombre de classes, nombre de caractéristiques,…), nous avons également étudié l'influence des paramètres du ce réseau sur les résultats de la classification.

Enfin nous terminons notre travail par une conclusion générale.

Chapitre I

Les réseaux de neurones

I.1 Introduction

Dans ce chapitre, nous allons définir c'est quoi un réseau de neurone artificiel par analogie avec les réseaux de neurones biologiques, nous introduisons, aussi, la modélisation mathématique des réseaux de neurones artificiels. Par la suite, nous verrons, le concept d'apprentissage qui est une faculté très importante pour les réseaux de neurones, d'où leur utilisation dans des domaines très variés.

I.2 Historique

De nombreux ouvrages ont permis de documenter l'histoire des recherches en réseaux de neurones. Les recherches menées dans le domaine du connexionnisme ont démarré avec la présentation en **1943** par **W. McCulloch** et **W. Pitts [1]** d'un modèle simplifié de neurone biologique communément appelé neurone formel (Artificiels). Ils ont montré théoriquement que les réseaux de neurones formels simples peuvent réaliser des fonctions logiques, arithmétiques et symboliques complexes.

En **1958**, **F. Rosenblatt [1]** développe le modèle du Perceptron. Ce dernier possède deux couches de neurones : une couche de perception (sert à recueillir les entrées) et une couche de décision. C'est le premier modèle pour lequel un processus d'apprentissage a pu être défini.

S'inspirant du perceptron, **Widrow et Hoff [1]**, développent, dans la même période, le modèle de l'Adaline (*Adaptive Linear Element*). Ce dernier sera, par la suite, le modèle de base des réseaux de neurones multicouches.

Les recherches sur les réseaux de neurones ont été pratiquement abandonnées lorsque **M. Minsky** et **S. Papert** [1] ont publié leur livre « Perceptrons » en (**1969**) et démontré les limites théoriques du perceptron, en particulier, l'impossibilité de traiter les problèmes non linéaires par ce modèle.

Une révolution survient alors dans le domaine des réseaux de neurones artificiels, une nouvelle génération de réseaux de neurones capable de traiter avec succès des phénomènes non-linéaires : le perceptron multicouche, il ne possède pas les défauts mis en évidence par **Minsky**. Proposé pour la première fois par **Werbos [1]**, le perceptron Multicouche apparait en **1986** introduit par **Rumelhart**, et simultanément sous une appellation voisine, chez le Cun(1985). Ces systèmes reposent sur la rétro-propagation du gradient de l'erreur dans des

systèmes a plusieurs couches, chacune de types **Adaline** de **Bernard Widrow**, proche du perceptron de **Rumelhart.** L'évolution de la théorie des réseaux de neurones formels est inspirée directement du développement des travaux biologique sur le cerveau humain.

I.3 Le neurone biologique

Le cerveau est l'organe de commande le plus complexe et le plus inconnu de la biologie de l'homme ou de l'animal. Les cellules nerveuses, appelées neurones, sont les éléments de base du système nerveux central.

Dans leur organisation générale et leur système biochimique, ils présentent, cependant, des caractéristiques particulières qui se distinguent par quatre fonctions spécialisées :

- Recevoir des signaux en provenance des neurones voisins ;
- Intégrer ces signaux ;
- Engendrer un influx nerveux ;
- Conduire et transmettre l'influx nerveux à un neurone capable de le recevoir ;

Le neurone biologique possède de quatre parties principales (voir figure I.1) :

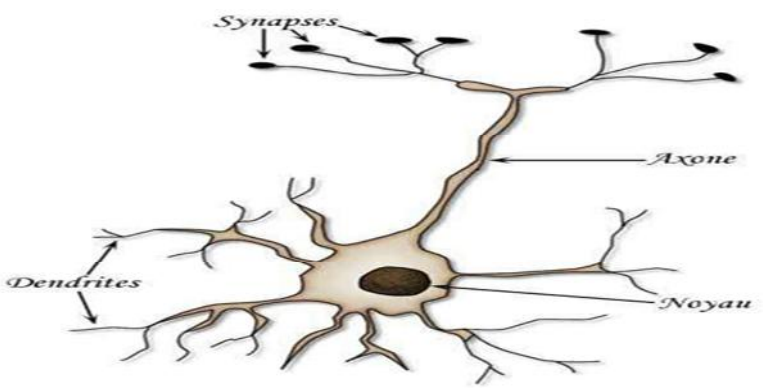


Figure I.1 : neurone biologique.

- **Axone :** l'axone transporte l'influx nerveux vers les synapses.
- **Synapses** : transmettent l'influx nerveux provenant de l'axone vers d'autres cellules à partir neurotransmetteurs.
- **Dendrites** : selon leur longueur et leur perméabilité, affectent la quantité d'influx nerveux qui se rend au noyau.
- **Noyau :** est le centre des réactions électrochimiques, si les stimulations externes sont suffisantes, le noyau provoque l'envoi d'un influx nerveux électrique à travers l'axone.

I.4 Neurone formel (Artificiel)

Inspiré du système biologique, le neurone formel est un automate caractérisé par un petit nombre de fonctions mathématiques. Il traite un signal recueilli (signal d'entrée) à travers ses connexions entrantes pour fournir un signal de sortie calculé par la fonction de transfert. En générale, on considère que chaque neurone fournit une information additive aux unités de calcul (neurones) qui lui sont connectées [1].

La valeur de la somme pondérée $\boldsymbol{n}$ est, simplement, la somme des sorties des différents neurones en amont avec lesquels il est connecté, La figure 2 résume la structure détaillée du neurone formel (Artificiel).

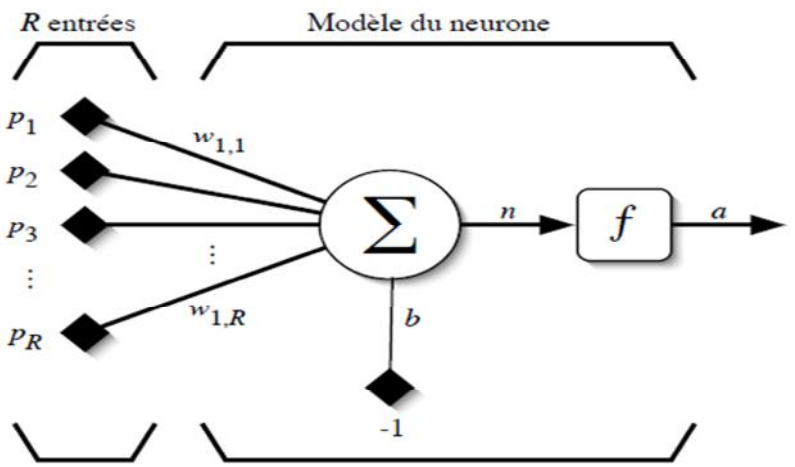


Figure I.2 : Modèle générale du neurone formel.

$P = [p_1, p_2, \cdots, p_R]^T$ R entrées des neurones correspondent au vecteur de caractéristiques du signal d'entrée.

$w = [w_{1,1}, w_{1,2}, \cdots, w_{1,R}]^T$ Vecteur des poids du neurone.

b biais interne.

$n = \sum_{j=1}^{R} w_{1,j} P_j - b$: Représente la valeur de la somme pondéré.

f : Fonction d'activation.

$a = f(n)$: Sortie du neurone.

I.5 Fonction d'activation

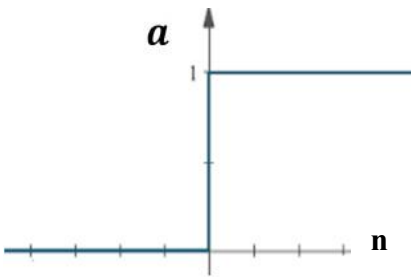
La fonction d'activation est une fonction mathématique appliquée à un signal en sortie d'un neurone artificiel. Le terme de "fonction d'activation" vient de l'équivalent biologique "potentiel d'activation", seuil de stimulation qui, une fois atteint entraîne une réponse du neurone. La fonction d'activation est souvent une fonction non-linéaire [1].

Les trois types de fonctions d'activation les plus utilisées sont :

Fonction seuil :

La relation d'entrée/ sortie

$$a = 0 \ si \ n < 0$$
$$a = 1 \ si \ n \geq 0 \qquad \text{(I.1)}$$

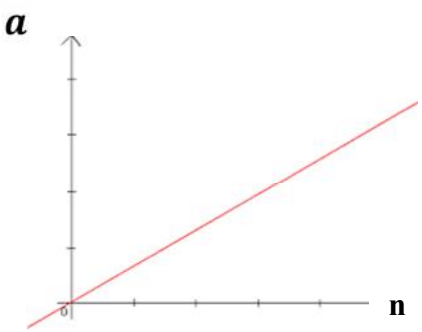


(1)

Fonction linéaire :

Relation d'entrée /sortie

$$a = n \qquad \text{(I.2)}$$

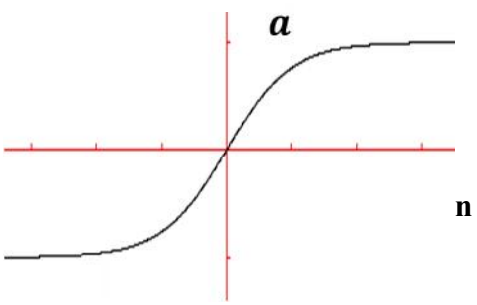


(2)

Fonction tangente hyperbolique :

Relation d'entrée / sortie

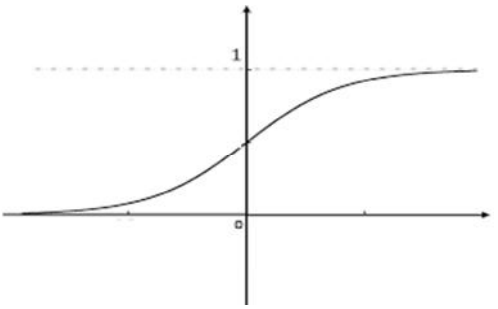
$$a = \frac{e^{n}-e^{-n}}{e^{n}+e^{-n}} \qquad \text{(I.3)}$$



(3)

Fonction sigmoïde :

La relation d'entrée/ sortie



$$a = \frac{1}{1+exp^{-n}} \qquad \text{(I.4)}$$

(4)

Figure I.3: Différents types de fonction de transfert
(1) seuil, (2) linéaire, (3) tangente hyperbolique et (4) sigmoïde.

Les caractéristiques des fonctions d'activation sont les suivantes :

- la monotonie : la fonction d'activation est toujours croissante.
- le seuillage : possède une valeur au-dessous de laquelle sa valeur est négligeable.
- la saturation : elle possède une valeur maximale au-dessus de laquelle sa valeur de réponse est essentiellement fixe, ceci permet d'éviter de propager de grandes valeurs dans le réseau.

I.6 Structure de réseau monocouche et multicouche

I.6.1 Réseau monocouche

La structure d'un réseau monocouche est telle que des neurones organisés en entrée soient entièrement connectés à d'autres neurones organisés en sortie par des poids modifiables [2].

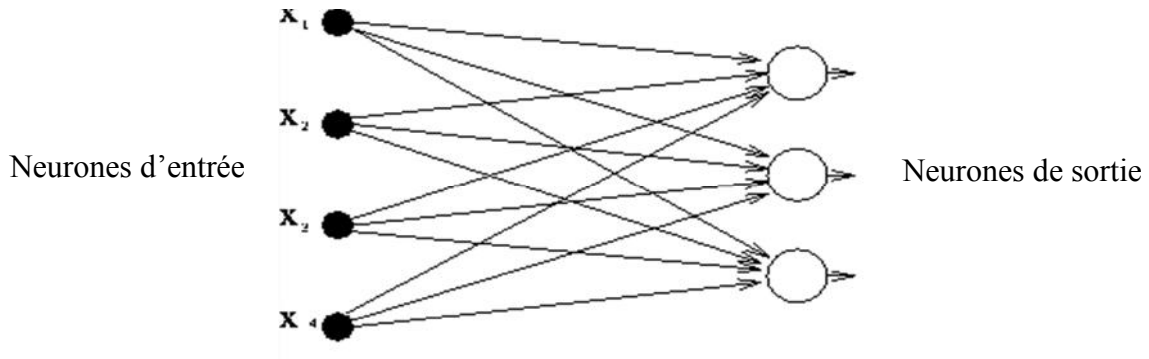


Figure I.4 Architecture de réseau à une seule couche

I.6.2 Réseau multicouches

Le réseau multicouches MLP (*multi Layer Perceptron*) est une extension de réseau monocouche, avec une ou plusieurs couches cachées entre l'entrée et la sortie (voir figure I.5). Les entrées des neurones de la deuxième couche sont, en fait, les sorties des neurones de la première couche. Les neurones de la première couche sont reliés au monde extérieur et reçoivent tous le même vecteur d'entrée. Ils calculent alors leurs sorties qui sont transmises aux neurones de la deuxièmes couche. Les sorties des neurones de la dernière couche forment la sortie du réseau [2].

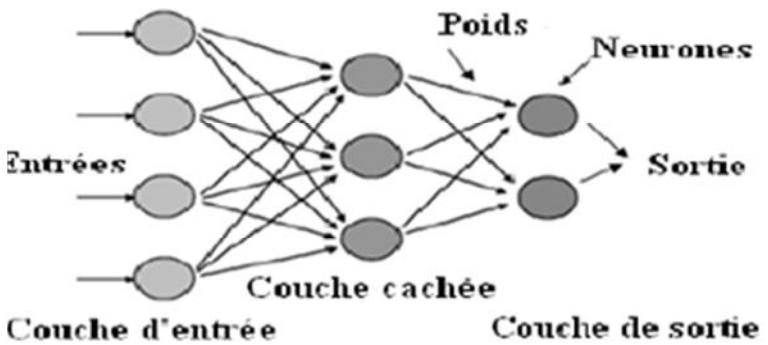


Figure I.5 : architecture de réseaux multicouche

I.7 Types de réseaux de neurones artificiels (RNA)

Un réseau de neurones artificiels (RNA) est un ensemble de neurones formels (d'unités de calcul simples, de nœuds processeur) associés en couches (ou sous-groupes) et fonctionnent en parallèle.

Dans un réseau, chaque sous-groupe fait un traitement indépendant des autres et transmet le résultat de son analyse au sous-groupe suivant. L'information donnée au réseau va

donc se propager couche par couche, de la couche d'entrée à la couche de sortie, en passant soit par aucune, soit par plusieurs couches intermédiaires (dites couches cachées).

On peut distinguer essentiellement deux types de réseaux de neurones artificiels (RNA), les réseaux non boucles (*feed-forward*) et les réseaux boucles (*back-forward*):

I.7.1 Réseaux de neurones non bouclés (statique ou non récurrent)

Un réseau de neurones non bouclé réalise une (ou plusieurs) fonction algébrique de ses entrées par composition des fonctions réalisées par chacun de ses neurones. Dans un tel réseau (voir figure I.6), le flux d'informations circule des entrées vers les sorties sans retour en arrière (acyclique) [7].

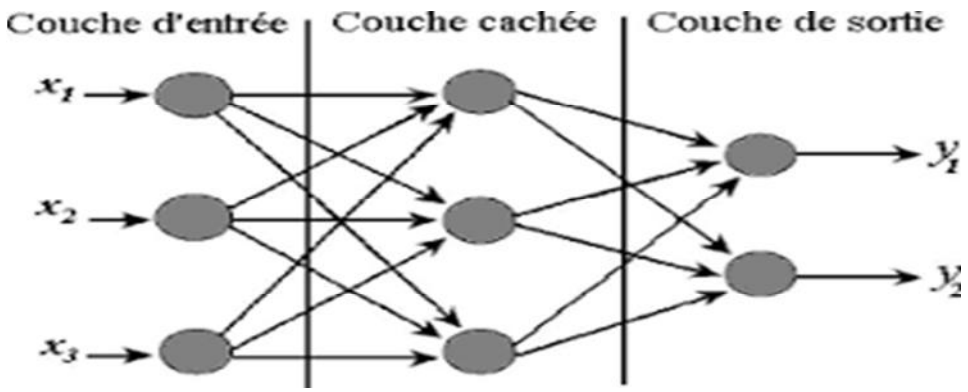


Figure I.6: Réseau de neurones non bouclé

Tout neurone dont la sortie est une sortie du réseau est appelé « neurone de sortie ». Les autres, qui effectuent des calculs intermédiaires, sont des « neurones cachés ».

I.7.2 Réseaux de neurones bouclés (dynamique ou récurrents).

L'architecture la plus générale pour un réseau de neurones est le « réseau bouclé », dont le graphe des connexions est cyclique, lorsque on se déplace dans le réseau en suivant le sens des connexions ; il est possible de trouver au moins un chemin qui revient à son point de départ (un tel chemin est désigné sous le terme de « cycle ») (voir figure I.7). La sortie d'un neurone du réseau peut donc être fonction d'elle-même ; cela n'est évidemment concevable que si la notion de temps est explicitement prise en considération [7].

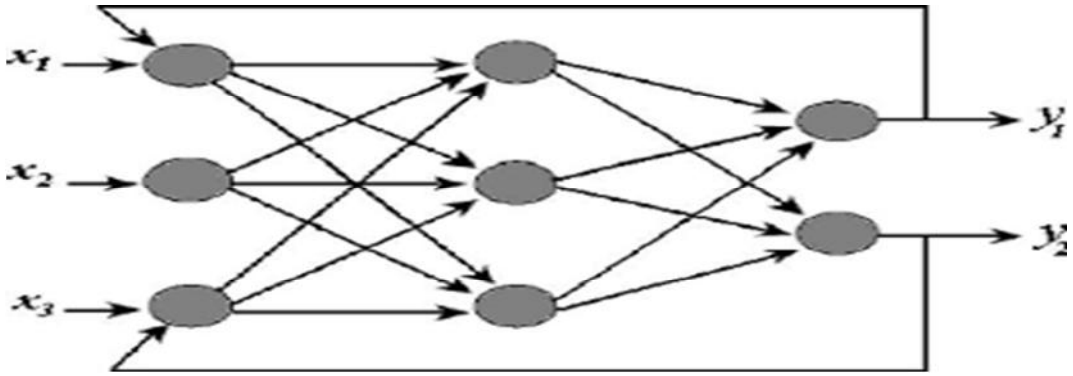


Figure I.7 : Réseau de neurone bouclé

Les connexions récurrentes ramènent l'information en arrière par rapport au sens de propagation défini dans un réseau multicouche. Pour éliminer le problème de la détermination de l'état du réseau par bouclage, on introduit sur chaque connexion « en retour » un retard qui permet de conserver le mode de fonctionnement séquentiel du réseau. Le graphe des connexions de réseaux récurrents est cyclique. Ces réseaux sont décrits par un système d'équations aux différences.

La structure par couche peut être utilisée pour résoudre un certains nombre de problèmes, par exemple l'approximation de fonctions et la classification. Dans ce travail, nous nous sommes intéressés aux réseaux de neurones perceptron multicouches de type non bouclé. Une fois l'architecture est bien choisie, il reste à former le réseau par apprentissage.

I.8 Apprentissage des réseaux de neurone artificiels

L'apprentissage des réseaux de neurones artificiels est une phase qui permet de déterminer ou de modifier les paramètres du réseau, afin d'adopter un comportement désiré. Cela est réalisé en adaptant, grâce à certaines règles, les vecteurs de poids. Plusieurs algorithmes d'apprentissage ont été développés depuis la première règle d'apprentissage de Hebb en 1949 [4]. Selon le degré de supervision, les algorithmes d'apprentissage peuvent être classés en trois catégories : supervisé, non supervisé et semi supervisé.

I.8.1 Apprentissage supervisé

Dans ce type d'apprentissage, on cherche à imposer au réseau un fonctionnement donné en forçant à partir des entrées qui lui sont présentées, les sorties du réseau à prendre des

valeurs données en modifiant les poids synaptiques. Le réseau se comporte alors comme un filtre dont les paramètres de transfert sont ajustés à partir des couples entrée/sortie présentés. L'adaptation des paramètres de réseau s'effectue à partir d'un algorithme d'optimisation, l'initialisation des poids synaptiques étant le plus souvent aléatoire [4].

I.8.2 Apprentissage non supervisé

Dans ce cas, des exemples au « prototypes » ou « patrons » sont présentés au réseau qu'on laisse s'auto-organiser au moyen de lois locales qui régissent l'évolution des poids synaptiques. Ce mode d'apprentissage est aussi appelé « apprentissage par compétition » [4]. Ce type d'apprentissage est généralement utilisé pour la classification automatique des entrées, où le réseau apprend les caractéristiques des données d'entrée.

I.8.3 Apprentissage semi supervisé

Le mode semi supervisé suppose qu'un comportement de référence précis n'est pas disponible, mais qu'en revanche, il est possible d'obtenir des indications qualitatives (correct / incorrect) sur les performances du réseau [2].

I.9 Règles d'apprentissage

L'apprentissage se fait selon des règles bien définies, parmi les règles d'apprentissage on peut citer :

I.9.1 Règle de Hebb

La loi d'apprentissage la plus simple est basée sur la règle de Hebb [4] qui suit en fait le comportement du neurone biologique : si deux neurones interconnectés sont simultanément activés, alors le poids de la connexion qui les relie doit être renforcé, d'où l'équation [12]:

$$w_{ij}(t+1) = w_{ij}(t) + \partial w_{ij}(t) \qquad \text{(I.4)}$$

Avec :

$w_{ij}(t)$: Valeur à l'instant t du poids liant le neurone j au neurone i

$w_{ij}(t) = x_i \square x_j$ (La coactivité est modélisée comme le produit des deux valeurs d'activation).

L'algorithme d'apprentissage modifie de façon itérative (petit à petit) les poids pour adapter la réponse obtenue à la réponse désirée. Il s'agit en fait de modifier les poids lorsqu'il y a erreur seulement comme suit [12] :

1. Initialisation des poids et du seuil S à des valeurs (petites) choisies aléatoire.

2. Présentation d'une entrée $E_l = (e_1, \dots e_n)$ de la base d'apprentissage.

3. Calcul de la sortie obtenue x pour cette entrée :

Avec $a = \sum(w_i \square e_i) - S$ (la valeur de seuil est introduite ici dans le calcul de la somme Pondérée)
et $x = signe\ (a)$ (si $a > 0$ alors $x = +1$ sinon $a \leq 0$ alors $x = -1$)

4. Si la sortie x est différente de la sortie désirée d_l pour cet exemple d'entrée E_l alors modification des poids (μ est une constante positive, qui spécifie le pas de modification des poids) :

$$w_{ij}(t+1) = w_{ij}(t) + \mu.(x_i \ . \ x_j) \qquad \text{(I.5)}$$

I.9.2 Règle de rétro propagation

L'algorithme d'apprentissage par rétro-propagation du gradient de l'erreur est un algorithme itératif qui a pour objectif de trouver les poids des connexions minimisant l'erreur quadratique moyenne commise par le réseau sur l'ensemble d'apprentissage **[5]**. Cette minimisation par une méthode du gradient conduit à l'algorithme d'apprentissage de rétro-propagation.
Avant d'énoncer l'algorithme de rétro-propagation, nous citons les différentes notations utilisées dans celui-ci, qui sont :

w_{ik}: La connexion entre le neurone i et le neurone k

w_{kh}: La connexion entre le neurone k et le neurone h

net_i: L'entrée totale du neurone i de la couche (3)

net_k : L'entrée totale du neurone k de la couche (2)

δ_i: L'erreur quadratique

α: Pas d'apprentissage

y La sortie

θ Le seuil

Les différentes étapes de la règle de rétro-propagation sont :

1- Appliquer un vecteur d'entrée, aux unités d'entrée :

$$x = (x_1, x_2, x_3, \cdots x_N)^p$$

2- Calculer les entrées totales des neurones cachés :

$$net_k^{(2)} = \sum_{j=1}^{p} w_{kh} x_j + \theta_k^{(2)} \qquad \text{(I.6)}$$

3- Calculer les sorties des neurones cachés :

$$y_k^{(2)} = F\left(net_k^{(2)}\right) \qquad \text{(I.7)}$$

4- Calculer les entrées totales des neurones de la couche de sortie :

$$net_i^{(3)} = \sum_{k=1}^{n} w_{ik} y_k^{(2)} + \theta_i^{(3)} \qquad \text{(I.8)}$$

5- Calculer les sorties réelles des unités de la 3éme couche (couche de sortie)

$$y_i^{(3)} = F\left(net_i^{(3)}\right) \qquad \text{(I.9)}$$

6- Calculer les termes d'erreur sur chaque unité de sortie

$$\delta_k^{(3)} = Y_i^{(3)} - Yd_i^{(3)} \qquad \text{(I.10)}$$

7- Calculer les termes d'erreur sur chaque unité cachée

$$\delta_k^{(2)} = F(net_k^{(2)}) \sum_{i=1}^{n} \delta_i^{(3)} w_{kh}^{(3)} \quad \text{(I.11)}$$

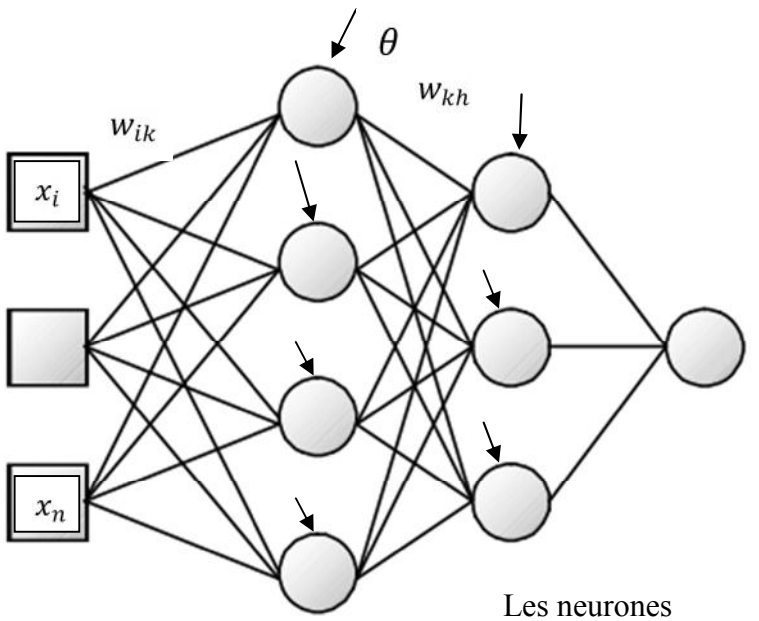
Notons que les termes d'erreur pour les unités de la couche cachée sont calculés avant que les connexions des poids aux unités de la couche cachée n'aient été mises à jour. Ainsi l'erreur est rétro-propagée de la sortie vers l'entrée, d'où le nom rétro-propagation.

8- Mettre à jour les poids de la couche de sortie

$$w_{ik}^{(3)}(t+1) = w_{ik}^{(3)}(t) - \alpha \delta_i^{(3)} Y_k^{(2)} \quad \text{(I.12)}$$

9- Mettre à jour les poids de la couche cachée

$$w_{kh}^{(2)}(t+1) = w_{kh}^{(2)}(t) - \alpha \delta_k^{(2)} Y_h^{(1)} \quad \text{(I.13)}$$



Les entrées x_i

La sortie y_h

Les neurones de la couche 2

Les neurones de la couche 3

Répéter ce processus jusqu'à ce que l'erreur soit atteinte pour tous les exemples, c'est-à-dire lorsqu'elle devienne suffisamment petite pour chaque paire d'apprentissage.

I.10 Intérêts et limites des réseaux de neurones

I.10.1 Intérêts des réseaux de neurones

On peut trouver plusieurs intérêts des réseaux de neurones [5]:

- **Souplesse** : les réseaux de neurones sont capables de traiter une gamme très étendue de problème. Leur résultat peut être une régression, une classification ou encore une analyse de clusters (*clustering neuronale*).

- **Bonne résolution** : ils donnent de bons résultats même dans des domaines complexes car ils sont beaucoup plus puissants que les statistiques ou les arbres de décisions.
- **Bonne adaptation** : une fois que les données sont codées, ils traitent aussi bien des variables continues qu'énumératives.

- **Parallélisme massif** : les réseaux de neurones sont constitués d'unités de calcul qui peuvent opérer d'une manière parallèle. La plupart des implémentations des réseaux de neurones peuvent être facilement converties d'une version séquentielle à une version parallèle.

- **Outils disponibles** : il existe de nombreux produits sur le marché intégrant la technique des réseaux de neurones.

I.10.2 Limite des réseaux de neurones

Nous pouvons facilement dégager les limites de l'approche des réseaux de neurones pour la résolution des problèmes [5][8].

- **Codage des entrées** : toutes les entrées d'un réseau de neurone doivent se trouver dans un intervalle défini en général entre 0 et 1, ce qui entraine des transformations qui impliquent des traitements supplémentaires, et risque de fausser les résultats.

- **Détermination de la taille** : afin que l'échantillon fournisse de bons résultats, sa taille doit être calculée en fonction du nombre d'entrée, du nombre des couches et de taux de connexion ce qui entraine une augmentation du nombre d'exemples qui ne sont pas toujours disponibles.

- **Performance** : le nombre de calcule à effectuer pour définir un réseau optimal peut être très consommateur de puissance, ce qui peut donner de mauvaises performances à cette technique.

- La faiblesse des capacités d'explication des résultats.

- la sensibilité des RNA a certains paramètres, qui fond que le comportement global est susceptible de changer drastiquement à cause d'une perturbation mineure.

I.11 Domaines d'application des réseaux de neurones

On peut trouver de nombreux exemples d'utilisations des réseaux de neurones [4] :

- **Traitement d'image** : reconnaissance de caractères et de signatures, compression d'image, reconnaissance de forme, cryptage, classification.
- **Traitement de signal** : filtrage, classification, identification de sources, traitement de la parole.
- **Contrôle** : commande de processus, diagnostic de pannes, contrôle qualité, robotique.
- **Optimisation** : planification, allocation de ressources, tournées de véhicules, régulation de trafic, gestion, gestion, finance.
- **Simulation** : simulation boite noir, prévision météorologique, recopie de modèles.

I.12 Conclusion

Les réseaux de neurones formels possèdent une propriété remarquable, qui est l'apprentissage, qui est à l'origine de leurs intérêts dans des domaines et des applications très divers.

Un réseau de neurones se distingue, en général, par le type de neurone formel qu'il utilise, la règle d'apprentissage qui le décrit et l'architecture définissant les interconnexions entre les neurones.

Pour un problème posé, le choix du type de réseau de neurones, a utilisé, est basé sur les points essentiels suivants : le nombre de couches, le nombre de neurones par couches, la fonction d'activation, le type d'apprentissage, le choix et la préparation de données utilisées pour l'apprentissage.

Dans le chapitre suivant nous nous intéresserons au type de réseaux de neurones, les plus utilisés dans la littérature, perceptron multi-couches non bouclés, en particulier l'ELM (*Extreme Learning Machine*). Ce réseau fera l'objet du 2ème chapitre.

Chapitre II

Description et Fonctionnement du réseau ELM

(*Extreme Learning Machine*)

II.1 Introduction

Pour que les réseaux de neurones multicouches de type *feed-forward* fonctionnent comme des approximateurs de fonctions, les algorithmes d'apprentissage conventionnels doivent toujours ajuster les paramètres du réseau, en particulier ceux de la couche cachée. Ce qui nécessite un temps d'apprentissage relativement long par rapport à la taille et la dimension des données utilisées. En plus, le choix de l'architecture du réseau est d'une grande importance pour la résolution du problème d'approximation.

Contrairement à ces considérations, un nouveau réseau a vu, récemment, qui ne se soucie pas beaucoup de taille du réseau. En plus, d'un point de vue d'approximation de fonctions, les paramètres de la couche cachée sont générés de manière aléatoire et restent constants au lieu d'être ajustés. Seuls les poids de sortie sont calculés par une simple régression linéaire. Ce réseau est connu sous l'appellation réseau de neurones ELM (*Extrême Learning Machine*). Dans ce chapitre, nous allons vous présenter ce réseau ELM qui a été proposée par **Guang-Bin Huang en 2006[9].**

II.2 Présentation du réseau ELM (*Extreme Learning Machine*)

Le terme *Extreme Learning Machine* (ELM) fait référence à une procédure d'apprentissage extrêmement rapide, ce qui est d'un grand intérêt dans l'histoire des réseaux de neurones. L'ELM est un nouvel algorithme d'apprentissage pour les réseaux neuronaux de type *feed-forward* à couche cachée unique. Par rapport à l'algorithme conventionnel d'apprentissage du réseau neuronal, il surmonte la vitesse d'entraînement lent et le problème d'ajustement excessif.

L'ELM repose sur la théorie de la minimisation du risque empirique et son processus d'apprentissage n'a besoin que d'une seule itération. L'algorithme évite plusieurs itérations et la minimisation locale. Sa spécifié réside dans l'initialisation aléatoire des poids de la couche cachée et qui ne sont jamais mis à jours. Les poids de sortie sont appris en une seule itération et sont calculés par une simple régression linéaire. Le nom "*Extreme Learning Machine*" (ELM) a été donné à ce modèle par Guang-Bin Huang [9].

II.2.1 Architecture du réseau ELM

Le réseau ELM est un réseau de type multicouche *feed-forward* avec une seule couche cachée comme représenté sur la figure II.1. La couche d'entrée permet d'introduire les

données à l'intérieur du réseau, le nombre de neurones dans cette couche, dépend de la dimension du vecteur de caractéristiques qui décrit les données d'entrée en question. Concernant le nombre de neurones de la couche de sortie, dans le cadre de classification, est égal au nombre de classes. Reste maintenant le nombre de neurones de la couche cachée, nous étudierons, dans les parties suivantes, son influence sur le taux de la classification.

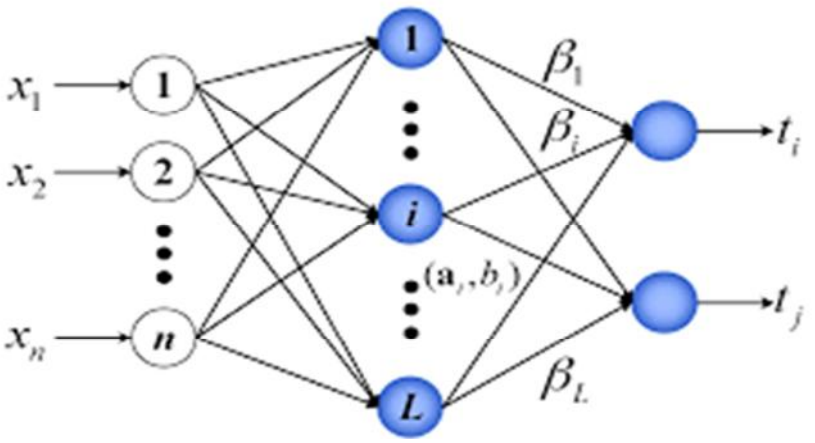


Figure II.1 L'architecture de réseau ELM

$\boldsymbol{a_i}$ **:** C'est le vecteur de poids qui relie le $i^{ème}$ nœud caché aux nœuds d'entrée.

$\boldsymbol{\beta_i}$ **:** C'est le vecteur de poids qui relie le $i^{ème}$ nœud caché aux nœuds de sortie.

$\boldsymbol{b_i}$ **:** C'est le seuil de $i^{ème}$ nœuds cachés.

II.2.2 Fonctionnement du réseau ELM

Pour N échantillons (x_i, t_i), où $x_i = [x_{i1}, x_{i2}, \cdots, x_{in}]^T \in R^n$ et le vecteur de caractéristiques de l'i^{éme} échantillons de dimension n et $t_i = [t_{i1}, t_{i2}, \cdots, t_{im}]^T \in R^m$, est le vecteur de valeurs d'apprentissage de l'i^{éme} échantillons ou m classes. $(x_i, t_i) \in R^n \times R^m (i = 1,2, \cdots, N)$, multicouche *feed-forward* avec une seule couche cachée standard avec $\tilde{N}$ noeuds cachés et fonction d'activation $f(x)$ sont mathématiquement des modèles comme :

$$\sum_{i=1}^{\tilde{N}} \beta_i f_i(x_i) = \sum_{i=1}^{\tilde{N}} \beta_i f(a_i . x_j + b_i) = t_j, j = 1, \cdots, N \qquad (II.1)$$

Avec $a_i = [a_{i1}, a_{i2}, \cdots, a_{in}]^T$ est le vecteur de poids qui relie le i^{eme} nœud caché et les nœuds d'entrée, et b_i est le seuil du i^{eme} nœud caché et $\beta_i = [\beta_{i1}, \beta_{i2}, \cdots, \beta_{im}]^T$ est le vecteur de poids reliant le i^{eme} noeud caché et les nœuds de sortie. $a_i \cdot x_j$ Représente le produit scalaire de $a_i \; et \; x_j$, et la fonction d'activation choisit généralement est une sigmoïde.

L'équation ci-dessus (1) peut être écrite de manière compacte comme suit :

$$H\beta = T \tag{II.2}$$

Avec $H(a_1, \cdots, a_{\tilde{N}}, b_1, \cdots, b_{\tilde{N}}, x_1, \cdots x_N)$

$$H = \begin{bmatrix} f(a_1 . x_1 + b_1) & \cdots & f(a_{\tilde{N}} . x_1 + b_{\tilde{N}}) \\ \vdots & \cdots & \vdots \\ f(a_1 . x_N + b_1) & \cdots & f(a_{\tilde{N}} . x_N + b_{\tilde{N}} \end{bmatrix}_{N \times \tilde{N}}$$

$$\beta = \begin{bmatrix} \beta_1^T \\ \vdots \\ \beta_{\tilde{N}}^T \end{bmatrix}_{\tilde{N} \times m} \quad , \quad T = \begin{bmatrix} t_1^T \\ \vdots \\ t_N^T \end{bmatrix}_{N \times m}$$

Comme indiqué par Huang et al [9], H s'appelle la matrice de sortie de couche cachée du réseau neuronal. L'équation (1) peut être écrite sous forme matricielle comme suit :

$$\begin{bmatrix} f(a_1 . x_1 + b_1) & \cdots & f(a_{\tilde{N}} . x_1 + b_{\tilde{N}} \\ \vdots & \cdots & \vdots \\ f(a_1 . x_N + b_1) & \cdots & f(a_{\tilde{N}} . x_N + b_{\tilde{N}} \end{bmatrix}_{N \times \tilde{N}} \begin{bmatrix} \beta_1^T \\ \vdots \\ \beta_{\tilde{N}}^T \end{bmatrix}_{\tilde{N} \times m} = \begin{bmatrix} t_1^T \\ \vdots \\ t_N^T \end{bmatrix}_{N \times m} \tag{II.3}$$

II.2.3 Algorithme ELM

L'algorithme d'apprentissage du réseau ELM proposé un algorithme **Guang-Bin Huang [9],** Compte tenu d'un jeu d'apprentissage $N = \{(x_i, t_i) | x_i \in R^n, t_i \in R^m, i = 1, \cdots, N\}$ fonction d'activation f(x) et nombre de nœud caché $\tilde{N}$, peut être résumé en ces étapes suivantes :

Étape 1: affectez de manière aléatoire les poids d'entrée $a_i$et bais b_i , $i = 1, \cdots, \widetilde{N}$.

Étape 2: calculer la matrice de sortie de la couche cachée H.

Étape 3: calculer le poids de sortie β, tel que :

$$\beta = H^{+}T \tag{II.4}$$

$avec\ T = [t_1, \cdots, t_N]^T$

Contrairement à la compréhension la plus commune, tous les paramètres des SLFN doivent être ajustés, les poids d'entrée a_i et le biais b_i ne sont, en fait, pas nécessairement réglés et la matrice de sortie de couche cachée H peut effectivement rester inchangée, une fois que les valeurs aléatoires ont été attribuées à ces paramètres au début de l'apprentissage.

Pour l'apprentissage de ce réseau ELM, **Guang-Bin Huang [9]** s'est basé sur les deux théorèmes suivants :

- ***Théorème 2 :*** *Compte tenu de toute petite valeur positive* $\varepsilon > 0$ *et fonction d'activation* $f: R \to R$ *qui est infiniment différentiable dans n'importe quel intervalle, il existe* $\widetilde{N} \leq N$ *tel que pour N échantillons* (x_i, t_i) *avec* $x_i \in R^n$ *et* $t_i \in R^m$ *, avec* x_i *est le vecteur de caractéristiques de l'échantillon i et* t_i *le vecteur des valeurs d'appartenance de l'échantillon i aux m classes. Pour tous les* a_i *et* b_i *choisis de façon aléatoire, selon une distribution de probabilité continue, alors avec une probabilité unique, on a :*

$$\| H_{N\times\widetilde{N}}\beta_{\widetilde{N}\times m} - T_{N\times m} \| < \varepsilon \tag{II.5}$$

D'après l'énoncé du théorème 2, la limite supérieure de nombre de neurones dans la couche cachée est le nombre d'échantillons pour l'apprentissage, soit $\widetilde{N} \leq N$ pour tous les poids d'entrée et les biais de couche cachée sont choisis de manière aléatoire et si seulement la fonction d'activation est infiniment différentiable.

- ***Théorème 1 :*** *Étant donné un réseau de neurone à une seule couche cachée standard avec N nœuds cachés et fonction d'activation* $f: R \to R$ *qui est infiniment différentiable dans n'importe quel intervalle, pour N échantillons* (x_i, t_i) *Où* $x_i \in R^n et\ t_i \in R^m$*, pour tout* a_i *et* b_i *choisi de manière aléatoire parmi tous les intervalles de* R^n *et* R*, respectivement, selon toute distribution de probabilité*

continue, alors avec une probabilité unique, la matrice de sortie de couche cachée H du réseau de neurone à une seule couche cachée est inversible, donc

$$\|H\beta - T\| = 0. \quad \text{(II.6)}$$

Si le nombre $\widetilde{N}$ de nœuds cachés est égal au nombre N d'échantillons d'entraînement, $\widetilde{N} = N$, la matrice H est carrée et inversible lorsque les vecteurs de poids d'entrée a_i et les biais cachés b_i sont choisis de manière aléatoire et les réseaux à une seule couche cachée de type *feed-forward* peuvent approximer ces échantillons de formation avec une erreur nulle.

Cependant, dans la plupart des cas, le nombre de nœuds cachés est beaucoup moins élevé que le nombre d'échantillons de formation, $\widetilde{N} \square N$, H est une matrice non carrée et elle ne peut d'être inverser, un tel système $H\beta = T$ n'admet pas de solution. La seule solution acceptable à ce problème est de chercher la solution de la norme minimale des moindres carrés qui est donnée par la relation suivante :

$$\beta' = \min_{\beta} \|H\beta - T\| \quad \text{(II.7)}$$

La solution de la norme minimale des moindres carrés est l'unique solution parmi toutes celles qui réalisent le minimum$\|H\beta - T\|$. Cette solution de la norme minimale est calculée en utilisant la relation suivante :

$$\beta' = H^{+}T \quad \text{(II.8)}$$

Avec : H^{+} est l'inverse généralisé de Moore-Penrose de la matrice H, en utilisant la décomposition en valeurs singulières (*SVD, Sigular Value Decomposition*).

II.2.4 Calcul l'inverse généralisé Moore-Penrose

En mathématiques, et en particulier l'algèbre linéaire. H^{+} Pseudo inverse d'une matrice H est une généralisation de la matrice inverse, le type de matrice pseudo inverse le plus largement connu est le pseudo inverse de Moore-Penrose, qui a été décrit indépendamment par **EH Moore** en **1920**, **Arne Bjerhammar** en **1951** et **Roger Penrose** en **1955[14]**. Le terme inverse généralisé est parfois utilisé comme synonyme de pseudo inverse.

Pour toute matrice H à coefficients réels avec n lignes et p colonnes, permettent de définir la décomposition en valeurs singulières d'une matrice. On peut montrer que cette matrice peut s'exprimer comme :

$$H = U\Delta V^T \tag{II.9}$$

Avec :

U : les vecteurs propres (normalisés) de la matrice HH^T $(U^T U = I)$ on les appelle les vecteurs singuliers à gauche de la matrice H.

V : les vecteurs propres de la matrice $H^T H$ $(V^T V = I)$ on les appelle vecteurs singuliers à droite de la matrice H

Δ : La matrice des valeurs singulières $\Delta = \Lambda^{1/2}$ avec Λ étant la matrice diagonale des valeurs propres de la matrice HH^T et de la matrice $H^T H$.

Cette décomposition en valeurs singulières est facile à vérifier comme une conséquence de la décomposition en valeurs propres d'une matrice symétrique. En particulier :

$$\begin{aligned} HH^T &= (U\Delta V^T) \times (U\Delta V^T)^T = U\Delta V^T U\Delta V^T \\ &= U\Delta^2 U^T \\ &= U\Lambda U^T \end{aligned} \tag{II.10}$$

Et

$$\begin{aligned} A^T A &= (U\Delta V^T)^T \times (U\Delta V^T) = U\Delta V^T U\Delta V^T \\ &= V\Lambda V^T \end{aligned} \tag{II.11}$$

II.3 Caractéristiques d'ELM

On peut trouver plusieurs Caractéristiques d'ELM :

- La vitesse d'apprentissage de l'ELM est extrêmement rapide.

- ELM est meilleures performances de généralisation que l'apprentissage classique

- L'algorithme d'apprentissage ELM semble Beaucoup plus simple que la plupart des algorithmes d'apprentissage pour les réseaux neuronaux feed-forward

- Contrairement aux algorithmes traditionnels classiques qui ne fonctionnent que pour des fonctions d'activation infiniment différentiables, l'algorithme d'apprentissage ELM pourrait facilement être utilisé pour former des SLFN avec de nombreuses fonctions d'activation non différentiables

II.8 Conclusion

Après la description et l'étude du réseau ELM proposé par **Huang**, nous pouvons conclure, que tous les paramètres de la couche cachée (ai et bi) de l'ELM pourraient être générés de manière aléatoire et restent constants, selon une distribution de probabilité continue donnée, sans aucune connaissance préalable (même avant que les données ne soient présentées et que l'apprentissage ELM commence). Donc ces paramètres sont indépendants des données de d'apprentissage, ce qui rend le réseau ELM un approximateurs universel. Vu que seul les poids de sortie sont calculés par une simple régression linéaire, implique que ce réseau a un temps de réponse beaucoup plus court. Dans le chapitre suivant, nous allons évaluer le réseau ELM sur un ensemble de bases de données.

Chapitre III

Evaluation des performances du réseau ELM en classification

III.1 Introduction

L'ELM habituellement utilisé pour la résolution de plusieurs problèmes de régression logistique et de classification. Afin de mesurer les performances du réseau ELM, nous le testons pour la classification de différentes bases de données. En parallèle, nous étudierons l'influence de deux paramètres de ce réseau ELM sur les résultats de la classification. Nous commençons ce chapitre par une brève introduction sur la classification de données et les différents types de classification selon l'aspect supervisé. Par la suite, nous allons décrire les bases de données utilisées pour la classification en utilisant le réseau ELM.

III.2 Définition de la classification

La classification est un outil d'organisation et d'analyse de données volumineuses et complexes. L'objectif de la classification est d'identifier les classes auxquelles appartiennent des objets à partir de traits descriptifs. La classification permet d'avoir des classes compactes (les exemples représentant une classe donnée sont plus proches entre eux que des exemples de toutes les autres classes) et séparables (les classes sont bornées et il n'y pas de recouvrement). Le problème de la classification se décline en deux variantes : l'approche non-supervisée et l'approche non supervisée.

III.2.1 Classification non supervisée

Ce type de classification est aussi appelé « classification automatique » ou encore « *clustering* ». Le but de de classification non-supervisée est de répartir un ensembles de données en un certain nombre de classes, dont les données sont homogènes et similaires au sens d'une seule information les vecteurs descriptifs de ces données, sachant que les exemples disponibles ne sont pas initialement identifiés comme appartenant à telle ou telle classe. Il faut pouvoir affecter un nouvel exemple à une classe identifiée auparavant.

III.2.2 Classification supervisée

Dans ce type de classification, nous disposons d'un ensemble d'objets déjà classés, nous avons des connaissances concernant les classes et leurs nombres, d'où vient l'appellation supervisée. L'objectif de la classification supervisée est d'apprendre une règle qui permet de prédire la classe de nouvel exemple parmi les classes disponibles, donc affecter à chaque nouvel exemple l'étiquette de la classe qui lui appropriée. Dans notre travail, nous avons utilisé le réseau ELM pour la classification supervisée.

III.3 Description des bases de données utilisées

Pour nos tests, nous avons choisis des bases de données selon différents aspects : grande et petite base, le nombre d'exemples par classe égal ou différent, vecteur de caractéristiques grand et petit, pour pouvoir évaluer le réseau ELM. Les bases de données que nous avons utilisés sont issues de la banque de données *UCI Machine Learning Repository* [13] disponible sur le web, qui est gérée par l'université d'Ivrine de l'état de Californie. Cette banque est une collection de 378 bases de données de domaines très variés, qui est mise à la disposition de la communauté d'apprentissage machine. Les bases de données que nous avons choisis sont : *Iris*, *Breast cancer Wisconsin*, *Dermatology* et *Shuttle*.

La base IRIS : C'est la base de données la plus utilisée dans le domaine de la classification. Elle contient 3 classes de 50 échantillons (soit 33.33%). Chaque classe se réfère à un type de plante Iris qui sont : *Iris Setosa*, *Iris Versicolour* et *Iris Virginica*. Chaque échantillons est décrit par un vecteur 4 caractéristiques qui représentent : la longueur et largeur de la sépale, ainsi ceux du pétale, ils sont exprimées en centimètres. Les trois classes ont les même nombre d'échantillons (ou bien exemples).

La base *Breast cancer Wisconsin* : C'est une base de données du cancer du sein qui a été obtenue auprès des hôpitaux de l'université de Wisconsin USA. Elle contient les informations médicales de 683 cas cliniques relatifs au cancer du sein classés comme bénin ou malin : 444 patientes (soit 65%) sont des cas bénins et 239 patientes (soit 35%) sont des cas malins. Les classes sont disproportionnelles. Chaque échantillon (patiente) est décrit par 10 caractéristiques qui sont calculées à partir d'une image numérisée d'une masse mammaire, qui sont : rayon (moyenne des distances du centre aux points sur le périmètre), texture, périmètre, zone, la douceur (variation locale des longueurs de rayon), compacité, concavité (gravité des parties concaves du contour), points concave (nombre de parties concaves du contour), symétrie et dimension fractale.

La base *Dermatology* : c'est base qui regroupe des échantillons sur différentes maladies dermatologiques (la peau). Elle contient les données de 358 cas relatif à 6 maladies différentes de la peau : *Psoriasis* (31.03%), *dermatite séboreique* (16.76%), *lichen planus* (19.83%), *pityriasis rosea* (13.40%), *dermatite chronique* (13.40%) et *pityriasis rubra pilaris* (5.58%). Ces maladies partagent de 34 caractéristiques histopathologies, qui sont déterminées par une analyse des échantillons sous microscope.

La base shuttle : Cette base de donnée été utilisé dans le projet Européen (Statlog) qui consiste à comparer les performances des algorithmes d'apprentissage machine statique et réseaux de neurones sur les ensembles des donné des zones industrielle.

Cette base de données réunie 14500 exemples qui sont répartis sur 7 classes : *Rad.Flow* (79.15%), *Fpv.Close* (0.0.89%), *Fpv.Open* (0.268%), *High* (14.86%), *Bypass* (5.58%), *Bpv.Close* (0.027%) et *Bpv.Open* (0.013%). 9 caractéristiques avec 7 niveaux classe suivant :

Dans le tableau ci-dessous (tableau III.1), nous avons résumé les 4 bases de données que nous avons utilisé pour évaluer le réseau ELM.

Bases	**N^{bre} d'exemples**	**N^{bre} de caractéristiques**	**N^{bre} de classes**	**N^{bre} d'exemples/classe**
IRIS	150	4	3	1(50), 2(50), 3(50)
Breast cancer Wisconsin	683	10	2	2(444), 4(239)
Dermatology	358	34	6	1(111), 2(60), 3(71), 4(48), 5(48), 6(20)
Shuttle	14500	9	7	1(11478), 2(13), 3(39), 4(2155), 5(809), 6(4), 7(2)

Tableau III.1 : Récapitulatif des toutes les bases utilisées

III.4 Tests et résultats

III.4.1 Tests

Pour toutes nos expériences nous avons utilisé un PC Portable de type du système 32 bits, Processeur AMDE1-2100APU with Radean (TM) HD Graphics 1.00 GHz et la mémoire installé (RAM) 2.00Go.

Nous avons choisi d'évaluer les performances du réseau ELM, en termes de précision et de temps de calcul de classification, à l'aide de ces 4 bases de données citées

précédemment. Nous avons divisé, chaque base de données, en deux ensembles : un ensemble d'apprentissage et un ensemble test. En s'appuyant sur les exemples de l'ensemble d'apprentissage, le réseau ELM est entraîné pour prédire l'appartenance d'un nouvel exemple à une classe déjà apprise. Le réseau entraîné sera ensuite utilisé pour la classification de nouveaux exemples, qui sont regroupe dans l'ensemble test, dont la classe d'appartenance est supposée inconnue et aussi évaluer le classifier ELM. Pour étudier l'influence des exemples d'apprentissage sur le taux de classification d'ELM, nous avons réparti les bases de données de 3 manières différentes comme indiqué dans le tableau ci-dessous.

	Ensemble d'apprentissage	**Ensemble de test**
1ère répartition	50%	50%
2ème répartition	25%	75%
3ème répartition	75%	25%

Tableau III.2 : les différentes répartitions

Le deuxième paramètre à étudier, concernant le réseau ELM, est l'influence du nombre de neurones dans la couche cachée. Nous l'avons fait varier le nombre de neurones dans l'intervalle [4,1000]. La performance du réseau ELM est mesurée par le taux de bonne classification sur les ensembles de test et le temps nécessaire pour l'apprentissage du réseau.

III.4.2 Résultats et interprétation de résultats

Le tableau III.3 montre les résultats de classification, des bases de données décrites précédemment, en utilisant le réseau ELM.

Bases	N^{bre} d'exemples d'apprentissage	N^{bre} d'exemples de test	N^{bre} de neurones	Taux de bonne classification (%)	Temps d'apprentissage (s)	Temps de test (s)
Iris	50(25%)	100(75%)	[10, 40]	[95%(10) 86%(40)]	[0.0014 0.0077]	[$2.9752*10^{-4}$ $9.7565*10^{-4}$]
	100(75%)	50(25%)	[10, 60]	[99.64%(20) 90.96%(40)]	[0.006 0.0214]	[$2.1237*10^{-4}$ $6.8634*10^{-4}$]
	75(50%)	75(50%)	[10, 60]	[95.84%(20) 85.25%(60)]	[$6.88*10^{-5}$ 0.0038]	[$1.1012*10^{-4}$ $3.12*10^{-4}$]
Breast cancer Wisconsin	171(25%)	512(75%)	[10, 100]	[96.82%(10) 89.74%(100)]	[0.0039 0.0579]	[0.0014 0.0064]
	512(75%)	171(25%)	[10, 100]	[97.53%(40) 93.91%(100)]	[0.0035 0.00725]	[$9.7154*10^{-4}$ 0.0032]
	342(50%)	341(50%)	[10, 100]	[96.97%(20) 92.95%(100)]	[0.0107 0.0692]	[0.0010 0.0090]
dermatology	92(25%)	274(75%)	[20, 80]	[95.59%(40) 91.93%(80)]	[0.0078 0.0326]	[0.0015 0.0032]
	274(75%)	92(25%)	[20, 100]	[97.69%(80) 95.51%(100)]	[0.0051 0.0630]	[$7.8790*10^{-4}$ 0.0036]
	183(50%)	183(50%)	[20, 100]	[96.98%(100) 94.15%(20)]	[0.0062 0.0568]	[0.0018 0.0029]
Shuttle	3625(25%)	10875(75%)	[20, 1000]	[99.54%(400) 95.65%(20)]	[0.0808 49.5212]	[0.0334 1.6808]
	10875(75%)	3625(25%)	[20, 1000]	[99.62%(200) 96.24%(20)]	[0.1129 68.6961]	[0.0166 0.7959]
	7250(50%)	7250(50%)	[20, 1000]	[99.57%(200) 96.30%(20)]	[0.1006 40.26]	[0.0252 0.6142]

Tableau III. 3 Les résultats de taux de classification (%) des ensembles des données avec l'algorithme ELM.

III.4.3 Influence du nombre d'exemples dans la base d'apprentissage : dans certains cas les données d'apprentissage ne représentent pas toujours parfaitement la réalité des données en question, ce qui peut dégrader les résultats de la classification.

III.4.4 Analyse des résultats

Pour l'analyse des résultats, nous allons étudier l'influence de deux paramètres : le nombre d'exemples d'apprentissage et le nombre de neurones dans la couche cachée.

- **Influence du nombre d'exemples dans la base d'apprentissage**

  Pour la base IRIS le meilleur taux de bonne classification, qui de 99,64%, est obtenu pour l'ensemble d'apprentissage constitué de 75% de l'ensemble de données de la base. Pour les deux autres partitions, c.-à-d. l'ensemble d'apprentissage forme 50% et 25% de l'ensemble des exemples de la base, le résultat est pratiquement le même 95%. Le taux de bonne classification est un peu moindre par rapport à la répartition de 75% de données dans la l'ensemble d'apprentissage, mais il reste toujours un résultat acceptable.

  La base *Breast cancer Wisconsin* nous remarquons presque la même chose.

  Le meilleur taux de bonne classification est 97% obtenu pour l'ensemble d'apprentissage constitué de 75% de l'ensemble de données de la base. Pour 50% et 25% de l'ensemble des exemples de la base, le résultat est pratiquement le même 96% de taux de bonne est un peu moindre par rapport à la répartition de 75% de données dans la l'ensemble d'apprentissage, mais il reste un résultat acceptable.

  La base *Dermatology* Le meilleur taux de bonne classification est 97% pour l'ensemble d'apprentissage constitué de 57% de l'ensemble de données. Et pour 50% on 'a trouvé 96% de taux de bon classification et pour 25% de données dans l'ensemble d'apprentissage c'est 95%, les résultats est presque la même.

  La base *shuttle* nous remarquons que les résultats presque la même chose, le meilleur taux de bonne classification est 99% qui obtenu dans les trois ensemble d'apprentissage qui on a repartes 75%, 50% et 25%, reste toujours un résultat acceptable.

  -Influence du nombre de neurones dans la couche cachée

  La base IRIS nous remarquons qu'avec 10 neurones dans la couche cachée, nous avons obtenus un taux de bonne classification de 95%. Avec un peu plus de neurones, le double, nous avons atteint un taux de bonne classification de 99,64%.en constat que L'écartes statique de l'apprentissage entre les trois repartes d'apprentissage (75%, 50% et 25%) est 4,64. On remarque que la base IRIS n'est pas satisfaite car l'erreur est très grande

La base *Breast cancer Wisconsin* avec 10 neurones dans la couche cachée, nous avons obtenus un taux de bonne classification 96%. Avec 40 neurones nous avons obtenus 97% de taux de bonne de classification. En conclu que l'écartes statique de l'apprentissage entre les trois repartes d'apprentissage est égale a 1

La base *Dermatology* nous remarquons qu'avec 20 neurones dans la couche cachée, nous avons obtenus un taux de bonne classification de 95%. Avec un peu plus de neurones, le double, nous avons atteint un taux de bonne classification de 97%.

Et l'écartes statique entre les trois repartes d'apprentissage qu'on a pré est 0,49.

La base *Shuttle* nous remarquons que quel que soit le nombre de neurone qu'on a pré le taux de bonne classification est 99% et sont écarte statique est de 0,08 en constat que cette base est une bonne performance.

-Influence du temps d'apprentissage

On remarque qu'à chaque fois le taux de bonne classification augmenter le temps de l'apprentissage est diminuait, alors nous avons conclu que la classification des bases de donné qui on a test sont presque condensée.

III.4.5 Résultats expérimentaux sur les graphes

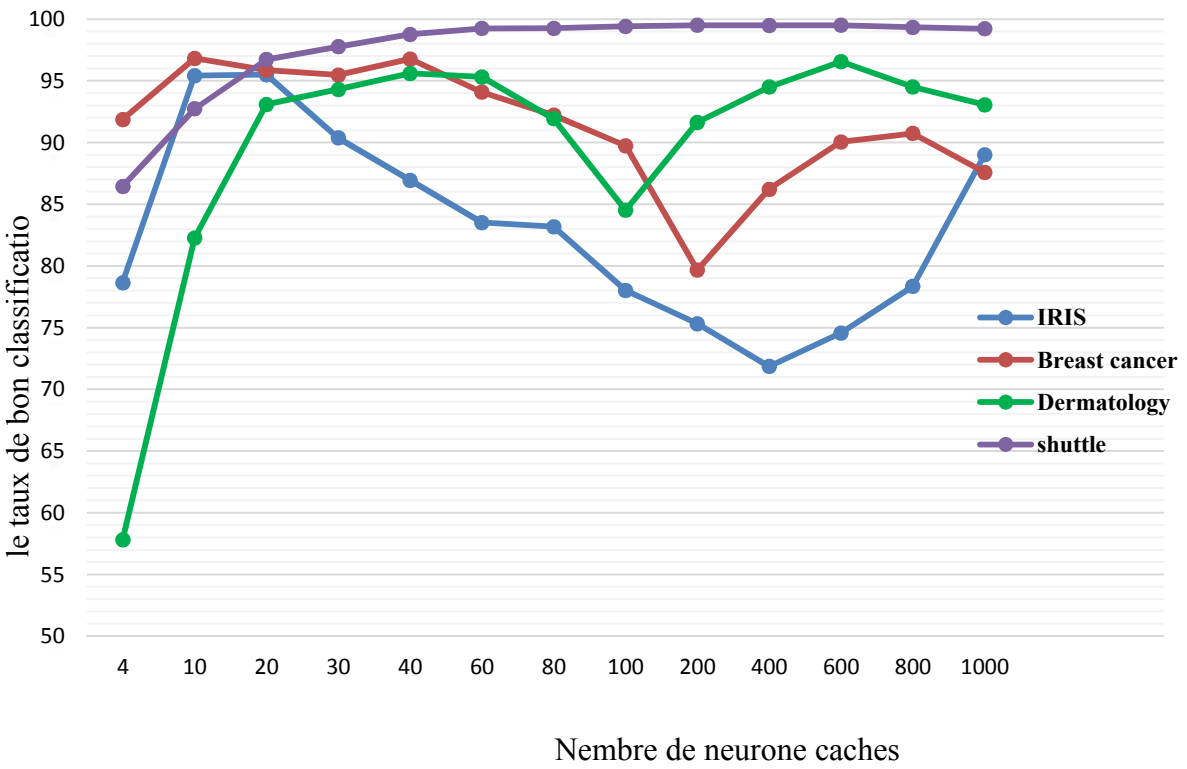


Figure III.1 les résultats de taux de bonne classification en fonction de nombre de neurone cachée avec la répartition d'apprentissage en 25%.

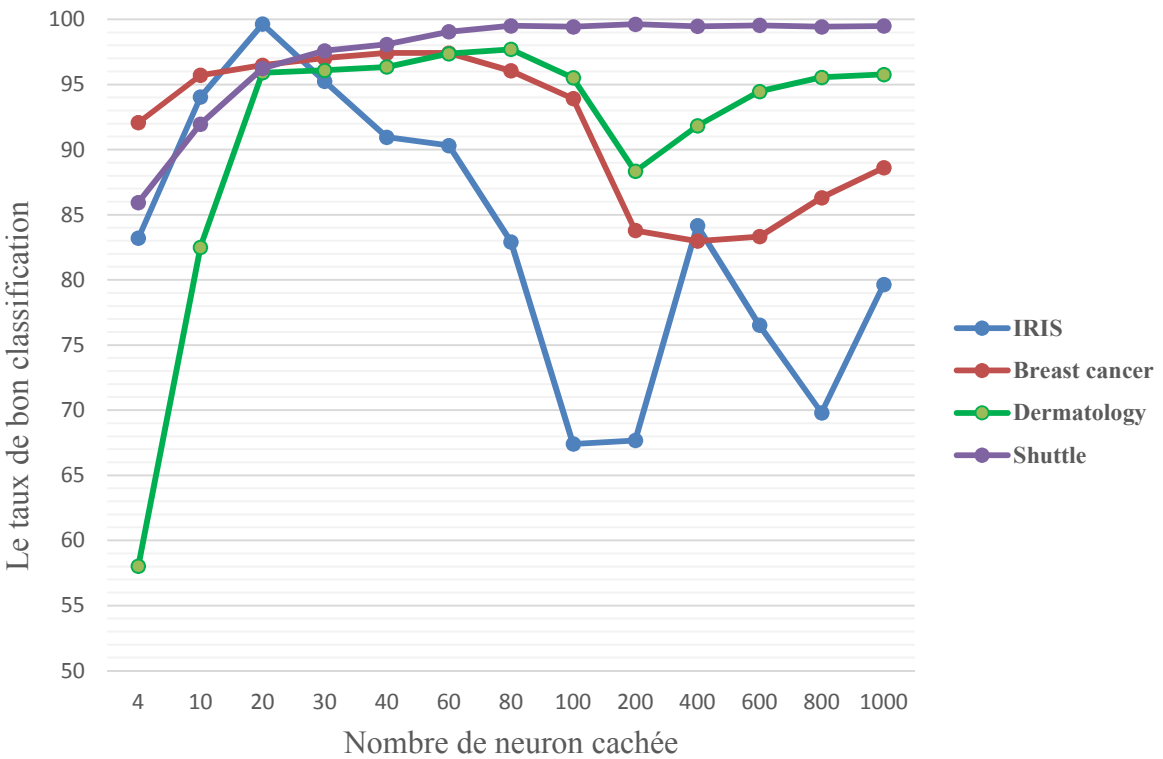


Figure III.2 Les résultats de taux de bonne classification en fonction de nombre de neurone cachée avec la répartition d'apprentissage en 75%.

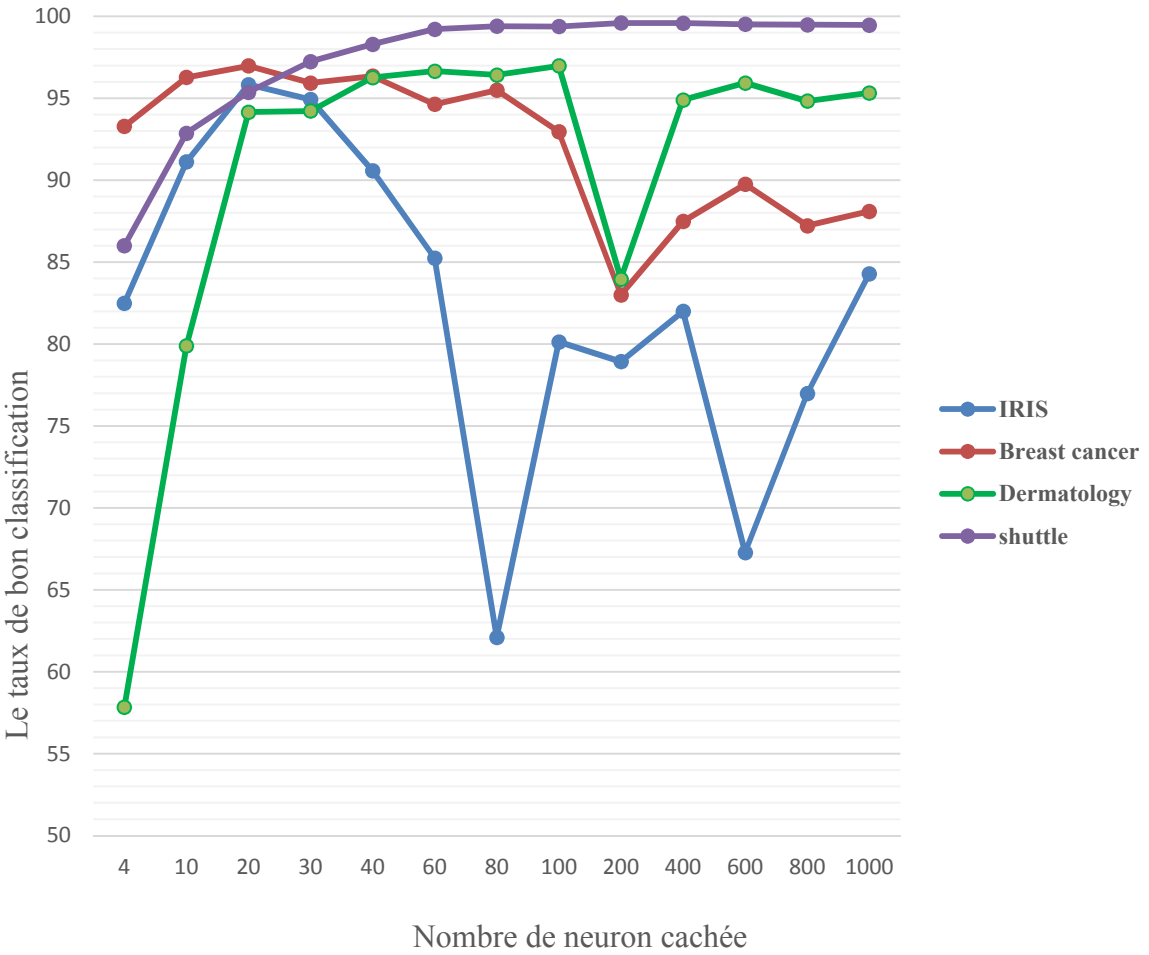


Figure III.3 Les résultats de taux de bonne classification en fonction de nombre de neurone cachée avec la répartition d'apprentissage en 50%.

-interprétation des figures (III.1-III.2-III.3)

D'après les résultats obtenus, le taux de bon classification augmenté à chaque foi augmentant le nombre de neurone cachée jusque a apprendre un bon résultat de taux de classification, mais va diminuait dans certain nombre de neurone cachés presque arrive au nombre d'exemple d'apprentissage qu'on a pré dans les trois réparti qui on a pré (25%, 50% et 75%).

Le taux de bonne classification augmente, jusque arrive a certain de nombre de neurone cachée.

On constate que les bon résultats de taux de bonne classification dans le nombre d' d'apprentissage qui on a près de la **r**épartition du 75% (figure III.2), les résultats de bon classification sont tous supérieur à 96% dans tous les base de donnée par a pour a l'autre répartition (50% et 25%).

III.6 Conclusion

D'prés la description et le fonctionnement de réseau ELM on 'a test la classification de différents base de données d'ELM, pour étudier l'influence des exemples d'apprentissage sur le taux de classification d'ELM.

A partir des résultats qui on a obtenir nous avons remarqué que dans certains cas les données d'apprentissage ne représentent pas toujours la réalité des données, ce qui est peut dégrader les résultats de la classification.

Mais le réseau ELM il reste toujours une bonne performance de généralisation et une rapidité d'apprentissage.

Conclusion générale

Les réseaux de neurone artificiel possèdent une propriété remarquable à l'origine de leurs avontage dans différents domaines et des applications très diverses. Le réseau de neurone se distingue en générale par le types de neurone artificiel qu'il utilisé, la règle d'apprentissage et l'architecture définissant entre les neurones.

Nous avons remarqué que pour un problème posé, c'est le choix de type de réseau de neurone, est basé sur le nombre de couches et le nombre de neurone par couches. pour les réseaux de neurone multicouche de type *Feed-forward* fonctionnent comme des approximateurs de fonctions, en plus le choix de l'architecteur du réseau est d'une grande importance pour la résolution du problème d'approximation.

Contrairement à ces considérations, un nouveau réseau d ELM (*Extrême Learning Machine*). Les paramètres de la couche cachée sont généralement de manière aléatoire est restent constants, selon une distribution de probabilité continue donnée. Donc ces paramètre sont indépendant des donnée de l'apprentissage ce qui rend le réseau ELM un approximateurs universel que les poids de sortie sont calculés par une simple régression linéaire, implique que ce réseau a un temps de réponse beaucoup plus court, et utilisé pour la résolution de plusieurs problème de régression logistique et de classification.

Nous avons testés pour la classification de différentes bases de données, en parallèle nous avons étudié l'influence de deux paramètres de ce réseau ELM sur les résultats de la classification dans l'environnement MATLAB 10.7.

Finalement a partir des résultats qui on a obtenir nous avons remarqué que dans certains cas les données d'apprentissage ne représentent pas toujours la réalité des données, ce qui est peut dégrader les résultats de classification.

Mais le réseau ELM reste toujours une bonne performance de généralisation on est une rapidité d'apprentissage.

[1] Marc Parizeau « Réseaux de Neurones. GIF- 21140 et GIF-64326 » Université LAVAL, 2006.

[2] O.OUKACINE « Utilisation de réseaux de neurones pour la reconstitution » mémoire Magister, Université Mouloud Mammeri de Tizi-Ouzou, 2012.

[3] Mémoire de fin d'études pour l'obtention de diplôme d'ingénieur en Automatique à l'université MMTO. Application des Réseaux de Neurone. Année 2010.

[4] Pierre Bome, Mohamed Benrejeb et Joseph Haggége « Les Réseaux de Neurone » TECHNIP, 2007.

[5] S.MEZAGUER « Classification par Réseaux de Neurones Application aux Médicales cérébrales » mémoire master en Automatique, Université Mouloud Mammeri de Tizi-Ouzou, 2013.

[6] Léon PERSONNAZ et Isabelle RIVALS « Réseaux de neurones formels pour la modélisation la commande et la classification », CNRS Paris, 2003.

[7] Dreyfus G., Martinez J.M., Samuelides M., Gordon M.B., Badran F., Thiria S. et Hérault L., « Réseaux de neurones Méthodologie et applications », EYROLLES, 2004.

[8] Jean-Michel Renders « Algorithmes génétiques et réseaux de neurones » Hermès reproduit et achevé d'imprimer par l'imprimerie FLOCH à Mayenne ,1995.

[9] l'article de base extrême Learning machine théorie and application par « Guang-Bin Huang, Qin-yu zhu et Kheong siew » en 2006.

[10] Hervé Abdi et Dominique Valentin « Mathématique pour les sciences cognitives avec des applications aux réseaux de neurones au traitement de signal à l'imagerie cérébrale et à la statistique » Presses Université de Grenoble, 2006.

[11] S.BAZIZ « Algorithme génétique pour la classification des données par la méthode des K-Means » mémoire Ingénieur d'état en électronique option contrôle, Université Mouloud Mammeri de Tizi-Ouzou, 2003.

[12] Claude TOUZET « Les Réseaux de Neurone Artificiel Introduction au connexionnisme » HAL, 1992.

Sites web

[13] http://www.ics.uci.edu/~mlearn/MLRepository.html

[14] https://en.wikipedia.org/wiki/Moore–Penrose_pseudoinverse

Logiciel

- MATLAB version 10.7.