UNIVERSITE SIDI MOHAMED BEN ABDELLAH

FACULTE DES SCIENCES ET TECHNIQUES

DEPARTEMENT DES MATHEMATIQUES

Master Mathématique et Application au Calcul Scientifique (MACS)

MEMOIRE DE FIN D'ETUDES

Pour l'obtention du Diplôme de Master Sciences et Techniques (MST)

Réseaux de Neurones Artificiels et Apprentissage Bayésien, Applications à la Reconnaissance des Formes

Réalisé par: El houssaine HSSAYNI

Encadré par: Mohamed ETTAOUIL

Soutenu le 17 juin 2017

Devant le jury composé de:

-Pr. ETTAOUIL Mohamed	Encadrant
-Pr. EZZAKI Fatima	Président
-Pr. EL KHAOULANI Rachid	Examinateur
-Pr. EL AYADI Rachid	Examinateur

Année Universitaire 2016 / 2017

UNIVERSITÉ SIDI MOHAMED BEN ABDELLAH
FACULTÉ DES SCIENCES ET TECHNIQUES DE FES

DÉPARTEMENT DE MATHÉMATIQUES

MÉMOIRE DE FIN D'ÉTUDES

Pour l'obtention du diplôme de
MASTER MATHÉMATIQUES ET APPLICATIONS
AU CALCUL SCIENTIFIQUE

Présenté par :
El houssaine HSSAYNI
Sous la direction du :
Pr. **Mohamed ETTAOUIL**

Thème :

Réseaux de Neurones Artificiels et Apprentissage Bayésien, Application à la Reconnaissance des Formes

soutenu le 17 Juin 2017

Devant le Jury :

Pr. Fatima EZZAKI	Faculté des Sciences et Techniques Fès	Président
Pr. Mohamed ETTAOUIL	Faculté des Sciences et Techniques Fès	Encadrant
Pr. Rachid EL AYADI	Faculté des Sciences et Techniques Fès	Examinateur
Pr. Rachid EL KHAOULANI	Faculté des Sciences et Techniques Fès	Examinateur

DÉDICACES

A mes chers parents, en témoignage de ma fidèle affection pour l'amour que vous m'avez toujours réservé et de ma reconnaissance pour vos sacrifices et vos encouragements ;

A mes chers frères et mes chères soeurs ainsi que leurs petites familles ;

A mes amis de la promotion et chacun par son nom ;

A tous mes amis ;

A tous ceux et celles qui me sont chers.

Je dédie ce modeste travail.

REMERCIEMENT

Durant la période pendant laquelle s'est déroulée ce travail, beaucoup de gens m'ont apporté leur soutien tant moral que matériel. C'est pourquoi j'aimerais les remercier ici.

Je tiens à exprimer ma profonde gratitude à Monsieur *Mohamed ETTAOUIL*, professeur à l'université Sidi Mohamed Ben Abdellah, pour avoir m'encadré et dirigé mes recherches. Je je tiens à le remercier pour m'avoir soutenu et appuyé tout au long de ce travail. Leurs précieux conseils, leur exigence, et leurs commentaires m'ont permis d'améliorer grandement la qualité de mes travaux et de ce mémoire.

Je remercie vivement les membres du jury : Madame *Fatima EZZAKI*, professeur directrice de laboratoire LMCS à la Faculté des Sciences et Techniques de FES, Monsieur *Rachid EL KHAOULANI*, professeur à la Faculté des sciences et techniques de FES et Monsieur *Rachid EL AYADI*, professeur à la Faculté des sciences et techniques de FES, qui m'ont honoré en acceptant d'évaluer ce travail.

Mes sincères remerciements sont également adressés à tout le corps enseignant de la FSTF, plus particulièrement aux enseignants du Département de Mathématique et Informatique pour les efforts qu'ils ont déployés au profit de mon cursus de formation au Master MACS.

Je remercie également mes chers collègues, les doctorants : Mr. *Hassan RAMCHOUN*, Mr. *Zakariae EN-NAIMANI*, et Mr. *Nour-eddine JOUDAR*, qui n'ont épargné ni temps ni effort pour m'aider á concrétiser ce travail.

Un merci très spécial a Mademoiselle *Souad DAKIR*, ma collègue et binôme de recherche, pour ses aides précieuses, sa gentillesse, ses encouragements tout au long de la préparation de ce mémoire.

J'exprime ma gratitude et mon amitié à tous mes collègues et mes amis pour leurs disponibilités et leurs encouragements au cours de mon mémoire ainsi que pour leurs aides, leurs gentillesses et leurs précieux conseils.

TABLE DES MATIÈRES

TABLE DES FIGURES

LISTE DES ABRÉVIATIONS

ACP :	ANALYSE EN COMPOSANTES PRINCIPALES
ARD :	AUTOMATIC RELEVANCE DETERMINATION
EF :	EVIDENCE FRAMEWORK
FDP :	FONCTION DENSITE DE PROBABILITE
IID :	INDÉPENDANT ET IDENTIQUEMENT DISTRIBUÉ
LBP :	LOCAL BINARY PATTERNS
MCMC :	MARKOV CHAINE MONTE CARLO
PMC :	PERCEPTRON MULTICOUCHE
RNA :	RESEAUX DE NEURONES ARTIFICIELS
TCL :	TÉORÈME CENTRAL LIMIT

INTRODUCTION GÉNÉRALE

La reconnaissance de visage est un domaine de la vision par ordinateur consistant à reconnaitre automatiquement une personne à partir d'une image de son visage. C'est un domaine de recherche riche en publications en raison de ses nombreuses applications. Parmi celles-ci les systèmes de contrôle d'accès, de surveillance, d'interaction homme machine dans des applications multimédia de divertissement ou à finalités éducatives, de vérification de carte de crédit et bien d'autres.

Parmi les nombreux modèles proposés pour résoudre le problème de la reconnaissance des visages, on a les réseaux neuronaux, qui occupent depuis environ vingt ans une place notable dans des problèmes difficiles de classification et de reconnaissance des formes que l'on rencontre précisément en reconnaissance des visages.

Beaucoup d'efforts considérables ont été effectués par les chercheurs dans le but d'améliorer des réseaux de neurones. Parmi ces améliorations on peut citer l'intégration des méthodes bayésiennes dans l'apprentissage des réseaux de neurones, qui peuvent apporter plusieurs avantages, car il n'est pas nécessaire de limiter la taille du réseau pour éviter le sur-ajustement, et que le nombre de neurones cachés peut tendre vers l'infini, le seul facteur qui doit limiter la taille du réseau est la capacité des ordinateurs utilisés et le temps disponible pour effectuer les calculs nécessaires.

Ce mémoire est constitué de quatre chapitres principaux :

Le premier chapitre englobe, dans une première partie, une introduction aux réseaux de neurones artificiels, en définissant les différentes structures fondamentales, les différents types d'apprentissage, ainsi que les propriétés fondamentales des réseaux de neurones. Dans une seconde, nous allons voir le modèle le plus célèbre des réseaux de neurones, nommé perceptron multicouches. Nous allons définir sa topologie, et réaliser son apprentissage par la méthode de rétropropagation du gradient. Et enfin, dans une troisième partie, nous allons introduire le problème de sur-ajustement ainsi que quelques méthodes pour y éviter.

Le deuxième chapitre représente le cadre théorique du formalisme bayésien, dans lequel nous allons présenter, dans un premier temps, les fondements de la statistique bayésienne, et après nous allons donner les principales méthodes du calcul bayésien.

Le troisième chapitre est une combinaison entre les deux premiers, dans lequel nous allons intégrer les méthodes bayésiennes dans l'apprentissage des réseaux de neurones. Après avoir présenté des généralités sur l'approche bayésienne des réseaux de neurones, à savoir : le principe, les avantages et les inconvénients, nous allons détailler l'approche bayésienne des réseaux de neurones la plus connue à la littérature nommée procédure d'évidence et nous allons implémenter et tester la performance de cette procédure par le problème de classification classique des iris.

Le dernier chapitre est consacré à l'application des réseaux de neurones à la reconnaissance des visages. Dans une première partie nous allons présenter la reconnaissance faciale comme étant une technique biométrique ainsi que les principales difficultés de cette technique. Dans une seconde partie nous allons présenter quelque méthodes de description de visage et d'extraction des caractéristiques de texture. Enfin nous allons adapter les réseaux de neurones à l'apprentissage et la reconnaissance des visages.

CHAPITRE 1

RÉSEAUX DE NEURONES ARTIFICIELS

Introduction

Un réseau de neurones artificiels, ou réseau neuronal artificiel, est un ensemble d'algorithmes dont la conception est à l'origine très schématiquement inspirée du fonctionnement des neurones biologiques. Il permet de traiter des problèmes de différentes natures que les outils classiques ont du mal à résoudre.
Dans ce chapitre, après avoir énoncé les différentes caractéristiques d'un réseau de neurones, nous allons nous intéresser au cas particulier du perceptron multicouche.

1.1 Modèles d'un neurone

Comme les réseaux de neurones mis au point par les informaticiens sont largement inspirés de ce que la biologie nous apprend sur ceux que l'on trouve chez les êtres vivants, il convient d'abord de décrire brièvement le modèle biologique.

1.1.1 Neurone Biologique

Chez les êtres vivants, les neurones sont les cellules nerveuses. Un neurone est doté de ramification que l'on nomme les dendrites par lesquelles transite l'information (sous forme des courants électriques) venue de l'extérieur vers le corps cellulaire. le neurone traite cette information et renvoie le résultat au travers de son axone. Ce signal émis par le neurone peut ensuite être transmis, au travers d'un synapse, à un autre neurone, ou encore à un muscle ou à une glande (Figure 1.1).

1.1.2 Neurone formel

Le modèle mathématique le plus souvent utilisé est celui de McCulloch et Pitts représenté sur la Figure 1.2. On y reconnait les divers éléments associés à un neurone biologique vivant :

- des synapses que nous appellerons par la suite connexions sur lesquelles arrivent les entrées x_i. Elles sont caractérisées par un poids : chaque signal x_i passant sur une connexion sera multiplié par le coefficient de pondération w_i

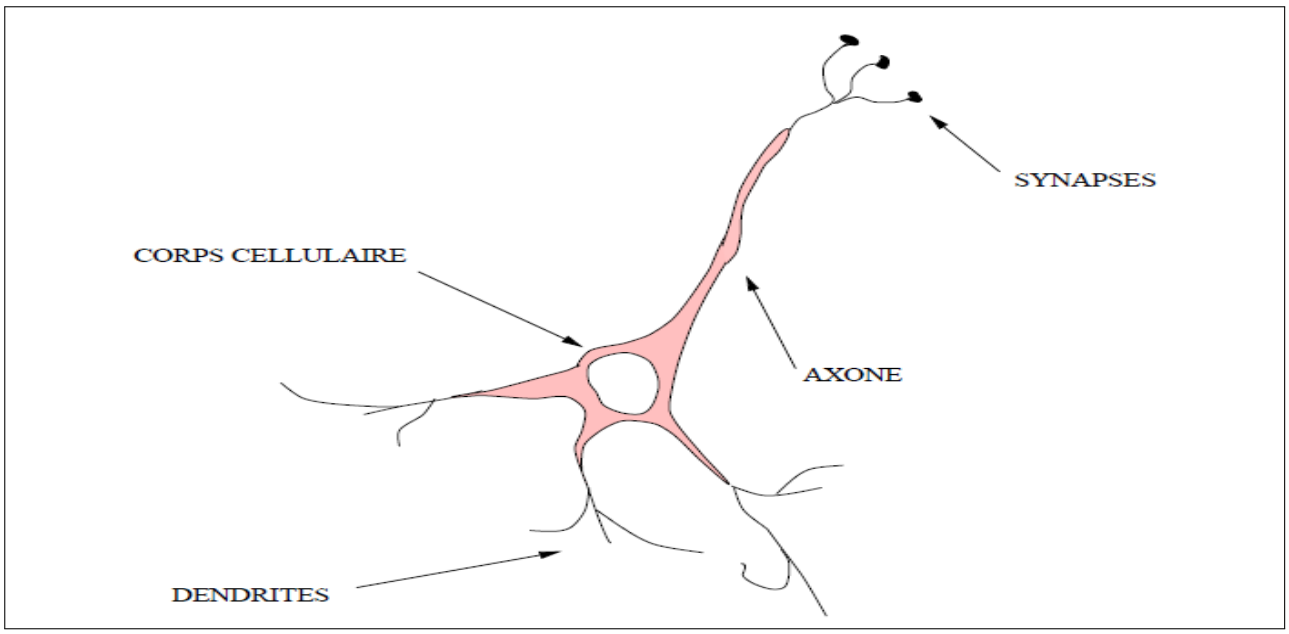

Figure 1.1 – Un neurone biologique et ses principales composantes

- un mécanisme de sommation permettant de faire une combinaison linéaire des entrées pondérées précédentes
- un biais b qui représente le seuil au dessus duquel le neurone sera excité. Il sert à centrer la zone d'action du neurone
- une sortie y qui peut être utilisée comme entrée pour d'autres neurones
- une fonction d'activation φ qui sert à limiter sa sortie et permet d'agir sur la forme de la zone sur laquelle agit le neurone (ainsi, si l'on considère qu'un neurone agit sur un certain espace, le biais représente l'endroit où le neurone se trouve et la fonction d'activation indique comment il intervient sur la zone de l'espace qui se trouve a proximité). En général, on utilisera une fonction de type sigmoïde comme tanh, c'est cette fonction qui introduit la non-linéarité dans le neurone.

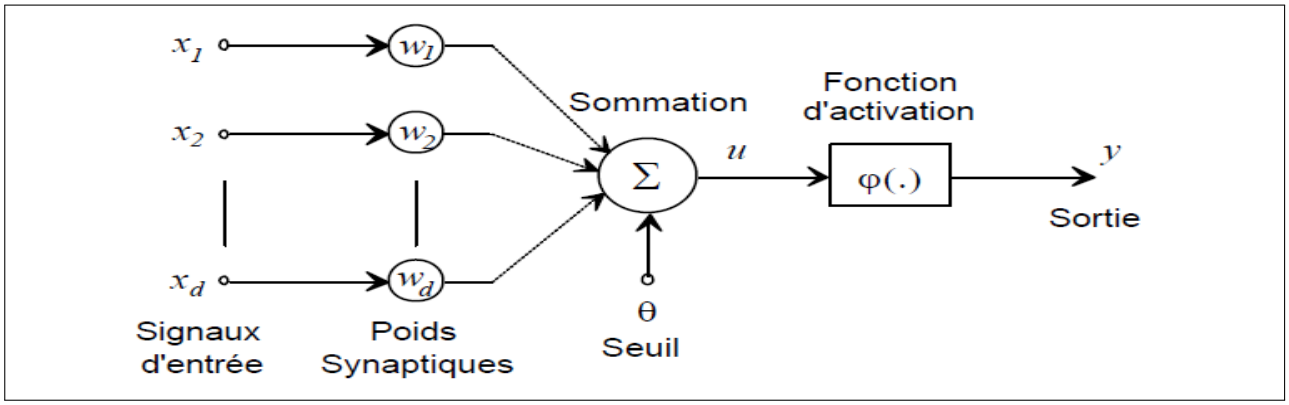

Figure 1.2 – Neurone Formel

1.2 Structure des réseaux de neurones

Les connexions entre les neurones qui composent le réseau décrivent la topologie du modèle. Les neurones sont organisés en couches successives qui sont reliées entre elles de différentes façons ce qui donne les différents types de réseaux artificiels qui suivent :

1.2.1 Réseaux de neurones feedforward ou non bouclés

On dit d'un réseau qu'il est feedforward, ou encore à sens unique, lorsqu'il ne contient pas de boucle interne, c'est à dire lorsque l'information qui le traverse ne circule que de l'entrée vers la sortie. Les neurones sont organisés en couches successives, les neurones d'une couche étant reliés à ceux de la suivante. Parmi ce type de réseaux, il existe les réseaux à deux couches. Ils sont composés d'un ensemble de neurones répartis sur deux couches distinctes appelées couche d'entrée et de sortie, les neurones de la première étant connectés à ceux de la seconde. Un exemple de ce type de réseau feedforward est celui de la figure 1.3.

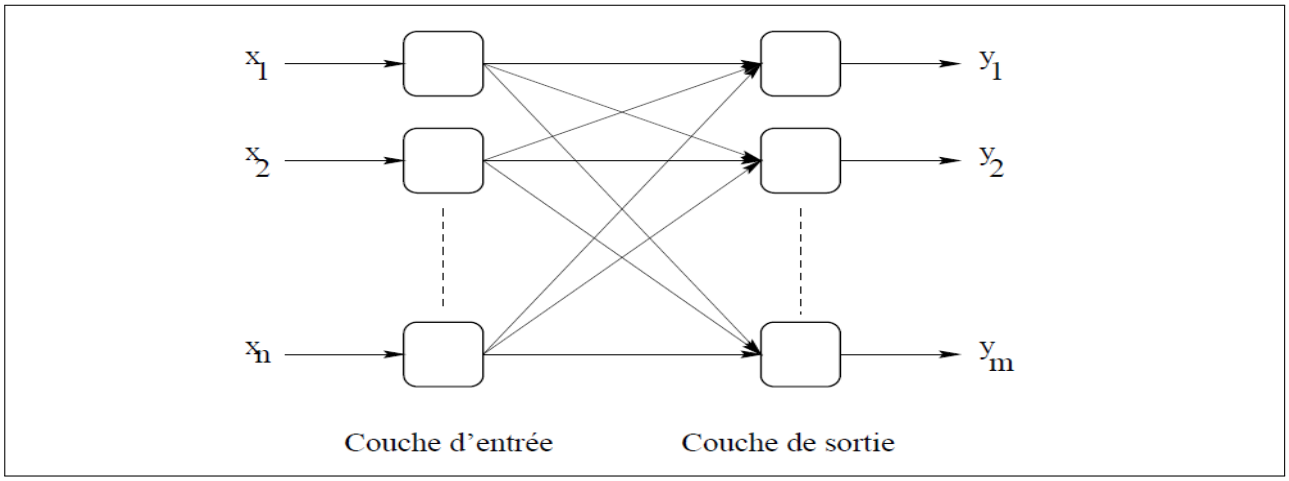

Figure 1.3 – Réseau de neurones feedforward à deux couches

Si l'on intercale entre la sortie et l'entrée une ou plusieurs couches (qui seront alors appelées couches cachées) on obtient un réseau dit réseau feedforward multi-couches.
L'intérêt de rajouter au moins une couche cachée est d'augmenter le nombre de connexions, ce qui accroît la capacité d'un réseau à extraire l'information des données fournies en entrée.

1.2.2 Réseaux de neurones récurrents

On parle de réseau de neurones récurrent lorsqu'il existe une boucle au moins dans l'ensemble des connexions. Il y a alors au minimum un feedback d'une couche sur la précédente. La figure 1.4 en donne un exemple très simple : un neurone de la couche de sortie renvoie sa sortie sur la couche d'entrée avec un délai. Cela permet d'utiliser une des sorties à l'instant t comme entrée à l'instant $t + 1$.

1.3 Propriétés des réseaux de neurones

Les réseaux de neurones à couches, présentés au paragraphe précédent, ont la propriété générale d'être des approximateurs universels parcimonieux. Il s'agit en fait de deux propriétés

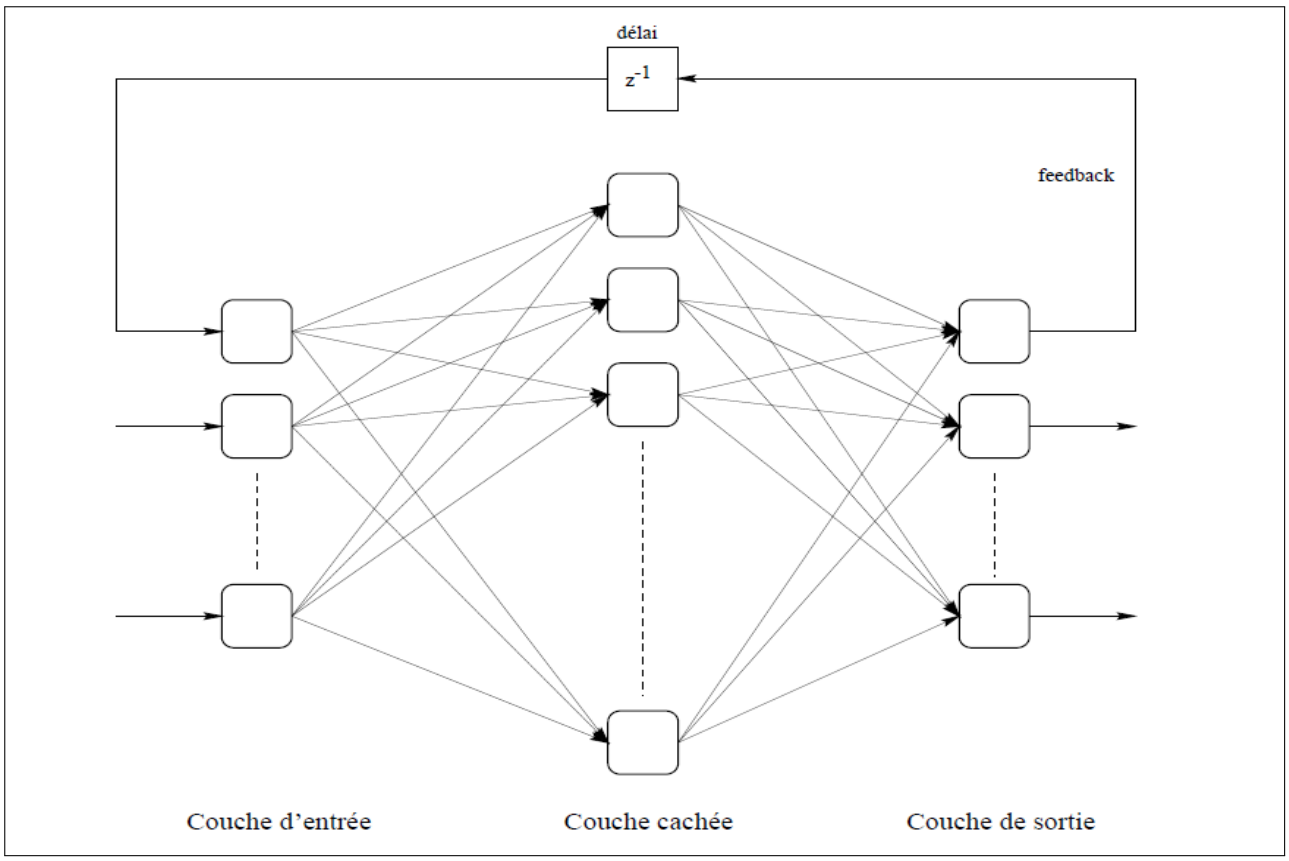

Figure 1.4 – Exemple de réseau récurrent simple

distinctes détaillées ci-dessous.

1.3.1 Propriété d'approximation universelle

La propriété d'approximation universelle a été démontrée par [Cybenko, 1989] et [Funahashi,1989] et peut s'énoncer de la façon suivante :

Propriété 1.1
Toute fonction bornée suffisamment régulière peut être approchée uniformément, avec une précision arbitraire, dans un domaine fini de l'espace de ses variables, par un réseau de neurones comportant une couche de neurones cachés en nombre fini, possédant tous la même fonction d'activation, et un neurone de sortie linéaire.

1.3.2 Propriété de parcimonie

Lorsque l'on cherche à modéliser un processus à partir des données, on s'efforce toujours d'obtenir les résultats les plus satisfaisants possibles avec un nombre minimum de paramètres ajustables. Dans cette optique, [Hornik et al., 1994] ont montré que :

Propriété 1.2
Si le résultat de l'approximation (c'est-à-dire la sortie du réseau de neurones) est une fonction non linéaire des paramètres ajustables, elle est plus parcimonieuse que si elle est une fonction linéaire de ces paramètres. De plus, pour des réseaux de neurones à fonction d'activation sigmoïdale, l'erreur commise dans l'approximation varie comme l'inverse du nombre de neurones cachés, et elle est indépendante du nombre de variables de la fonction à approcher. Par conséquent, pour une précision donnée, donc pour un nombre de neurones cachés donné, le nombre de paramètres du réseau est proportionnel au nombre de variables de la fonction à approcher.

Ce résultat s'applique aux réseaux de neurones à fonction d'activation sigmoïdale puisque la sortie de ces neurones n'est pas linéaire par rapport aux poids synaptiques. Cette propriété montre l'intérêt des réseaux de neurones par rapport à d'autres approximateurs comme les polynômes dont la sortie est une fonction linéaire des paramètres ajustables : pour un même nombre d'entrées, le nombre de paramètres ajustables à déterminé est plus faible pour un réseau de neurones que pour un polynôme. Cette propriété devient d'autant plus intéressante dans le cas du filtrage des textes car le nombre d'entrées est typiquement de l'ordre de plusieurs dizaines.

1.4 Apprentissage des réseaux de neurones

L'apprentissage est une phase du développement d'un réseau de neurones durant laquelle le comportement du réseau est modifié jusqu'à l'obtention du comportement désiré.
L'apprentissage neuronal fait appel à des exemples de comportement.
Il existe essentiellement deux sortes d'apprentissages (selon les exemples sont étiquetés ou non) :

- L'apprentissage supervisé
- L'apprentissage non supervisé

1.4.1 Apprentissage supervisé

L'apprentissage dit supervisé est caractérisé par la présence d'un ensemble de N exemples étiquetés $\{(x_1, d1), (x_2, d2), ..., (x_N, dN)\}$
où x_i désigne un stimulus (entrée) et d_i la cible pour ce stimulus, c'est-à-dire la sortie désirée du réseau. Chaque couple (x_i, di) correspond donc à un cas d'espèce de ce que le réseau devrait produire (la cible) pour un stimulus donné. Pour cette raison, l'apprentissage supervisé est aussi qualifié d'apprentissage par des exemples.
On peut distinguer deux grands types d'apprentissage supervisé : la classification et la régression.

1.4.1.1 Classification

Lorqu'on fait de la classification, l'entrée est l'instance d'une classe et l'étiquette est la classe correspondante. En reconnaissance de caractères, par exemple, l'entrée serait une suite de pixels représentant une lettre et la classe serait la lettre représentée (ou son index).
La classification consiste donc à apprendre une fonction f_{class} de $\mathbb{R}^d$ dans $\mathbb{N}$ qui associe à un vecteur sa classe. Dans certains cas, on pourra vouloir que la fonction f_{class} soit à valeurs dans

$[0,1]^k$ telle que chaque élément du vecteur de sortie représente la probabilité d'appartenance à une classe (la somme des éléments sera donc 1).

$$\begin{aligned} f_{class} : \mathbb{R}^d &\longrightarrow \mathbb{N} \\ entrée &\longmapsto f_{class}(entrée) = classe \end{aligned}$$

1.4.1.2 Régression

Dans le cas de la régression, l'entrée n'est pas associée à une classe mais à une ou plusieurs quantités continues. Ainsi, l'entrée pourrait être les caractéristiques d'une personne (son âge, son sexe, son niveau d'études) et l'étiquette son revenu.
La régression consiste donc à apprendre une fonction f_{regr} de $\mathbb{R}^d$ dans $\mathbb{R}^k$ qui associe à un vecteur sa valeur associée.

$$\begin{aligned} f_{regr} : \mathbb{R}^d &\longrightarrow \mathbb{R}^k \\ entrée &\longmapsto f_{regr}(entrée) = valeur \end{aligned}$$

1.4.2 Apprentissage non supervisé

L'apprentissage non supervisé des réseaux de neurones consiste, comme dans le cas d'apprentissage supervisé, à modifier les poids des connections des neurones. Dans ce cas, les exemples de base d'apprentissage sont des données seules : il n'est pas possible de modifier les poids du réseau en fonction d'une erreur sur les réponses souhaitées, puisqu'aucune réponse n'est connue a priori.
La figure 1.5 illustre les deux types d'apprentissage : supervisé et non supervisé.

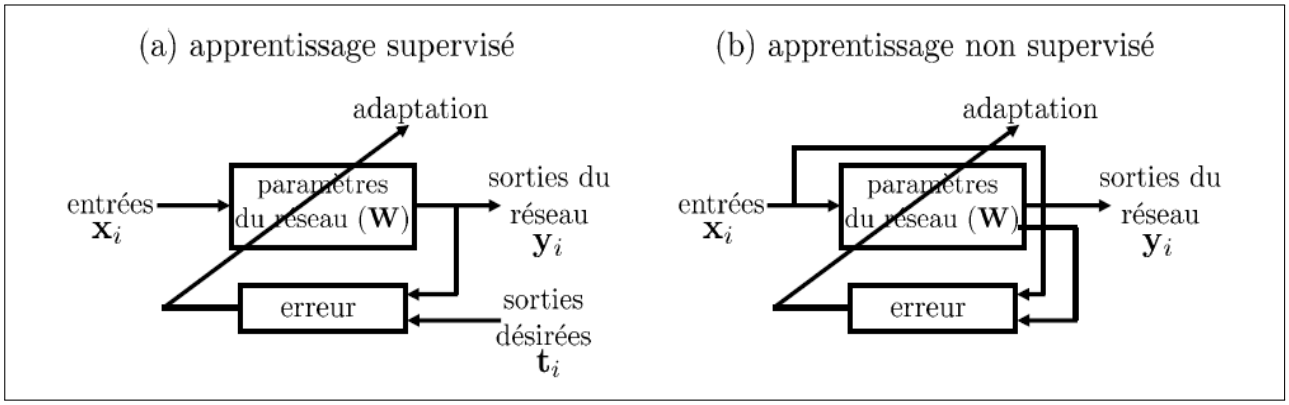

Figure 1.5 – Synoptique de l'apprentissage supervisé et non supervisé d'un réseau de neurones

1.5 Perceptron multicouche et son algorithme d'apprentissage

Le perceptron a été inventé en 1957 par Frank Rosenblatt au laboratoire d'aéronautique de l'université Cornell. C'est un modèle inspiré des théories cognitives de Friedrich Hayek et de Donald Hebb.

1.5.1 Perceptron multicouche (PMC)

Le perceptron multicouche est un type des réseaux de neurones les plus utilisés pour des problèmes d'approximation, de classification et de prédiction. Il est habituellement constitué d'au moins trois couches totalement connectés.

- La couche d'entrée :
  ne joue généralement que le rôle d'interface avec l'extérieur, c'est à dire qu'elle n'effectue pas de traitement sur l'information.

- Une ou plusieurs couches cachées :
  appelées aussi couches de traitement ou couches intermédiaires, leurs intérêt est d'augmenter le nombre de connexions, ce qui accroît la capacité d'un réseau à extraire l'information des données fournies en entrée.

- La couche de sortie :
  traite également l'information qu'elle reçoit de la couche précédente mais elle ne communique pas son résultat aux autres neurones. L'information fournie par chacune de ses unités constitue le vecteur de sortie du réseau. Celui-ci est donc le résultat du traitement d'un vecteur d'entrée par l'ensemble du réseau.

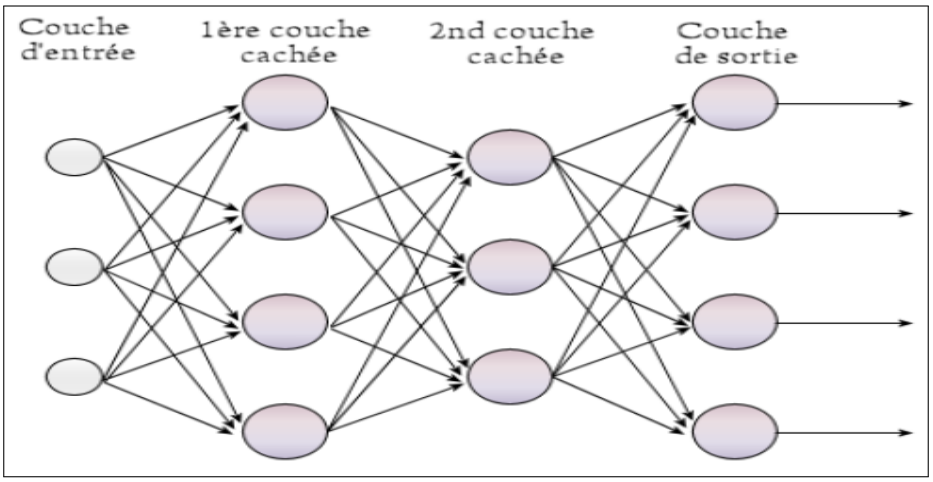

Figure 1.6 – Un perceptron multicouche avec deux couches cachées

1.5.2 Algorithme de Rétro-propagation du gradient

Le problème de l'apprentissage dans les perceptrons multi-couches est de connaitre la contribution de chaque poids dans l'erreur globale du réseau. L'algorithme de rétro-propagation de l'erreur permet de faire cela :

1. Propagation de l'entrée jusqu'à la sortie
2. Calcul de l'erreur en sortie
3. Rétro-propagation de l'erreur jusqu'aux entrées

Nous allons maintenant déterminer les équations de mise à jour des pondérations :

- Pour la couche de sortie :
  Considérons un neurone j de la couche de sortie. Lorsque l'exemple numéro n lui est présenté, il produit la sortie $y_j(n)$ alors que c'est la valeur $d_j(n)$ qui est attendue. Notons

$$e_j(n) = d_j(n) - y_j(n) \tag{1.1}$$

  l'erreur associée à ce neurone. L'erreur globale du réseau sur la pattern n est

$$\varepsilon(n) = \frac{1}{2}\sum_{j=1}^{m} e_j^2(n) \tag{1.2}$$

  avec m est le nombre de neurones dans la couche de sortie.
  Le but de l'apprentissage est de minimiser l'erreur moyenne correspondante aux N exemples d'apprentissage, c'est à dire minimiser

$$\varepsilon_{moy} = \frac{1}{N}\sum_{n=1}^{N} \varepsilon(n) \tag{1.3}$$

  Soit

$$v_j(n) = \Big(\sum_i w_{ij}(n) y_i(n)\Big) - b_j(n) \tag{1.4}$$

  l'activité interne d'un neurone dont la fonction d'activation est f_j. Sa sortie s'écrit alors sous forme de

$$y_j(n) = f_j(v_j(n)) \tag{1.5}$$

  Le mécanisme de rétro-propagation est basé sur une correction $\Delta W_{ji}(n)$ et $\Delta b_j(n)$. En ce qui concerne les poids, on utilisera la règle dite du delta (delta rule), c'est à dire que la modification sera proportionnelle au gradient suivant :

$$\frac{\partial \varepsilon(n)}{\partial w_{ji}(n)} = \frac{\partial \varepsilon(n)}{\partial e_j(n)}\frac{\partial e_j(n)}{\partial y_j(n)}\frac{\partial y_j(n)}{\partial v_j(n)}\frac{\partial v_j(n)}{\partial w_{ji}(n)} \tag{1.6}$$

  qui détermine la direction dans laquelle on recherche les valeurs de $w_{ij}(n)$. Nous appellerons taux d'apprentissage le facteur de proportionnalité et nous le noterons η par la suite. D'après l'équation 2.2, on peut écrire :

$$\frac{\partial \varepsilon(n)}{\partial e_j(n)} = e_j(n) \tag{1.7}$$

  de plus avec 2.1

$$\frac{\partial e_j(n)}{\partial y_j(n)} = f_j^{'}(v_j(n)) \tag{1.8}$$

  et finalement

$$\frac{\partial v_j(n)}{\partial w_{ji}(n)} = y_i(n) \tag{1.9}$$

  à partir de 2.4. Soit $\delta_j(n)$ le gradient local défini par

$$\delta_j(n) = -\frac{\partial \varepsilon(n)}{\partial e_j(n)}\frac{\partial e_j(n)}{\partial y_j(n)}\frac{\partial y_j(n)}{\partial v_j(n)} \tag{1.10}$$

d'après ce qui précède ceci est aussi égal à

$$\delta_j(n) = e_j(n) f_j^{'}(v_j(n)) \tag{1.11}$$

La correction appliquée au poids $W_{ji}(n)$ sera donc la suivante (règle du delta) :

$$\Delta W_{ji}(n) = -\eta \frac{\partial \varepsilon(n)}{\partial w_{ji}(n)} \tag{1.12}$$

soit encore

$$\Delta W_{ji}(n) = \eta \delta_j(n) y_i(n) \tag{1.13}$$

De même, la correction apportée au biais s'écrira sous la forme

$$\Delta b_j(n) = \eta \frac{\partial \varepsilon(n)}{\partial b_j(n)} \tag{1.14}$$

en suivant un raisonnement identique à ce qui précède on obtient

$$\frac{\partial \varepsilon(n)}{\partial b_j(n)} = \frac{\partial \varepsilon(n)}{\partial e_j(n)} \frac{\partial e_j(n)}{\partial y_j(n)} \frac{\partial y_j(n)}{\partial v_j(n)} \frac{\partial v_j(n)}{\partial b_j(n)} \tag{1.15}$$

$$= -\delta_j(n) \frac{\partial v_j(n)}{\partial b_j(n)} \tag{1.16}$$

$$= \delta_j(n) \tag{1.17}$$

D'où

$$\Delta b_j(n) = -\eta \delta_j(n) \tag{1.18}$$

Ces deux équations de mise à jour sont valables pour la couche de sortie d'un réseau de neurones car nous avons utilisé l'erreur de sortie $e_j(n) = d_j(n) - y_j(n)$

- Pour les couches cachées :
  En ce qui concerne les autres couches, on ne peut utiliser cette formulation pour l'erreur, voyons alors quelles vont être les nouvelles équations de mise à jour. Pour cela considérons un neurone j sur la dernière couche cachée, nous pouvons définir le gradient local par

$$\delta_j(n) = \frac{\partial \varepsilon(n)}{\partial y_j(n)} \frac{\partial y_j(n)}{\partial v_j(n)} \tag{1.19}$$

$$= \frac{\partial \varepsilon(n)}{\partial y_j(n)} f_j^{'}(v_j(n)) \tag{1.20}$$

  en utilisant une formule analogue à 2.11. D'après l'équation 2.2 on peut écrire

$$\frac{\partial \varepsilon(n)}{\partial y_j(n)} = \sum_k e_k(n) \frac{\partial e_k(n)}{\partial y_j(n)} \tag{1.21}$$

$$= \sum_k e_k(n) \frac{\partial e_k(n)}{\partial v_k(n)} \frac{\partial v_k(n)}{\partial y_j(n)} \tag{1.22}$$

  Or le neurone k est sur la couche de sortie, d'où

$$\frac{\partial e_k(n)}{\partial v_k(n)} = \frac{\partial (d_k(n) - y_k(n)}{\partial v_k(n)} \tag{1.23}$$

$$= \frac{\partial (d_k(n) - f_k(v_k(n))}{\partial v_k(n)} \tag{1.24}$$

$$= f_k^{'}(v_k(n)) \tag{1.25}$$

de plus, d'après 2.4

$$\frac{\partial e_k(n)}{\partial y_j(n)} = w_{kj}(n) \tag{1.26}$$

nous obtenons alors

$$\frac{\partial \varepsilon(n)}{\partial y_j(n)} = -\sum_k e_k(n) f_k^{'}(v_k(n)) w_{kj}(n) \tag{1.27}$$

$$= -\sum_k \delta_k(n) w_{kj}(n) \tag{1.28}$$

Le gradient local pour un neurone de la dernière couche cachée est alors donné par :

$$\delta_j(n) = \Big(\sum_k \delta_k(n) w_{kj}(n)\Big) f_k^{'}(v_j(n)) \tag{1.29}$$

Ceci pouvant se généraliser aux autres couches internes, nous avons ainsi obtenu les règles de mise à jour des poids et des biais avec la méthode de rétro-propagation du gradient. Ainsi, les poids et les biais doivent être actualisés en utilisant :

$$w_{ji} \longleftarrow w_{ji} + \eta \delta_j(n) y_j(n)$$

$$b_j \longleftarrow b_j - \eta \delta_j(n)$$

avec

$$\delta_j(n) = e_j(n) f_j^{'}(v_j(n))$$

et

$$e_j(n) = \begin{cases} d_j(n) - y_j(n) & \text{si } j \text{ est un neurone de sortie} \\ \sum_k \delta_k(n) w_{kj}(n) & \text{si } j \text{ est un neurone interne} \end{cases}$$

Ainsi, l'erreur de sortie se trouve propagée vers l'entrée (c'est à dire dans le sens inverse de celui qui est utilisé pour propager un signal au travers d'un réseau) afin de corriger de couche en couche les valeurs des poids et des biais.

La mise en oeuvre de l'algorithme

Algorithme 1 (Back-propagation)

1. *Initialisation :*
   Initialiser tous les poids et les biais du réseau de neurones.
2. *Présentation d'un exemple en entrée :*
   Présenter un exemple en entrée $x(n) = (x_1(n), x_2(n), ..., x_p(n))$ *ainsi que la sortie désirée correspondante* $d(n) = (d_1(n), d_2(n), ..., d_m(n))$.
3. *Calculer les sorties :*
   calcules successivement les sorties des différentes couches pour l'entrée x.
4. *La mise à jour des poids et des biais :*
   modifier récursivement les poids et les biais du réseau suivant les règles détaillées avant :

$$w_{ji} \longleftarrow w_{ji} + \eta \delta_j(n) y_j(n)$$

$$b_j \longleftarrow b_j - \eta \delta_j(n)$$

avec

$$\delta_j(n) = e_j(n) f_j^{'}(v_j(n))$$

et

$$e_j(n) = \begin{cases} d_j(n) - y_j(n) & \text{si } j \text{ est un neurone de sortie} \\ \sum_k \delta_k(n) w_{kj}(n) & \text{si } j \text{ est un neurone interne} \end{cases}$$

5. *Critère d'arrêt :*
   Répéter les étapes 2, 3 et 4 jusqu'à un nombre maximum d'itérations ou jusqu'à ce que l'erreur quadratique moyenne soit inférieure à un certain seuil.

1.6 Problème de surajustement et motivation pour l'approche Bayésienne

1.6.1 Définition de surajustement

Si l'on considère un ensemble d'apprentissage et une fonction de coût quadratique, en vertu de la propriété d'approximation universelle exposée au paragraphe 1.3.1, il est toujours possible d'obtenir une fonction de coût aussi petite que l'on veut sur l'ensemble d'apprentissage, à condition de mettre suffisamment de neurones cachés. Cependant, le but de l'apprentissage n'est pas d'apprendre exactement la base d'apprentissage, mais le modèle sous-jacent qui a servi à engendrer les données. Or, si la fonction apprise par le réseau de neurones est ajustée trop finement aux données, elle apprend les particularités de la base d'apprentissage au détriment du modèle sous-jacent : le réseau de neurones est surajusté.

1.6.2 Méthodes pour limiter le surajustement

On distingue deux familles de méthodes pour prévenir le surajustement : les méthodes passives et les méthodes actives. Les philosophies de ces deux familles de méthodes sont différentes.

- Les méthodes passives essayent de détecter le surajustement à posteriori pour supprimer les mauvais modèles. Parmi les méthodes les plus classiques figurent l'utilisation d'une base de validation pendant l'apprentissage, et les mesures de critère d'information.

- Les méthodes actives interviennent pendant la phase d'apprentissage pour empêcher le modèle de faire du surajustement. Les méthodes de régularisation comme l'arrêt prématuré ou la pénalisation entrent dans ce cadre.

1.6.2.1 Méthodes passives

Les méthodes passives, issues de la description du problème de surajustement en termes de biais-variance ne propose comme solution qu'une limitation de la complexité du modèle par l'intermédiaire d'une limitation du nombre de neurones cachés.

1.6.2.2 Méthodes actives : les méthodes de régularisation

Les méthodes de régularisation, par opposition, peuvent être qualifiées d'actives, car elles ne cherchent pas à limiter la complexité du réseau, mais elles contrôlent la valeur des poids pendant l'apprentissage. Il devient possible d'utiliser des modèles avec un nombre élevé de poids et donc un modèle complexe, même si le nombre d'exemples d'apprentissage est faible.
Bartlett (1997) a montré que la valeur des poids était plus importante que leur nombre afin d'obtenir de modèles qui ne sont pas surajustés. Il montre, que si un grand réseau est utilisé et que l'algorithme d'apprentissage trouve une erreur quadratique moyenne faible avec des poids de valeurs absolues faibles, alors les performances en généralisation dépendent de la taille des poids plutôt que de leur nombre.
Plusieurs méthodes de régularisation existent dans la littérature, a savoir :

a) L'arrêt prématuré (early stopping) :
Comme l'apprentissage consiste à minimiser, grâce à un algorithme itératif, une fonction de coût calculée sur la base d'apprentissage. La méthode de l'arrêt prématuré (early stopping) consiste à arrêter les itérations avant la convergence de l'algorithme. Si la convergence n'est pas menée à son terme, le modèle ne s'ajuste pas trop finement aux données d'apprentissage : le surajustement est limité.

b) Weight Decay :
Lorsque les poids du réseau sont grands en valeur absolue, les sigmoïdes des neurones cachés sont saturées, si bien que les fonctions modélisées peuvent avoir des variations brusques. Pour obtenir des fonctions régulières, il faut travailler avec la partie linéaire des sigmoïdes, ce qui implique d'avoir des poids dont la valeur absolue est faible.
La méthode de régularisation du weight decay limite la valeur absolue des poids en faisant l'apprentissage en minimisant l'erreur régularisée suivante :

$$E^{'} = E + \frac{\alpha}{2}\sum_{i=1}^{p} w_i^2 \tag{1.30}$$

où p est le nombre de poids que comporte le réseau.

1.6.3 Motivation

L'identification de la valeur optimale de l'hyperparamètre α doit être faite par validation croisée. Or, elle a un coût de calcul important. Pour contourner ce problème, MacKay propose d'utiliser une approche probabiliste appelée régularisation bayésienne durant la procédure d'apprentissage pour contrôler automatiquement la complexité du RNA.

Conclusion

Les réseaux de neurones s'avèrent un outil très puissant que ça soit pour des problèmes de classification, de régression et de prédiction dans divers domaines d'applications grâce à leurs propriétés universelles.
Dans ce chapitre nous avons présenté les concepts généraux d'un réseau de neurones, et en particulier le modèle perceptron multicouches ainsi que son algorithme d'apprentissage. Nous avons également énoncé le problème de sur-ajustement ainsi que quelques méthodes pour y éviter.

CHAPITRE 2

CALCUL BAYÉSIEN : DÉFINITIONS ET PRINCIPES

Introduction

La statistique bayésienne est une approche statistique fondée sur l'inférence bayésienne. On utilise le terme de bayésienne pour la différencier de la statistique classique qui ne sait traiter que les grands échantillons. La statistique bayésienne est surtout utilisée lorsque l'on n'a que de petits échantillons, typiquement quand chaque observation est elle-même très coûteuse. Contrairement à la statistique classique, elle n'exige pas au départ qu'on se fixe une hypothèse précise à confirmer ou infirmer, ce qui la rend utile en data mining.
Dans ce chapitre nous présentons les fondements de la statistique bayésienne ainsi que quelques méthodes de calcul bayésien.

2.1 Fondements de la statistique bayésienne

2.1.1 Théorème de Bayes

Le théorème de Bayes est un résultat de base en théorie des probabilités, issu des travaux du révérend Thomas Bayes et retrouvé ensuite indépendamment par Laplace. Le théorème de Bayes est utilisé dans l'inférence statistique pour mettre à jour ou actualiser les estimations d'une probabilité ou d'un paramètre quelconque, à partir des observations et des lois de probabilité de ces observations.

Théorème 2.1
soit deux événements A et B, avec $P(B) \neq 0$. Si on connaît a priori la probabilité que l'événement B se produise et la probabilité que l'événement A se produise sachant que B s'est produit, alors la probabilité que B se produise sachant que A s'est produit est donnée par :

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)} \tag{2.1}$$

Dans le cadre de l'apprentissage machine,nous sommes intéressés par la probabilité d'avoir un ensemble de paramètres particulier θ étant donné que nous avons à notre disposition l'ensemble d'apprentissage D. Alors le théorème de Bayes s'écrit :

$$P(\theta|D) = \frac{P(D|\theta)P(\theta)}{P(D)} \tag{2.2}$$

2.1.2 Définition d'un modèle bayésien

Pour introduire la notion du modèle bayésien, on va tout d'abord identifier le modèle statistique paramétrique.

Définition 2.1
Un modèle paramétrique statistique consiste en l'observation d'une variable aléatoire x distribuée selon $f(x|\theta)$, où seulement le paramètre θ est inconnu et appartient à un espace de dimension finie.

Définition 2.2
Un modèle statistique bayésien est constitué d'un modèle statistique paramétrique, $f(x|\theta)$, et d'une distribution a priori pour les paramètres, $\pi(\theta)$.

2.1.3 Distributions a priori et a posteriori

Supposons désormais que, en plus d'une distribution d'échantillonnage,$f(x|\theta)$, une distribution a priori sur θ, $\pi(\theta)$, soit disponible, c'est-à-dire que nous disposions d'un modèle complètement bayésien. Une fois ces deux distributions sont données, nous pouvons en construire plusieurs autres, à savoir :

- la distribution jointe de (θ, x) :

$$\varphi(\theta, x) = f(x|\theta)P(\theta) \tag{2.3}$$

- la distribution marginale de x :

$$m(x) = \int \varphi(\theta, x)d\theta \tag{2.4}$$

$$= \int f(x|\theta)\pi(\theta)d\theta \tag{2.5}$$

- la distribution a posteriori de θ, obtenue par la formule de Bayes :

$$\pi(\theta|x) = \frac{f(x|\theta)\pi(\theta)}{\int f(x|\theta)\pi(\theta)d\theta} \tag{2.6}$$

$$= \frac{f(x|\theta)\pi(\theta)}{m(x)} \tag{2.7}$$

2.1.4 Paradigme bayésien

Étant donné un modèle paramétrique d'observation $x \sim f(x|\theta)$, où $\theta \in \Theta$, un espace de dimension finie, l'analyse statistique bayésienne vise à exploiter le plus efficacement possible l'information apportée par x sur le paramètre θ, pour ensuite construire des procédures d'inférence sur θ. Bien que x ne soit qu'une réalisation [aléatoire] d'une loi gouvernée par θ, elle apporte une actualisation aux informations préalablement recueillies par l'expérimentateur. Pour des raisons diverses dont certaines apparaitront dans le prochain paragraphe, l'information fournie par l'observation x est contenue dans la densité $f(x|\theta)$, que l'on représente classiquement sous la forme inversée de vraisemblance,

$$\ell(\theta|x) = f(x|\theta) \tag{2.8}$$

pour traduire qu'il s'agit d'une fonction de θ, qui est inconnu, dépendant de la valeur observée x. L'inversion des rôles de x et de θ par rapport à la modélisation probabiliste reflète le but premier de la statistique qui est de reconstruire [avec un certain degré de précision] le paramètre θ au vu de la réalisation aléatoire x. C'est donc pourquoi elle est naturellement liée au théorème de Bayes qui formalise l'inversion des conditionnements dans les probabilités :
Si A et E sont des événements tels que $P(E) \neq 0$, alors $P(A|E)$ et $P(E|A)$ sont reliés par :

$$P(A|E) = \frac{P(E|A)P(A)}{P(E)} \tag{2.9}$$

Une version continue de ce résultat permet d'inverser les densités conditionnelles, à savoir,

$$g(y|x) = \frac{f(x|y)g(y)}{\int f(x|y)g(y)dy} \tag{2.10}$$

Le lien entre ces propriétés probabilistes et l'inférence bayésienne est que, dans le paradigme bayésien, le paramètre inconnu θ n'est plus considéré comme inconnu et déterministe, mais comme une variable aléatoire.
On considère ainsi que l'incertitude sur le paramètre θ d'un modèle peut être décrite par une distribution de probabilité π sur Θ, appelée distribution a priori [par opposition à la distribution a posteriori qui inclut l'information contenue dans l'observation x], ce qui revient à supposer que θ est distribué suivant $\pi(\theta)$, $\theta \sim \pi(\theta)$, "avant" que x ne soit généré suivant $f(x|\theta)$, le conditionnement implicite dans cette notation prenant alors tout son sens. Sans vouloir nous engager dans un débat philosophique sur la nature du hasard, notons que le rôle central de la distribution a priori dans l'analyse statistique bayésienne ne réside pas dans le fait que le paramètre d'intérêt θ puisse (ou ne puisse pas) être perçu comme étant distribué selon π, ou même comme étant

une variable aléatoire, mais plutôt dans la démonstration que l'utilisation d'une distribution a priori et de l'appareillage probabiliste qui l'accompagne est la manière la plus efficace [au sens de nombreux critères] de résumer l'information disponible (ou le manque d'information) sur ce paramètre ainsi que l'incertitude résiduelle.
Un point plus technique est que le seul moyen de construire une approche mathématiquement justifiée opérant conditionnellement aux observations, tout en restant dans un schéma probabiliste, est d'introduire une distribution correspondante pour les paramètres. Des arguments de nature différente sur l'optimalité de cet outil sont aussi fournis dans [Bernardo et Smith, 1994] et [Robert, 2007].

2.1.5 Estimation ponctuelle

Par application directe du théorème de Bayes, si θ est supposé être une variable aléatoire de densité (a priori ou marginale) $\pi(\theta)$ et si $f(x|\theta)$ est interprétée comme loi conditionnelle de x conditionnellement à θ, la loi de θ conditionnelle a x, $\pi(\theta|x)$, appelée distribution a posteriori, est définie par :

$$\pi(\theta|x) = \frac{f(x|\theta)\pi(\theta)}{\int f(x|\theta)\pi(\theta)d\theta} \tag{2.11}$$

Cette densité est centrale pour l'inférence bayésienne en ce qu'elle suffit à déterminer les procédures de décision et, par extension, à conduire toute inférence liée à θ. Celle-ci joue donc le rôle d'un résumé exhaustif de l'information apportée par les données, ce qui justifie aussi l'utilisation de la vraisemblance comme traduction de l'information contenue dans ces données.
Notons que le dénominateur de (2.11) sert à la fois de constante de normalisation [pour que $\pi(\theta|x)$ soit effectivement une densité de probabilité] et de vraisemblance marginale pour x, utile dans la comparaison de modèles et les tests d'hypothèses. Comme ce dénominateur est uniquement défini en fonction du numérateur, on utilise fréquemment la notation de proportionnalité $\pi(\theta|x) \propto f(x|\theta)\pi(\theta)$ qui signifie que la densité en θ, $\pi(\theta|x)$, est égale au produit $f(x|\theta)\pi(\theta)$ à une constante près et que cette constante s'obtient par l'intégration en θ. La constante dépend donc de x, vecteur des observations, ce qui n'est pas paradoxal puisque l'analyse bayésienne raisonne intégralement par conditionnement sur x.
Par exemple, si l'on cherche à déterminer un estimateur ponctuel du paramètre θ, on peut comparer les approximations d de θ au moyen d'une fonction de coût, $L(d, \theta)$, qui quantifie les conséquences de l'erreur commise en remplaçant le "vrai" paramètre θ par son approximation d. Une fois construite la loi a posteriori $\pi(\theta|x)$, les approximations d ont un coût moyen égal à $\mathbb{E}^{\pi}(L(d, \theta)|x)$ où cette notation signifie l'espérance sous $\pi(\theta|x)$ et l'approximation ou estimation optimale est celle qui minimise cette erreur.
On définit donc un estimateur bayésien $\delta(x)$ comme la procédure qui à chaque observation associe la solution du problème de minimisation

$$\delta(x) = \underset{d\in\Theta}{argmin} \quad \mathbb{E}^{\pi}\big(L(d, \theta)|x\big) \tag{2.12}$$

Cette solution exprime bien l'importance du choix de π. En pratique, il est souvent difficile de construire la fonction de perte véritablement associée au problème de décision.
Une fonction de perte par défaut est alors la fonction de perte quadratique :

$$L(d, \theta) = (d - \theta)^2 \tag{2.13}$$

Dans ce cas, l'estimateur bayésien est, si elle existe, l'espérance de la loi a posteriori

$$\delta(x) = \mathbb{E}^{\pi}\big(\theta|x\big) \tag{2.14}$$

2.1.6 Estimation par intervalles

La connaissance de la distribution a posteriori permet la détermination des régions de confiance sous forme des régions de plus forte densité à posteriori (Highest Posterior Den-sity, HPD), c'est-à-dire des régions de la forme :

$$\big\{\theta; \quad \pi(\theta|x) \geq k\big\} \tag{2.15}$$

dans le cas multidimensionnel comme dans le cas unidimensionnel. La motivation conduisant à cette forme de région de confiance (traditionnellement renommée région de crédibilité dans le cas bayésien pour distinguer la couverture au niveau nominal α en θ de la couverture au niveau nominal α en x même si l'utilisation est la même que pour les régions de confiance fréquentielles) est que ces régions sont de volume minimal à un niveau nominal donné.

2.2 Méthodes de calcul bayésien

2.2.1 Méthodes classiques d'approximation

Cette section couvre brièvement quelques techniques classiques qui peuvent faciliter les calculs bayésiens.

2.2.1.1 Intégration numérique

À partir de la simple méthode de Simpson, plusieurs approches ont été conçues en Mathématiques appliquées pour l'approximation numérique d'intégrales. Par exemple, la quadrature polynomiale est censée approcher les intégrales liées à des distributions proches de la loi normale,[Naylor et Smith, 1982]. L'approximation de base est donnée par :

$$\int_{-\infty}^{+\infty} e^{-t/2} f(t) dt \approx \sum_{i=1}^{n} w_i f(t_i)$$

où

$$w_i = \frac{2^{n-1} n! \sqrt{n}}{n^2 [H_{n-1}(t_i)]^2}$$

et t_i est le i-ième zéro du n-ième polynôme d'Hermite, $H_n(t)$.

D'autres approximations d'intégrales reliées à la méthode précédente sont disponibles, qui reposent sur différentes bases orthogonales classiques,[Abramowitz et Stegun, 1964],ou les ondelettes,[Muller etVidakovic, 1999].

Cependant, quel que soit la méthode d'intégration numérique utilisée, sa précision diminue dramatiquement lorsque la dimension de Θ augmente. De façon plus spécifique, l'erreur associée aux méthodes numériques se comporte comme une puissance de la dimension de Θ.

2.2.1.2 Méthodes de Monte Carlo

Dans un problème statistique, l'approximation de l'intégrale

$$\int_{\Theta} g(\theta) f(x|\theta) \pi(\theta) d\theta \tag{2.16}$$

doit tirer avantage de la nature particulière de (2.16), à savoir le fait que π soit une densité de probabilité (en supposant qu'il s'agisse d'une loi a priori propre) ou plutôt, que $f(x|\theta)\pi(\theta)$ soit proportionnel à une densité. Une conséquence naturelle de cette perspective est d'utiliser la méthode de Monte Carlo, introduite par Metropolis et Ulam (1949) et von Neumann (1951). Par exemple, s'il est possible de produire des variables aléatoires $\theta_1, \theta_2, ..., \theta_m$ de loi $\pi(\theta)$, la moyenne

$$\frac{1}{m} \sum_{i=1}^{m} g(\theta_i) f(x|\theta_i) \tag{2.17}$$

converge (presque surement) vers (2.16) lorsque m tend vers $+\infty$, selon la Loi des Grands Nombres. De la même façon, si un échantillon iid de θ_i de $\pi(\theta|x)$ peut être simulé, la moyenne

$$\frac{1}{m} \sum_{i=1}^{m} g(\theta_i) \tag{2.18}$$

converge vers

$$\frac{\int_{\Theta} g(\theta) f(x|\theta) \pi(\theta) d\theta}{\int_{\Theta} f(x|\theta) \pi(\theta) d\theta} \tag{2.19}$$

De plus, si la variance a posteriori $var(g(\theta)|x)$ est finie, le Théorème Central Limite s'applique à la moyenne (2.18), qui est alors asymptotiquement normale, de variance $var(g(\theta)|x)/m$.

La mise en œuvre de cette méthode nécessite la production d'une suite iid θ_i par ordinateur, reposant sur un générateur pseudo-aléatoire déterministe imitant la génération de $\pi(\theta)$ ou de $\pi(\theta|x)$ comme suit : un échantillon iid d'une loi uniforme $\mathcal{U}([0,1])$ est généré, puis transformé en variables de la loi d'intérêt.

2.2.1.3 Approximation analytique de Laplace

Lorsque la fonction à intégrer dans (2.16) est assez régulière, il existe une solution alternative analytique mais asymptotique aux simulations de Monte Carlo.
Cette méthode a été introduite par Laplace et est par conséquent appelée approximation de

Laplace. Soit une espérance a posteriori

$$\mathbb{E}[g(\theta)|x] = \frac{\int_{\Theta} g(\theta) f(x|\theta)\pi(\theta)d\theta}{\int_{\Theta} f(x|\theta)\pi(\theta)d\theta} \tag{2.20}$$

Ce rapport d'intégrales peut s'écrire

$$\mathbb{E}[g(\theta)|x] = \frac{\int_{\Theta} b_N(\theta) exp\{-nh_N(\theta)\}d\theta}{\int_{\Theta} b_D(\theta) exp\{-nh_D(\theta)\}d\theta} \tag{2.21}$$

où la dépendance en x est supprimée par souci de simplicité et où n est normalement la taille de l'échantillon.
Pour une fonction donnée h admettant un minimum unique $\widehat{\theta}$, le développement de Laplace d'une intégrale générale est donné par

$$\begin{aligned}\int_{\Theta} exp(-nh(\theta))d\theta = \sqrt{2\pi}\sigma exp(-n\widehat{h})\Big(\widehat{b} + \frac{1}{2n}\Big[\sigma^2\widehat{b}'' - \sigma^4\widehat{b}'\widehat{h}''' \\ + \frac{5}{12}\widehat{b}(\widehat{h}''')^2\sigma^6 - \frac{1}{4}\widehat{b}\widehat{h}^4\sigma^4\Big]\Big) + O(n^{-2})\end{aligned} \tag{2.22}$$

où $\widehat{b}$, $\widehat{h}$, etc., sont les valeurs prises par b, h et leurs dérivées pour $\theta = \widehat{\theta}$, et $\sigma^2 = [h''(\widehat{\theta})]^{-1}$.

2.2.2 Méthodes Monte Carlo par chaînes de Markov (MCMC)

2.2.3 Introduction aux chaînes de Markov

Définition 2.3
Soit $(X_n)_{n\geq 0}$ Un processus aléatoire à valeurs dans l'ensemble $S = (s_1, s_2, ..., s_k)$.

On dit que $(X_n)_{n\geq 0}$ est une chaîne de Markov homogène de matrice de transition P d'espace des états S, de loi initiale $\mathcal{L}_0$ si :

- *$\mathcal{L}_0$ est la loi suivie par X_0,*
- *quel que soit l'entier $n \geq 0$ et quels que soient les états $i, j, i_0, i_1, ..., i_{n-1}$ dans $\{1, ..., k\}$ tels que $P(X_0 = i_0, X_1 = i_1, ..., X_{n-1} = i_{n-1}, X_n = i) > 0$, on a*

$$\begin{aligned}P(X_{n+1} = s_j|X_0 = s_{i0}, X_1 = s_{i1}, ..., X_n = s_i) &= P(X_{n+1} = s_j|X_n = s_i) \\ &= P_{i,j}\end{aligned} \tag{2.23}$$

$P_{i,j}$ est la probabilité de passer de l'état s_i à l'état s_j en une étape.

Autrement : une chaine de Markov sur un espace S est un processus aléatoire où l'état actuel de la chaine étant donnée l'état précédent est indépendant de tous les états passés.

Théorème 2.2
Soit une chaîne de Markov (X_n) sur S de matrice de transition P et de loi initiale μ. Alors pour tout n, la loi de (X_n) est μP^n.

Définition 2.4
Étant donné une chaîne de Markov sur S de matrice de transition P, on dit que deux états s_i et s_j communiquent si la chaîne partant de s_i atteint un jour s_j avec une probabilité strictement positive, et réciproquement, si partant de s_j, elle atteint un jour s_i avec une probabilité strictement positive.
Une chaîne dont tous les états communiquent est dite irréductible ou ergodique.

Définition 2.5
Étant donné une chaîne de Markov sur S de matrice de transition P, on définit la période d'un état s_i par :

$$d(s_i) = pgcd\{n \geq 1 : \quad (P^n)_{ii} > 0\} \tag{2.24}$$

Une chaîne de Markov est dite apériodique si ses états ont pour période 1.

On peut facilement voir que si deux états communiquent, ils ont même période.
Le résultat suivant est une conséquence importante de l'apériodicité.

Théorème 2.3
Soit une chaîne de Markov apériodique (X_n) sur S de matrice de transition P.
Alors il existe $N < +\infty$ tel que

$$(P^n)_{i,i} > 0$$

pour tout $n \geq N$ et tout $i = 1, ..., k$.

L'apériodicité et l'irréductibilité donnent un résultat intermédiaire qui nous conduira au théorème de convergence des chaînes de Markov.

Théorème 2.4
Soit une chaîne de Markov apériodique irréductible (X_n) sur S de matrice de transition P. Alors il existe $M < +\infty$ tel que

$$(P^n)_{i,j} > 0$$

pour tout $n \geq M$ et tous $i, j = 1, ..., k$.

Dans la suite, on considère une chaîne de Markov (X_n), sur l'espace d'états fini S, de matrice de transition P.

Définition 2.6
Le vecteur-ligne $\pi = (\pi_1, ..., \pi_k)$ est une distribution stationnaire pour la chaîne (X_n) si :

i) *pour tout* $i = 1, ..., k, \quad \pi_i \geq 0 \quad$ *et* $\quad \sum_i \pi_i = 1$

ii) $\pi P = \pi$

En particulier, si la loi initiale de la chaîne est π, alors pour tout n, la loi de (X_n) est π.

Théorème 2.5 (Convergence des chaînes de Markov)
Pour une chaîne de Markov irréductible et apériodique, de loi initiale μ. Alors il existe une unique loi stationnaire π et la loi de (X_n) converge vers π pour la distance en variation totale, ie.

$$d_{VT}(\mu P^n, \pi) = \frac{1}{2} \sum_{i=1}^{k} |(\mu P^n)_i - \pi_i| \longrightarrow 0$$

On dit alors que la chaîne tend vers l'équilibre. La probabilité π_i d'être alors dans l'état s_i est la proportion moyenne de temps passé dans cet état.

Définition 2.7
Une distribution π sur S est dite réversible pour la chaîne (X_n) (ou pour la matrice de transition P) si pour tous i, j dans $\{1, ..., k\}$

$$\pi_i P_{i,j} = \pi_j P_{j,i}$$

Une chaîne de Markov est dite réversible si elle admet une distribution réversible.

Théorème 2.6
Si π est réversible pour la chaîne (X_n), alors elle est aussi une distribution stationnaire pour la chaîne.

Théorème 2.7 (Théorème ergodique)
On considère une chaine de Markov (X_n) de distribution limite π. Pour toute fonction intégrable h on a :

$$\lim_{n \longrightarrow +\infty} \sum_{i=0}^{n} h(X_i) = \int h(x)\pi(x)dx$$

2.2.4 Introduction aux méthodes MCMC

Nous considérons dans cette section une méthode de Monte Carlo plus générale, permettant d'approcher la génération de variables aléatoires d'une loi a posteriori $\pi(\theta|x)$ lorsque cette loi ne peut pas être simulée directement.
L'avantage de cette méthode sur les méthodes de Monte Carlo classiques décrites dans la Section 2.2.1.2 est qu'elle ne nécessite pas la construction précise d'une fonction d'importance, puisqu'elle prend en compte les caractéristiques de $\pi(\theta|x)$. Cette extension, appelée Monte Carlo par chaînes de Markov (et abrégée en MCMC), a des applications presque illimitées, même si ses performances varient largement, selon la complexité du problème. Elle tire son nom de l'idée que, pour produire des approximations acceptables d'intégrales et d'autres fonctions dépendant d'une loi d'intérêt, il suffit de générer une chaîne de Markov $(\theta_{(m)})_m$ de loi limite la loi d'intérêt. Cette idée d'utiliser le comportement limite d'une chaîne de Markov apparaît à la même époque que la technique de Monte Carlo originelle, au moins dans la littérature de Physique particulaire [Metropolis et al., 1953], mais elle nécessite une puissance de calcul qui n'était alors pas suffisamment grande pour être appréciée dans sa globalité.

Si les chaînes de Markov $(\theta_{(m)})_m$ produites par des algorithmes MCMC sont irréductibles, c'est-à-dire si elles peuvent visiter (avec probabilité non nulle) tout ensemble A tel que $\pi(A|x) > 0$, alors, de par leur nature même, ces chaînes sont récurrentes positives, de loi stationnaire $\pi(\theta|x)$, c'est-à-dire que le nombre moyen de visites d'un ensemble arbitraire A de mesure positive est infini. Ces chaînes de Markov sont aussi ergodiques, ce qui signifie que la loi de $(\theta_{(m)})$ converge vers $\pi(.|x)$ pour presque toute valeur initiale $\theta_{(0)}$; en d'autres termes, l'influence de la valeur initiale disparaît. Par conséquent, pour k suffisamment grand, le $\theta_{(k)}$ résultant est distribué approximativement selon $\pi(\theta|x)$, quelle que soit la valeur initiale $\theta_{(0)}$.

2.2.4.1 Algorithmes de Metropolis-Hastings

Dans sa version moderne, l'algorithme de Metropolis-Hastings peut être décrit de la façon suivante. Pour une densité donnée $\pi(\theta)$, connue à un facteur de normalisation près, et une

densité conditionnelle $q(\theta'|\theta)$, l'algorithme génère la chaîne $(\theta^{(m)})_m$ comme suit :

Algorithme 2 (Metropolis-Hastings)

Itération 0 : *Initialiser avec une valeur arbitraire* $\theta^{(0)}$

Itération m : *Mettre à jour* $\theta^{(m)}$ *par* $\theta^{(m+1)}$ *de la façon suivante :*

a) *Générer* $\xi \sim q(\xi|\theta^{(m)})$

b) *Poser*

$$\varrho(\theta^{(m)}, \xi) = \frac{\pi(\xi) q(\theta^{(m)}|\xi)}{\pi(\theta^{(m)}) q(\xi|\theta^{(m)})} \wedge 1$$

c) *Prendre*

$$\theta^{(m+1)} = \begin{cases} \xi & \text{avec la probabilité } \varrho(\theta^{(m)}, \xi), \\ \theta^{(m)} & \text{sinon.} \end{cases}$$

La loi de densité $\pi(\theta)$ est souvent appelée loi cible ou loi objet, tandis que la loi de densité $q(.|\theta)$ est dite loi de proposition. Une propriété stupéfiante de cet algorithme est d'autoriser un nombre infini de lois de proposition produisant toutes une chaîne de Markov convergeant vers la loi d'intérêt.

Théorème 2.8

Si la chaîne $(\theta^{(m)})_m$ *est irréductible, c'est-à-dire si, pour tout sous-ensemble* A *tel que* $\pi(A) > 0$, *il existe* M *tel que* $P_{\theta^{(0)}}(\theta^{(m)} \in A) > 0$, *alors* π *est la loi stationnaire de la chaîne. Si de plus la chaîne est apériodique, elle est aussi ergodique de loi limite pi, pour presque toute valeur initiale* $\theta^{(0)}$, *au sens où*

$$\lim_{m \longrightarrow \infty} \sup_A |P_{\theta^{(0)}}(\theta^{(m)} \in A) - \pi(A)| = 0 \quad p.s.$$

2.2.4.2 Èchantillonnage de Gibbs

La technique de Metropolis-Hastings présentée dans la section précédente est attrayante de par son universalité, mais, d'un autre coté, le manque de connexion entre le mécanisme de proposition q et la loi cible π peut être néfaste pour les propriétés de convergence de la méthode et, dans la pratique, peut facilement empêcher la convergence si la probabilité d'atteindre des parties éloignées du support de la loi π est trop petite.

L'approche de l'échantillonnage de Gibbs, qui repose sur une perspective différente, est pour sa part fondée sur la loi π. Cette méthode tire son nom des champs aléatoires de Gibbs, où elle a été utilisée pour la première fois par Geman (1984).
D'un point de vue général, l'échantillonnage de Gibbs tire profit des structures hiérarchiques d'un modèle, par exemple lorsque celui-ci peut s'écrire sous la forme :

$$\pi(\theta|x) = \int \pi_1(\theta|x, \lambda)\pi_2(\lambda|x)d\lambda \tag{2.25}$$

L'idée est alors de simuler la loi jointe $\pi_1(\theta|x, \lambda)\pi_2(\lambda|x)$, afin d'obtenir $\pi(\theta|x)$ comme la loi marginale. Bien entendu, lorsque les deux lois $\pi_1(\theta|x, \lambda)$ et $\pi_2(\lambda|x)$ sont connues et peuvent être simulées, la génération de θ de $\pi(\theta|x)$ est équivalente à la génération de λ de $\pi_2(\lambda|x)$, puis de θ de $\pi_1(\theta|x, \lambda)$.
Une première technique d'échantillonnage de Gibbs, d'abord appelée augmentation des données parce que utilisée dans ce contexte, a été introduite par Tanner et Wong (1987) afin de tirer profit des lois conditionnelles selon l'algorithme itéré suivant :

Algorithme 3 (Échantillonnage de Gibbs)
Itération 0 : *Commencer par une valeur arbitraire* $\lambda^{(0)}$.
Itération t : *pour* $\lambda^{(t-1)}$ *donné, générer*
a. $\theta^{(t)}$ *selon* $\pi_1(\theta|x, \lambda^{(t-1)})$
b. $\lambda^{(t)}$ *selon* $\pi_2(\lambda|x, \theta^{(t)})$

2.3 Rappel sur la distribution gaussienne

En théorie des probabilités et en statistique, la loi normale est l'une des lois de probabilité les plus adaptées pour modéliser des phénomènes naturels issus de plusieurs événements aléatoires. Elle est en lien avec de nombreux objets mathématiques dont le mouvement brownien, le bruit blanc gaussien ou d'autres lois de probabilité. Elle est également appelée loi gaussienne, loi de Gauss ou loi de Laplace-Gauss des noms de Laplace (1749-1827) et Gauss (1777-1855), deux mathématiciens, astronomes et physiciens qui l'ont étudiée.

2.3.1 Distribution gaussienne dans le cas unidimensionnel

la loi gaussienne est une loi de probabilité absolument continue qui dépend de deux paramètres : son espérance, un nombre réel noté μ, et son écart type, un nombre réel positif noté σ. La densité de probabilité de la loi normale est donnée par :

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} exp\Big(-\frac{1}{2}\big(\frac{x-\mu}{\sigma}\big)^2 \Big)$$

La courbe de cette densité est appelée courbe de Gauss ou courbe en cloche, entre autres. C'est

la représentation la plus connue de cette loi. La loi normale de moyenne nulle et d'écart type unitaire est appelée loi normale centrée réduite ou loi normale standard.

Lorsqu'une variable aléatoire X suit la loi normale, elle est dite gaussienne ou normale et il est habituel d'utiliser la notation avec la variance σ^2 :

$$X \sim \mathcal{N}(\mu, \sigma^2)$$

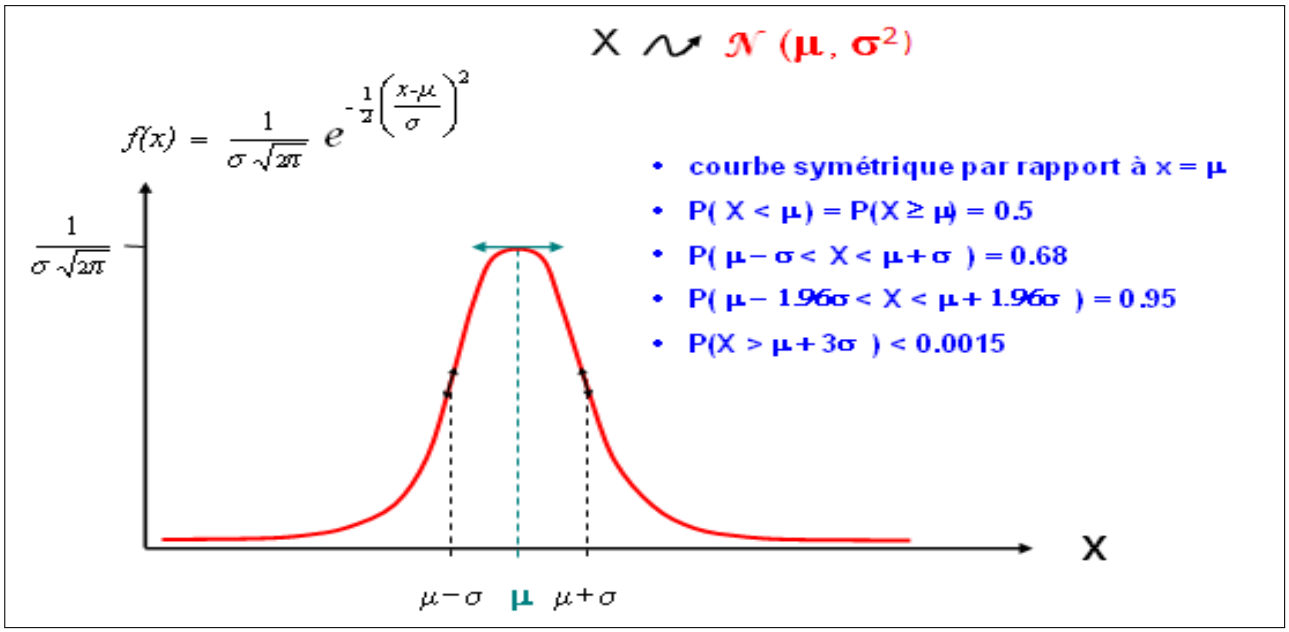

Figure 2.1 – La courbe de la loi gaussienne

2.3.2 Distribution gaussienne dans le cas multidimensionnel

On appelle loi normale multidimensionnelle, ou Normale multivariée ou loi multinormale ou loi de Gauss à plusieurs variables, une loi de probabilité qui est la généralisation multidimensionnelle de la loi normale.
Alors que la loi normale classique est paramétrée par un scalaire μ correspondant à sa moyenne et un second scalaire σ^2 correspondant à sa variance, la loi multinormale est paramétrée par un vecteur $\boldsymbol{\mu} \in \mathbb{R}^N$ représentant son centre et une matrice semi-définie positive $\Sigma \in \mathcal{M}_N(\mathbb{R})$ qui est sa matrice de variance-covariance.
Dans le cas non dégénéré où Σ est définie positive, donc inversible, la loi normale multidimensionnelle admet la densité de probabilité suivante :

$$f_{\mu,\Sigma}(x) = \frac{1}{(2\pi)^{N/2}|\Sigma|^{1/2}} exp\Big(-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)\Big)$$

où $|\Sigma|$ est le déterminant de la matrice variance-covariance Σ.

Cette loi est habituellement notée $\mathcal{N}(\boldsymbol{\mu}, \Sigma)$ par analogie avec la loi normale unidimensionnelle.

Conclusion

L'utilisation du théorème de Bayes permet de proposer une formulation bayésienne de l'apprentissage, qui est très générale, et qui peut s'appliquer à la recherche de tous types de modèles pour peu que ceux-ci soient représentés par un ensemble de paramètres sur lesquels on peut faire une hypothèse de distribution. Nous avons énoncé dans ce chapitre les notions de base du formalisme bayésien et nous avons détaillé quelques méthodes du calcul bayésien.

CHAPITRE 3

INTÉGRATION DES MÉTHODES BAYÉSIENNES DANS L'APPRENTISSAGE DES RÉSEAUX DE NEURONES

Introduction

Beaucoup d'efforts considérables ont été effectués par les chercheurs dans le but d'améliorer des réseaux de neurones. Parmi ces améliorations on peut citer l'intégration des méthodes bayésiennes dans l'apprentissage des réseaux de neurones. L'approche bayesienne a été appliquée ces dernières années aux réseaux de neurones par différents auteurs, notamment dans les travaux de MacKay [12], [13], repris dans Neal [36]. Une synthèse de ces différentes approches peut être trouvée dans Bishop [8].

3.1 Généralités sur l'approche bayésienne pour les réseaux de neurones

3.1.1 Principe de l'approche bayesienne pour les RNA

L'apprentissage classique était effectué en trouvant une valeur du vecteur des poids qui minimise une fonction de coût ou d'erreur.
Dans les méthodes bayésiennes, tous les paramètres, notamment les poids du réseau, sont considérés comme des variables aléatoires issues d'une distribution de probabilité.
L'apprentissage d'un réseau de neurones consiste donc à déterminer la distribution de probabilité des poids connaissant les données d'apprentissage : on attribue aux poids une probabilité fixée a priori, et une fois que les données d'apprentissage ont été observées, cette probabilité a priori est transformée en probabilité a posteriori grâce au théorème de Bayes. Ainsi, si D représente l'ensemble des données d'apprentissage, $P(w)$ est la densité de probabilité a priori des poids, $P(D|w)$ la densité de probabilité d'observer les données connaissant les poids du réseau, et $P(w|D)$ la probabilité a posteriori que l'on cherche à déterminer, alors le théorème de Bayes s'écrit :

$$P(w|D) = \frac{P(D|w)P(w)}{P(D)} \tag{3.1}$$

3.1.2 Avantages de l'approche bayesienne pour les RNA

D'un point de vue théorique, Neal [36] a montré que, lorsque les probabilités a priori des poids sont convenablement choisies, il n'est pas nécessaire de limiter la taille du réseau pour éviter le surajustement, et le nombre de neurones cachés peut tendre vers l'infini. Selon cette étude, le seul facteur qui doit limiter la taille du réseau est la capacité des ordinateurs utilisés et le temps disponible pour effectuer les calculs nécessaires.
D'un point de vue pratique, la théorie de l'approche bayesienne pour l'apprentissage des réseaux de neurones apporte d'importantes améliorations :

- Le concept de régularisation peut être interprété de façon naturelle dans le contexte bayésien.
- Les hyperparamètres intervenant dans la fonction de régularisation sont calculés lors de la phase d'apprentissage sans utiliser de base de validation.
- Le calcul de l'évidence explicité à la section 3.2 permet de sélectionner, parmi une famille de modèles, le meilleur modèle, uniquement grâce à la base d'apprentissage.
- Comme tous les calculs se font à partir de la base d'apprentissage, il n'est plus nécessaire de disposer d'une base de validation. Il est donc possible d'utiliser toutes les données dont on dispose pour estimer les poids du réseau.
- Des barres d'erreurs peuvent être calculées pour les problèmes de régression.
- L'incertitude sur les poids peut être prise en considération pour corriger la probabilité calculée par un réseau dans un problème de classification.
- Les entrées peuvent être sélectionnées grâce à la méthode Automatic Relevance Determination : un hyperparamètre est associé à chaque entrée et après l'apprentissage, les hyperparamètres avec de grandes valeurs indiquent des entrées non pertinentes.

3.1.3 Inconvénients de l'approche bayesienne pour les RNA

Comme les paramètres utilisés sont maintenant issus de distributions de probabilité, il est nécessaire, pour connaître un paramètre, de calculer des intégrales faisant intervenir les distributions des autres paramètres. Il est, en général, impossible de calculer ces intégrales analytiquement, et plusieurs approches ont été proposées pour effectuer ces calculs. Mais soit ces méthodes sont très lourdes à implémenter, soit elles reposent sur des approximations qui peuvent fausser les résultats.
Finalement les résultats théoriques proposés par l'approche bayesienne sont souvent inapplicables en l'état dans le cadre des réseaux de neurones.
Dans ses travaux, Neal [36] utilise des méthodes de Monte Carlo couplées à des modèles de Markov cachés pour calculer les différentes intégrales intervenant dans les différentes étapes.

Les calculs sont très lourds à mettre en place et nécessitent beaucoup de temps de calcul. Nous n'avons pas cherché dans ce travail à utiliser cette approche.
MacKay [12], [13] et [16], a proposé des approximations reposant sur des hypothèses gaussiennes des probabilités a posteriori. Grâce à ces hypothèses, les calculs d'intégrales se trouvent simplifiés et peuvent être effectués plus ou moins simplement. Ces approximations sont parfois discutables, surtout pour les problèmes de classification. Néanmoins, grâce à ces approximations, les calculs sont simplifiés de sorte que l'approche bayesienne devient utilisable en pratique. L'approche proposée par MacKay est connue sous le nom de evidence framework.

3.2 Procédure d'évidence

En se basant sur les travaux de MacKay, on propose d'utiliser des hypothèses d'approximations gaussiennes, on supposera pour ce faire que le bruit sur les données utilisées, est gaussien et que donc, la distribution des erreurs le sera aussi. Les FDP, a priori, des paramètres auront également une distribution gaussienne.

3.2.1 Distribution a priori des poids et des biais

Partons du principe que l'on a qu'une petite idée de la valeur des poids du réseau. Pour n'écarter aucune possibilité, on choisit de considérer que la PDF des paramètres du réseau de neurones a priori, est une gaussienne avec une variance élevée :

$$P(w|\alpha) = \frac{1}{Z_w(\alpha)} exp(-\alpha E_w) \tag{3.2}$$

Avec α représente l'inverse de la variance dans l'ensemble des poids et des biais, et $Z_w(\alpha)$ représente la constante de normalisation de la FDP. Dans le cadre Bayésien, α est un hyperparamètre qui contrôle la distribution des autres paramètres. Le choix d'une probabilité à priori gaussienne pour les poids permet une interprétation probabiliste de E_w ainsi que son interprétation comme étant le minimum du log de la distribution de la probabilité à priori sur les paramètres (poids et biais du réseau).
En considérant que α est connu, et comme on a choisi une distribution gaussienne, on en déduit la constante de normalisation $Z_w(\alpha)$:

$$Z_w(\alpha) = \int exp(-\alpha E_w) dw \tag{3.3}$$

$$= \left(\frac{2\pi}{\alpha}\right)^{m/2} \tag{3.4}$$

avec m représente le nombre total de paramètres du réseau de neurones.

3.2.2 Fonction de vraisemblance

Considérons l'échantillon $D = \{x_i, d_i\}_{i=1}^N$ de taille N, utilisé pour l'apprentissage. Le réseau de neurones a pour objectif de trouver une relation R entre les entrées x_i et les sorties désirées d_i (cibles). Les données utilisées ayant de forte chances d'être bruitées, la relation se traduit par :

$$d_i = R(x_i) + \varepsilon_i \tag{3.5}$$

Considérons que les erreurs ε_i ont une distribution normale centrée en zéro et une variance $\sigma^2 = 1/\beta$. La distribution du bruit s'écrit alors :

$$P(\varepsilon_i|\beta) = \sqrt{\frac{\beta}{2\pi}}\Big(- \frac{\beta}{2}\varepsilon i\Big) \tag{3.6}$$

La probabilité commune à l'échantillon de N valeurs de bruit s'écrira :

$$P(\varepsilon|\beta) = P(\varepsilon_1, \varepsilon_2, ..., \varepsilon_N|\beta) \tag{3.7}$$

$$= \prod_{i=1}^{N} P(\varepsilon_i|\beta) \tag{3.8}$$

$$= \Big(\frac{\beta}{2\pi}\Big)^{N/2} exp\Big(- \frac{\beta}{2}\sum_{i=1}^{N}\varepsilon_i^2\Big) \tag{3.9}$$

Or $\varepsilon_i = d_i - y_i$, en remplaçant dans l'expression précédente on obtient :

$$P(\varepsilon|\beta) = \Big(\frac{\beta}{2\pi}\Big)^{N/2} exp\Big(- \frac{\beta}{2}\sum_{i=1}^{N}(d_i - y_i)^2\Big) \tag{3.10}$$

Alors la fonction de vraisemblance est donnée par :

$$P(D|w, \beta) = \frac{1}{Z_D(\beta)} exp(-\beta E_D) \tag{3.11}$$

Avec β est un hyperparamètre qui représente l'inverse de la variance du bruit de la cible.
On suppose que l'hyperparamètre β est connu. Alors la distribution gaussienne du bruit permet de calculer le facteur de normalisation $Z_D(\beta)$ qui est donné par :

$$Z_D(\beta) = \Big(\frac{2\pi}{\beta}\Big)^{N/2} \tag{3.12}$$

De plus, le fait de choisir une fonction vraisemblance gaussienne conduit à une interprétation probabiliste de la fonction erreur puisque la fonction erreur peut être interprétée comme étant le minimum de la fonction log-vraisemblance de l'équation (3.11).

3.2.3 Distribution à posteriori des paramètres

Nous avons définit, précédemment, la fonction de vraisemblance et la probabilité à priori comme étant égale respectivement à :

$$P(D|w, \beta) = \frac{1}{Z_D(\beta)} exp(-\beta E_D) \tag{3.13}$$

$$P(w|\alpha) = \frac{1}{Z_w(\alpha)} exp(-\alpha E_w) \tag{3.14}$$

Alors il est possible de définir la FDP a postiori en vertu du théorème de Bayes :

$$P(w|\alpha,\beta,D) = \frac{P(D|w,\beta)P(w|\alpha)}{P(D|\alpha,\beta)} \tag{3.15}$$

On obtient alors :

$$P(w|\alpha,\beta,D) = \frac{1}{Z_S(\alpha,\beta)} exp[-S(w)] \tag{3.16}$$

avec

$$S(w) = \alpha E_w + \beta E_D \tag{3.17}$$

Et $Z_S(\alpha,\beta)$ la constante de normalisation donnée par :

$$Z_S(\alpha,\beta) = \int exp(-S(w))dw \tag{3.18}$$

Dans le cas général cet intégral ne peut pas être calculer analytiquement.
Dans ce cadre Makay a proposé une estimation de la probabilité a postériori via l'hypothèse de l'approximation gaussienne.

3.2.4 Approximation gaussienne de la distribution a postriori

La distribution de la probabilité à postériori, est l'exemple type d'une distribution multimodale dans l'espace des poids et chaque minimum local correspond une interprétation différente des données. En effet chaque jeu de poids w trouvé (localement) en minimisant $S(w)$ est interprété comme le minimum local le plus probable et est noté w_{MP}. Nous pouvons donc localement approximer $S(w)$ comme étant une fonction quadratique. En utilisant l'Hessien de la fonction erreur, noté A, calculée en fonction du jeu de poids, nous obtenons :

$$S(w) \approx \Big(- S(w_{MP}) - \frac{1}{2}(w - w_{MP})^t A (w - w_{MP})\Big) \tag{3.19}$$

Où A est la matrice hessienne de la fonction erreur $S(w)$:

$$A = \nabla\nabla S(w) \tag{3.20}$$

L'approximation gaussienne de la probabilité a postériori permet aussi de calculer le facteur normalisation après l'approximation $Z_S^{'}(\alpha,\beta)$ qui est donné par :

$$Z_S^{'}(\alpha,\beta) = (2\pi)^{m/2} * |A|^{-1/2} * exp(-S(w_{MP})) \tag{3.21}$$

Où $|A|$ est le déterminant de la matrice hessienne A de la fonction $S(w)$.

3.2.5 Structure d'évidence

Pour calculer le facteur de normalisation $Z_S^{'}(\alpha, \beta)$ et la constante de normalisation $Z_w(\alpha)$ nous avons considéré, dans les deux cas, que les hyperparamètres α et β étaient connus. Mais ces hyperparamètres doivent eux aussi être estimés.
Mackay donne une méthode basée sur la structure de l'évidence qu'on va détailler dans cet axe. Tout d'abord appliquons le théorème de Bayes aux hyper-paramètres afin d'avoir leurs probabilités a postériori, on obtient :

$$P(\alpha, \beta|D) = \frac{P(D|\alpha, \beta)P(\alpha, \beta)}{P(D)} \tag{3.22}$$

Avec $P(\alpha, \beta)$ la distribution a priori des des hyperparamètres, mais cette fois ci nous n'avons aucune idée de la valeur des hyperparamètres, alors on considère que leur distribution suit une loi uniforme.
De plus, le facteur de normalisation $P(D)$ ne dépend pas des hyperparamètres α et β.
Donc maximiser la probabilité a postériori $P(\alpha, \beta|D)$ revient à maximiser le terme $P(D|\alpha, \beta)$ appelé l'évidence pour α et β.
En utilisant les approximations gaussiennes pour la FDP a postriori des poids, on peut écrire le terme de l'évidence sous la forme suivant :

$$P(D|\alpha, \beta) = \frac{Z_S^{'}(\alpha, \beta)}{Z_w(\alpha)Z_D(\beta)} \tag{3.23}$$

$$= \frac{(2\pi)^{m/2} * |A|^{-1/2} * exp[-S(w_{MP})]}{\left(\frac{2\pi}{\alpha}\right)^{m/2}\left(\frac{2\pi}{\beta}\right)^{N/2}} \tag{3.24}$$

Les valeurs optimales des hyperparamètres α_{MP} et β_{MP} correspondent à une maximisation de l'évidence. Pour ce faire, nous utilisons la fonction logarithmique log, ce qui équivaut à considérer que maximiser une probabilité revient à minimiser la fonction erreur qui s'y rattache.

$$logP(D|\alpha, \beta) = -\frac{1}{2}log|A| - S(w_{MP}) - \frac{N}{2}log(2\pi) + \frac{N}{2}log(\beta) + \frac{m}{2}log(\alpha) \tag{3.25}$$

En dérivant l'équation ci-dessus par rapport à α puis par rapport à β on trouve les valeurs optimales α_{MP} et β_{MP} suivantes :

$$\alpha_{MP} = \frac{\gamma}{2E_w^{MP}} \tag{3.26}$$

$$\beta_{MP} = \frac{N - \gamma}{2E_D^{MP}} \tag{3.27}$$

Avec : $E_w^{MP} = E_w(w_{MP})$ et $E_D^{MP} = E_D(w_{MP})$.

La quantité γ désigne le nombre de paramètres convenablement déterminés et vaut :

$$\gamma = \sum_{k=1}^{m} \frac{\lambda_k}{\lambda_k + \alpha} \tag{3.28}$$

λ_k représente les valeurs propres de la matrice hessienne H de l'erreur E_D, i,e. $H = \beta \nabla \nabla E_D$.

Maintenant que les hyper-paramètres peuvent être optimisés résumons le processus conduisant au jeu de poids le plus probable.

Algorithme 4 (evidence framework)

1. *Initialisation :*
   On choisit des valeurs de α et β. Ce qui permet d'initialiser les poids du réseau en utilisant les valeurs provenant de la distribution à priori.
   À l'itération i on aura donc une estimation des poids w^i, des hyper-paramètres α^i et β^i et de l'erreur $S^i(w)$.

2. *déterminer les valeurs w_{MP}^{i+1} qui minimisent l'erreur $S^i(w)$ en utilisant,une méthode d'optimisation non linéaire, par exemple, l'algorithme du gradient conjugué. Après obtention du jeu de poids le plus probable à l'instant $i+1$ on calcule E_w^{i+1} et E_D^{i+1}.*

3. *La ré-estimation des hyper-paramètres :*
   On calcule les nouvelles valeurs des hyper-paramètres α^{i+1} et β^{i+1} suivant trois étapes :
   a)
   $$\gamma^{i+1} = \sum_{k=1}^{m} \frac{\lambda_k}{\lambda_k + \alpha^i}$$
   b)
   $$\alpha^{i+1} = \frac{\gamma^{i+1}}{2E_w^{i+1}}$$
   c)
   $$\beta^{i+1} = \frac{N - \gamma^{i+1}}{2E_D^{i+1}}$$

4. *Répétez les étapes 2 et 3 en utilisant les nouvelles valeurs des hyper-paramètres.*

Le processus s'arrêtera lorsque l'erreur $S(w)$ sera égal à la moitié du nombre de points des données, soit $S(w) = \frac{N}{2}$ alors $\alpha = \alpha_{MP}$ et $\beta = \beta_{MP}$.

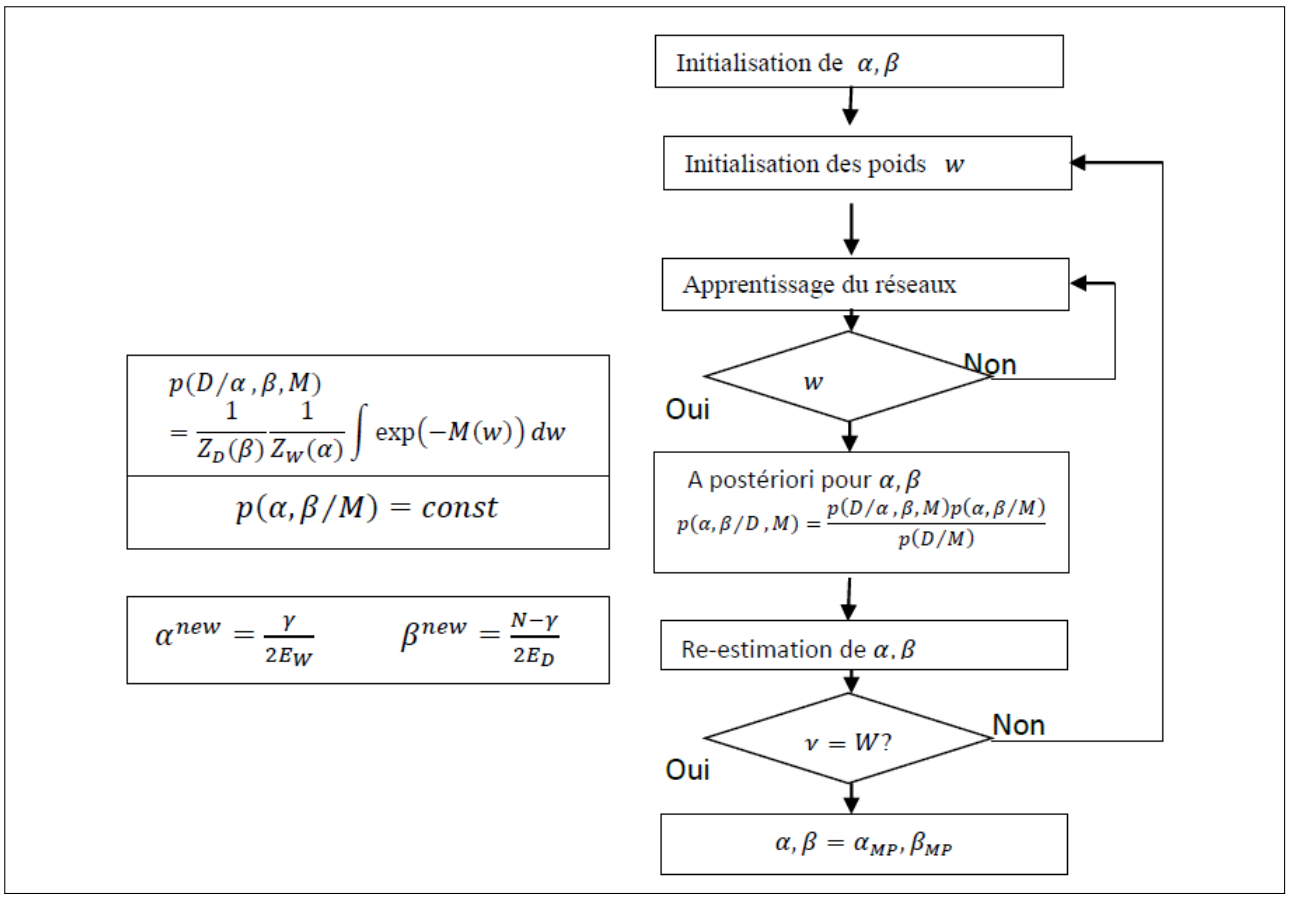

Figure 3.1 – L'architecture générale de la procédure d'évidence

3.2.6 Méthode "Automatic Relevance Determination" (ARD)

La probabilité à priori qui mène à $S(w)$ définie par l'équation (3.17), n'est pas la meilleure car elle applique les mêmes restrictions (α) aux jeux de poids des différentes couches, ce qui n'est pas conforme aux propriétés d'échelles d'un réseau de neurones.
De meilleurs résultats peuvent être obtenus si on sépare les poids en différents groupes (un groupe pour les poids et les biais de chaque couche).
La valeur d'un α pourrait alors être utilisée comme indicateur de la pertinence des entrées qu'il contrôle : si le α associé est très grand, le processus de minimisation forcerait les poids à être très petits et l'entrée aurait alors peu de pertinence sur la sortie. Cette méthode s'appelle "the automatic relevance determination" (ARD) et permet de déterminer l'influence des entrées sur la sortie du réseau. Ainsi les entrées de faible pertinence ne sont pas retirer du réseau, l'algorithme réduit simplement l'influence des ces entrées en réduisant leurs poids.

3.2.7 Distribution des sorties du réseau

En utilisant les règles de probabilité, on peut écrire la densité des sorties du réseau en présence d'un vecteur en entrée x sous la forme :

$$p(t|x, D) = \int p(t|x, w)p(w|D)dw \tag{3.29}$$

Avec $p(w|D)$ est la distribution a posteriori des poids.
La distribution $p(t|x, w)$ est simplement le modèle de distribution de bruit sur les données cibles,

pour une valeur fixe du vecteur poids w, et est donné par :

$$p(t|x, w) \propto exp\Big(-\frac{\beta}{2}\big(y(x, w) - t\big)^2\Big) \tag{3.30}$$

Avec $y(x, w)$ représente la sortie donnée par le réseau en présence du vecteur x en entré.

Pour évaluer cette distribution, nous considérons l'approximation gaussienne (3.26) pour la distribution a posteriori des poids, ainsi que l'expression (3.30) pour la distribution des sorties du réseau. Cela donne :

$$p(t|x, D) \propto \int exp\Big(-\frac{\beta}{2}\big(y(x, w) - t\big)^2\Big) exp\Big(-\frac{1}{2}\Delta w^T A\, \Delta w\Big) dw \tag{3.31}$$

On faisant un développement de $y(x, w)$ à l'ordre 1 autour de son minimum w_{MP} :

$$y(x, w) = y(x, w_{MP}) + g^T \Delta w \tag{3.32}$$

Avec

$$g = \nabla_w y|_{w_{MP}} \tag{3.33}$$

$$\Delta w = w - w_{MP} \tag{3.34}$$

On remplaçant $y(x, w)$ par son expression dans la formule (3.), on obtient :

$$p(t|x, D) \propto \int exp\Big(-\frac{\beta}{2}\big(t - y_{MP} - g^T \Delta w\big)^2 - \frac{1}{2}\Delta w^T A\, \Delta w\Big) dw \tag{3.35}$$

Par des calculs d'intégrale gaussienne, on obtient :

$$p(t|x, D) = \frac{1}{(2\pi\sigma_t^2)^{1/2}} exp\Big(-\frac{(t - y_{MP})^2}{2\pi\sigma_t^2}\Big) \tag{3.36}$$

Cette distribution a une moyenne donnée par y_{MP} et une variance donnée par :

$$\sigma_t^2 = \frac{1}{\beta} + g^T A^{-1} g \tag{3.37}$$

Nous pouvons interpréter l'écart-type σ_t de la distribution prédictive pour t comme une barre d'erreur sur la valeur moyenne y_{MP}. Cette barre d'erreur comporte deux contributions, une résultant du bruit intrinsèque sur les données cibles, correspondant au premier terme dans (3.37), et une résultant de la largeur de la distribution postérieure des poids du réseau, correspondant au deuxième terme en (3.37).
Nous voyons que le formalisme bayésien nous permet de calculer les barres d'erreur sur les sorties du réseau, au lieu de simplement fournir une seule sortie "meilleure estimation". Dans une mise en œuvre pratique, nous trouvons d'abord les poids W_{MP} les plus probables en minimisant la fonction d'erreur normalisée $S(w)$. Nous pouvons ensuite attribuer des barres d'erreur à cette fonction réseau en évaluant la matrice Hessien et en utilisant (3.37).

3.2.8 Sélection du modèle bayésien

Comme indiqué avant, les méthodes bayésiennes sont utilisées pour des problèmes d'estimation de paramètres, mais l'approche bayésienne peut également être utilisée pour des problèmes de sélection de modèle. En effet, le formalisme bayésien (en utilisant encore le théorème de Bayes) peut calculer et associer une probabilité à un modèle. Cette probabilité (également appelée l'évidence du modèle) peut être utilisée comme mesure pour sélectionner le meilleur modèle. En d'autres termes, le modèle qui présente la plus grande évidence est le plus probable, compte tenu des données.
Dans le contexte RNA, les réseaux de neurones avec différents nombres de neurones cachés peuvent être comparés et classés en fonction de leurs évidence.
Pour illustrer ces énoncés, considérons un ensemble de modèles M_i (qui peut correspondre dans notre cas à un ensemble de réseaux de neurones avec différents nombres de neurones cachés). Du théorème de Bayes, on peut calculer les probabilités a posteriori des différents modèles M_i, compte tenu des données D

$$p(M_i|D) = \frac{p(D|M_i)p(M_i)}{p(D)} \tag{3.38}$$

Où $p(M_i)$ est la probabilité a priori attribuée au modèle M_i, $p(D)$ est la constante de normalisation et la quantité $p(D|M_i)$ est appelée l'évidence pour M_i.
Comme nous n'avons aucune raison d'attribuer des prieurs différents à différents modèles, les modèles sont ensuite comparés et classés en fonction de leurs évidences.
En utilisant certaines hypothèses (et en particulier l'approximation gaussienne de la distribution a posteriori des poids), Bishop [**?**] donne une expression de log de l'évidence d'un réseau de neurones ayant h neurones cachées comme suit :

$$\begin{aligned} log\, p(D|M_i) = &-\alpha_{MP} E_w^{MP} - \beta_{MP} E_D^{MP} - \frac{1}{2} log|A| \\ &+ \frac{m}{2} log(\alpha_{MP}) + \frac{N}{2} log(\beta_{MP}) + log(h!) \\ &+ 2log(h) + \frac{1}{2} log(\frac{2}{\gamma}) + \frac{1}{2} log(\frac{2}{N-\gamma}) \end{aligned} \tag{3.39}$$

3.2.9 Procédure d'evidance pour un problème de classification

On considère un pb de classification avec c classes $\{C_1, ..., C_c\}$, et soit un perceptron multicouche, dont la couche de sortie est composée de c neurones.
Afin de discriminer les modèles appartenant à une classe C_k parmi les c classes, les unités de sortie $\{y_1, ..., y_c\}$, doivent être interprétées comme les probabilités a posteriori $p(C_k|x)$ que le modèle x appartient à la classe C_k. Ceci est obtenu en imposant deux conditions sur les fonctions de sortie :

$$y_k(x) \in [0, 1] \quad et \quad \sum_{k=1}^{c} y_k(x) = 1$$

La fonction d'activation pour les neurones de sortie est donnée par la fonction softmax (softmax function) :

$$y_k(x) = \frac{exp(a_k(x))}{\sum_{l=1}^{c} exp(a_l(x))} \tag{3.40}$$

Avec $a_k(x)$ est donné par la somme pondérée des valeurs de sortie de toutes les neurones de la couche précédente.
La probabilité de mauvaise classification est minimisée en attribuant le modèle x à la classe C_k lorsque

$$y_k(x) > y_l(x) \quad \forall I = 1, ..., c, \ I \neq k$$

La prédiction bayésienne pour une nouvelle entrée x est donnée par la marginalisation de la classification effectuée par le modèle $p(C_k|x, w)$ par rapport à la distribution a posteriori des paramètres :

$$p(C_k|x, D) = \int p(C_k|x, w)p(w, \alpha|D)dwd\alpha \tag{3.41}$$

Le deuxième terme de l'intégrale est donné par :

$$p(w, \alpha|D) = p(w|\alpha, D)p(\alpha|D) \tag{3.42}$$

Comme l'intégrale (3.41) ne peut pas être calculé analytiquement, nous devons utiliser des approximations numériques.

Procédure d'évidence

La procédure d'évidence donne une approximation pour l'équation (3.40) en supposant que la probabilité a posteriori de l'hyperparamètre $p(\alpha|D)$ est fortement creusé autour de son maximum α^{mp}.Pour refléter l'absence de connaissances a priori sur la meilleure valeur de α, la probabilité a priori $p(\alpha)$ est choisi pour avoir la forme du produit $p(\alpha) = \prod_g p(\alpha_g)$ où g indexe un groupe tel que chaque $p(\alpha_g)$ étant constant sur une échelle logarithmique.Ainsi, on peut trouver la valeur d'une maximisation du $log\, p(\alpha|D)$ en maximisation $log\, p(D|\alpha)$
$p(D|\alpha)$ est donnée par la formule :

$$p(D|\alpha) = \int p(D|w)p(w|\alpha)dw \tag{3.43}$$

Dans la procédure d'évidence l'intégrale est approximée en utilisant l'approximation de Laplace autour de w_{mp}, la valeur de w qui maximise $p(w|\alpha, D)$. Il s'avère que les composants de la valeurs optimale α^{mp} sont donnés par :

$$\alpha_g^{mp} = \frac{\gamma_g}{\sum_{i \in g} (w_i^{mp})^2} \tag{3.44}$$

Avec

$$\gamma_g = w_g - \alpha_g \sum_{i \in g} ((\nabla\nabla G + \alpha I)^{-1})_{ii}$$

w_g est le nombre total des poids contrôlés par l'hyperparamètre α_g,
G est le négatif de la fonction vraisemblance pénalisée : $G = -In\, p(D|w) - In\, p(w|\alpha)$.
La somme prend en compte les éléments de la matrice $(\nabla\nabla G + \alpha I)^{-1}$ correspondant aux poids appartenant au groupe g.

3.2.10 Implémentation de la procédure d'évidence

Dans ce paragraphe on va appliquer la procédure d'évidence dans l'apprentissage du PMC permettant de classifier les Iris de Fisher et on va donné les résultats numériques trouvés.

3.2.10.1 Description de la base des données

Les iris de Fisher sont des données proposées en 1933 par le statisticien Ronald Aylmer Fisher comme données de référence pour l'analyse discriminante et la classification. Il est parfois aussi appelé Iris d'Anderson du nom d'Edgar Anderson qui a collecté ces données afin de quantifier les variations de morphologie des fleurs d'iris de trois espèces.
Le jeu des données comprend 50 échantillons de chacune des trois espèces d'iris (Iris setosa, Iris virginica et Iris versicolor). Quatre caractéristiques ont été mesurées à partir de chaque échantillon : la longueur et la largeur des sépales et des pétales, en centimètres.

Figure 3.2 – Iris de Ficher

3.2.10.2 Codage et architecture du réseau

Généralement, dans un problème de classification à k classes, la couche de sortie contient souvent le nombre de classes neurones, chaque neurone représente une classe.
Dans notre cas, on a besoin de trois neurones dans la couche de sortie et nous allons utiliser le codage suivant :
La sortie désirée correspondante à la classa a) (Setosa) est : $(1, 0, 0)^t$
La sortie désirée correspondante à la classa b) (Virginica) est : $(0, 1, 0)^t$
La sortie désirée correspondante à la classa c) (Versicolor) est : $(0, 0, 1)^t$
Comme on s'intéresse aux quatre caractéristiques, alors chaque exemple est un vecteur de quatre composantes, et par suite la couche d'entrée doit contenir quatre neurones.
Concernant la couche cachée plusieurs travaux ont été effectués dans le cadre d'optimisation d'architecture, ils ont trouvé que le nombre optimal de neurones dans la couche cachée pour ce problème est 10 neurones.
Nous allons utiliser donc un PMC avec une seule couche cachée contient 10 neurones, 4 neurones dans la couche d'entrée et 3 neurones dans la couche de sortie. La fonction d'activation utilisée pour tout le réseau est la fonction sigmoide.

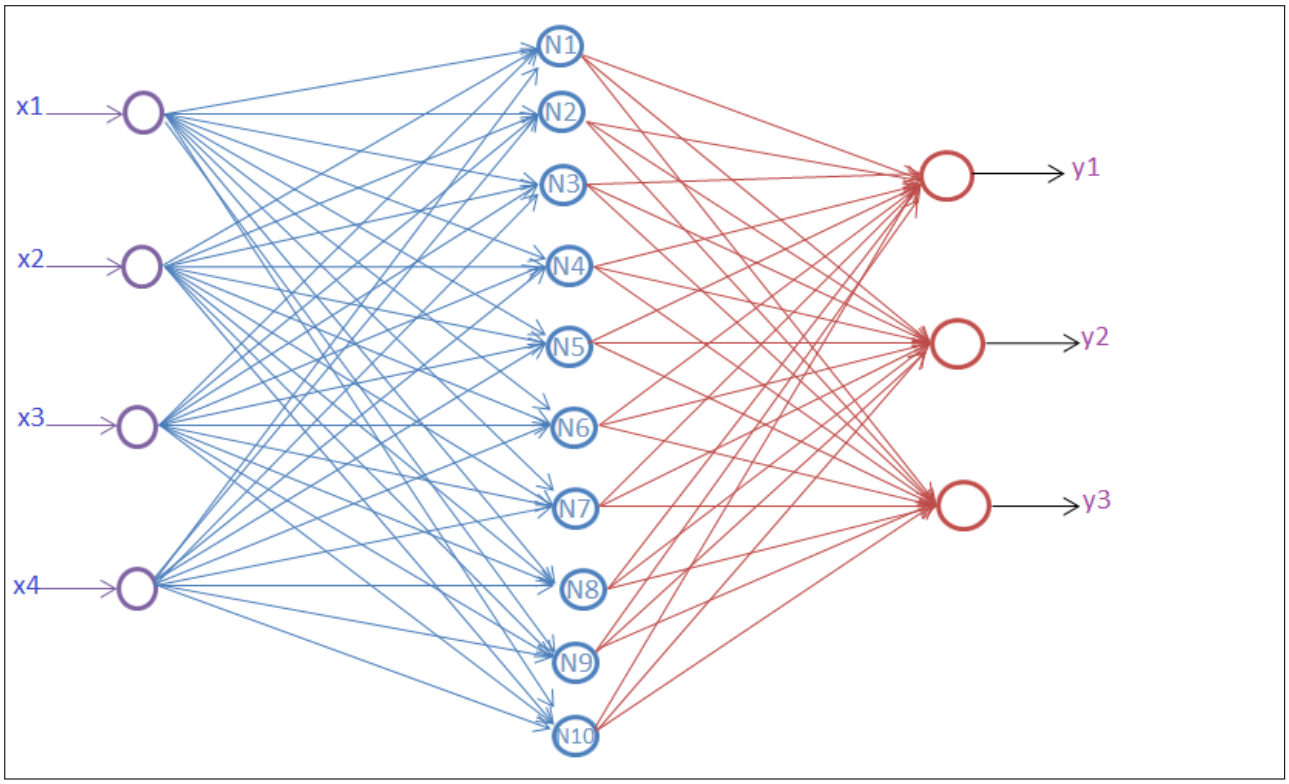

Figure 3.3 – Architecture du réseau utilisé

3.2.10.3 Normalisation et prétraitement des données

Les données collectées sont des mesures en centimètres, alors ne sont pas forcément comprises entre 0 et 1, donc pour les entrer au réseau il faut tout d'abord les normaliser.
La normalisation des données se fait suivant la formule :

$$x^k_{i\,new} = \frac{|x^k_i - min(x_i)|}{max(x_i) - min(x_i)} \tag{3.45}$$

3.2.10.4 Implémentations et résultats numériques

Afin de bien profiter de la base des donnés nous avons utiliser 75 exemple pour l'apprentissage et 75 exemple pour le test avec 25 exemple pour chaque type de fleur dans les deux base.

	Base d'apprentissage	Base test
Setosa	25	25
Virginica	25	25
Versicolor	25	25
Total	75	75

On va réordonner la base de données de telle sorte que :

- Les exemples de 1 à 75 seront les données d'apprentissage.
- Les exemples de 76 à 150 seront les données du test.

On considérant comme mesure d'erreur l'erreur quadratique :

- Sur l'ensemble d'apprentissage :

$$E_{app} = \frac{1}{2} \sum_{i=1}^{75} \sum_{j=1}^{3} (y_{ij} - d_{ij})^2$$

- Sur l'ensemble de test :

$$E_{test} = \frac{1}{2} \sum_{i=76}^{150} \sum_{j=1}^{3} (y_{ij} - d_{ij})^2$$

Après avoir tourné l'algorithme de rétropropagation du gradient pour le perceptron multicouche dont l'architecture est décrite avant, en utilisant $\varepsilon = 0.25$ comme pas d'apprentissage, on a aboutit à les courbes représentées dans la figure 3.4.

Commentaires

- On remarque que l'erreur sur l'ensemble d'apprentissage est entrain de diminuer au cours des itérations jusqu'à l'atteinte de l'erreur minimale désirée, cependant l'erreur sur l'ensemble du test est entrain de diminuer jusqu'à un certain moment ou elle commence à augmenter, ce qui signifie que la fonction apprise par le réseau apprend les particularités de la base d'apprentissage au détriment du modèle sous-jacent.
- On conclut qu'il est possible d'obtenir une erreur aussi petite que l'on veut sur l'ensemble d'apprentissage Cependant, le but de l'apprentissage n'est pas d'apprendre exactement la base d'apprentissage, mais le modèle sous-jacent qui a servi à engendrer les données.

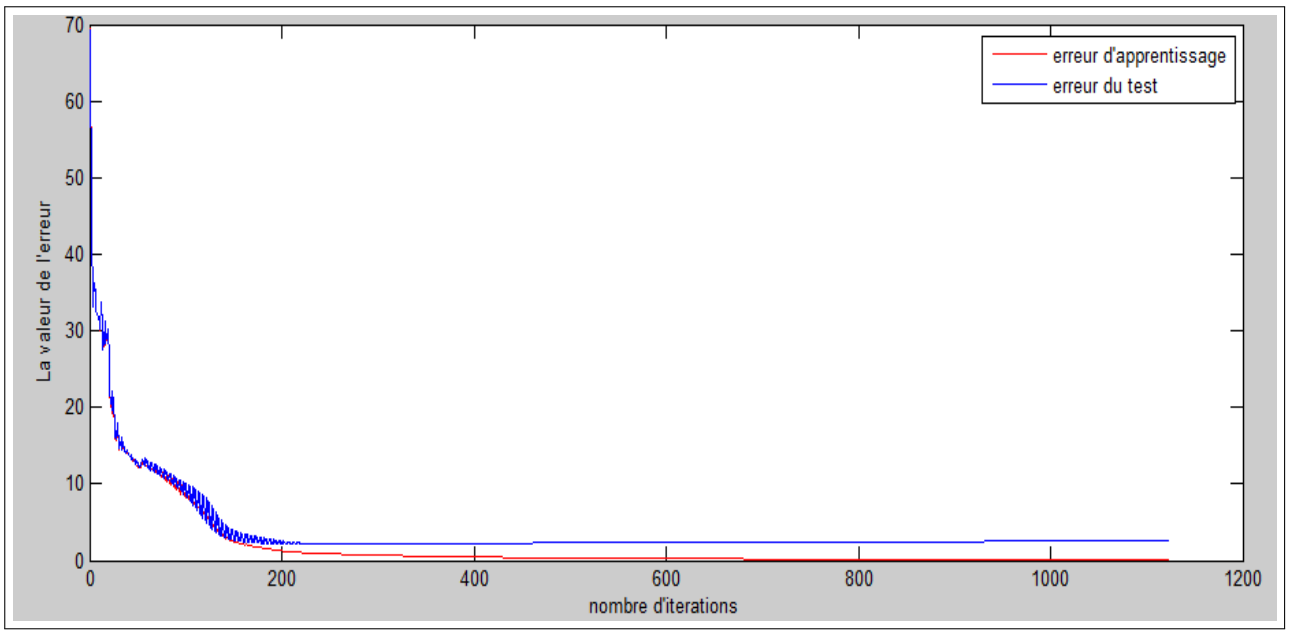

Figure 3.4 – La variation de l'erreur au cours des itérations

L'application de la procédure d'évidence

Après avoir tourné l'algorithme d'évidence avec les données précédentes, et en utilisant l'approximation de Gauss-Newton pour le calcul de la matrice hessienne de l'erreur quadratique [6], on a aboutit à des courbes suivantes :

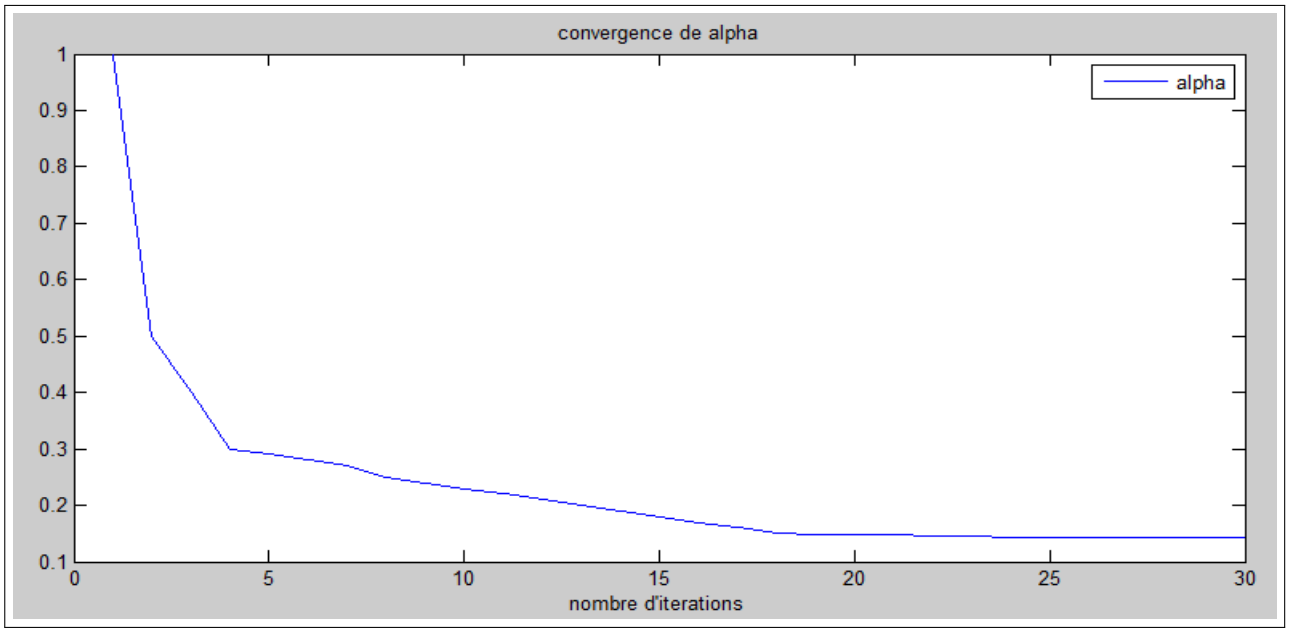

Figure 3.5 – La variation de alpha au cours des itérations

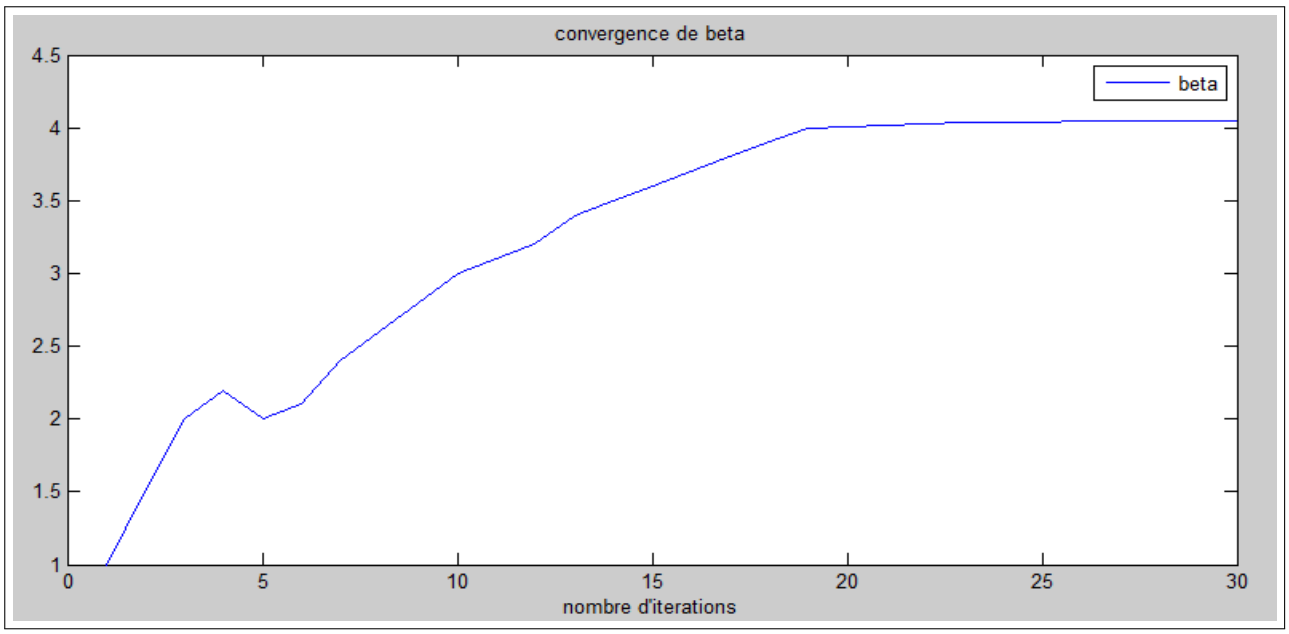

Figure 3.6 – La variation de beta au cours des itérations

Commentaires

- D'après Bishop γ mesure le nombre effectif de poids dont les valeurs sont contrôlées par les données plutôt que par l'apriori. Ces poids sont appelés des paramètres bien déterminés.

- Dans notre cas on a au totalité 83 paramètres dans le réseau, et la figure 3.7 montre que γ converge vers 64.5, ce qui signifie que 64 paramètres sont bien déterminés.

- Après 30 itérations, nous avons obtenu 4 exemples mal classés parmi tous les 150 exemples, ce qui montre la performance de l'approche bayésien pour les réseaux de neurones.

	Nombre d'exemples	Nombre d'exemples mal classés	Taux de reconnaissance (%)
Données d'apprentissage	75	0	100
Données du test	75	4	94.66

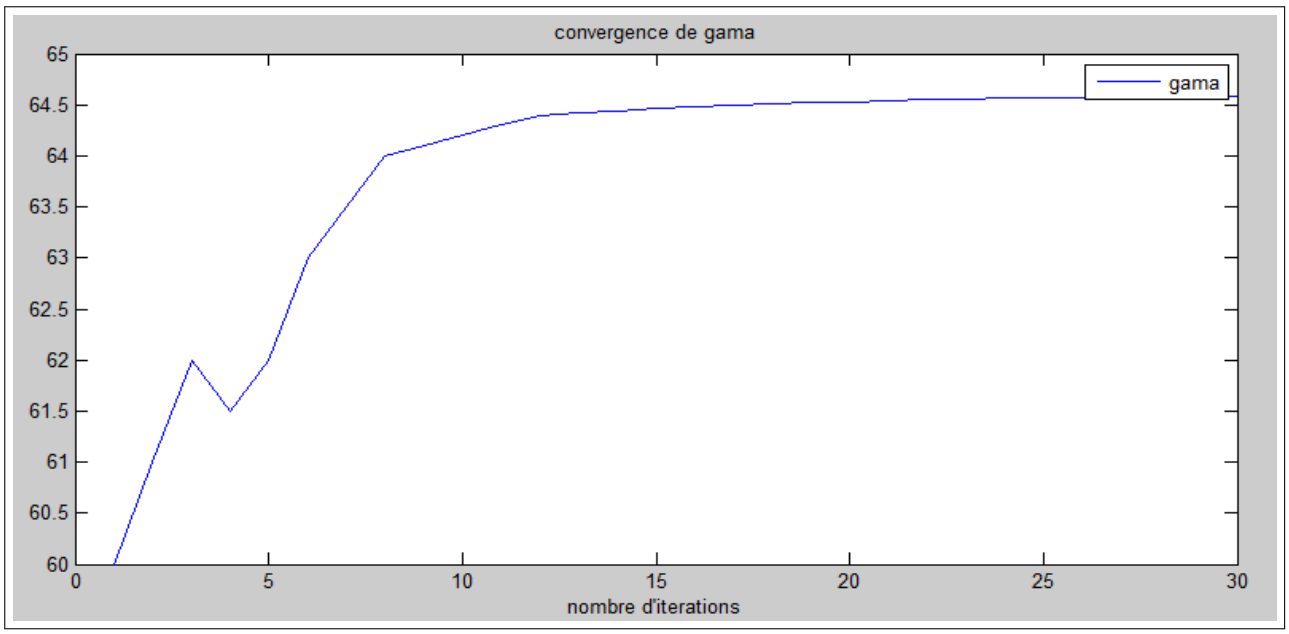

Figure 3.7 – La variation de gama au cours des itérations

3.3 Intégration des méthodes MCMC dans l'apprentissage des RNA

Les méthodes Monte Carlo par chaînes de Markov (MCMC), s'inscrivent dans le cadre du formalisme Bayésien. On peut dire que l'application de ces méthodes à l'apprentissage des réseaux de neurones peut apporter plusieurs avantages, tout d'abord, l'introduction de connaissances a priori est susceptible d'améliorer l'estimation des paramètres. De plus, le formalisme Bayésien permet une analyse complète des incertitudes et des probabilités des données.

Le processus d'apprentissage bayésienne commence par la définition du modèle, et la distribution a priori $p(\theta)$ pour le modèle paramétrique. La distribution a priori exprime les connaissance sur les valeurs des paramètres avant que les données n'aient été observées. Après avoir observé de nouvelles données $D = \{(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), ..., (x^{(n)}, y^{(n)})\}$, la distribution a priori est transformé en une distribution a posteriori via le théorème de Bayes :

$$p(\theta|D) = \frac{p(D|\theta)p\theta}{p(D)} \propto L(\theta|D)p(\theta) \tag{3.46}$$

Où la fonction vraisemblance $L(\theta|D)$ donne la probabilité que les données observées fonctionnent selon les paramètres inconnu du modèle. Dans le cas de points de données indépendants et échangeables, nous obtenons

$$L(\theta|D) = \prod_{i=1}^{n} p(y^{(i)}|x^{(i)}, \theta), \tag{3.47}$$

Où n est le nombre nombre des données.

Nous considérons un réseau PMC avec des poids w. PMC prend le vecteur d'entrée x et génère

une sortie $o = f(x, w)$ qui représente la fonction de régression. Dans les problèmes de régression, on suppose généralement que la distribution de Les données cibles peuvent être décrites par une fonction déterministe des entrées, corrompues par un bruit Gaussien additif avec une matrice de covariance constante. La densité de probabilité pour une cible y est alors

$$p(y|x, w, \Sigma) = (2\pi)^{-d/2}|\Sigma|^{-1/2}exp\Big(-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)\Big) \tag{3.48}$$

Où Σ est la matrice variance-covariance.
En utilisant différentes paramétrisations de la matrice de covariance, nous faisons différentes hypothèses sur le bruit. Les paramètres qui ont été généralement utilisés sont des matrices de covariance diagonales constantes ($\sigma^2 I$) et diagonales ($diag(\sigma_1^2, ..., \sigma_d^2)$). Ceux-ci correspondent à l'hypothèse d'un bruit indépendant entre les sorties, avec un niveau de bruit commun ou des niveaux de bruit indépendants, respectivement. Les densités de probabilité pour une cible y sont données respectivement par

$$p(y|x, w, \sigma^2) = (2\pi\sigma^2)^{-1/2}exp\bigg(-\frac{1}{2\sigma^2}(y-o)^2\bigg) \tag{3.49}$$

$$p(y_j|x, w, \sigma_j^2) = (2\pi\sigma_j^2)^{-1/2}exp\bigg(-\frac{1}{2\sigma_j^2}(y_j-o_j)^2\bigg) \tag{3.50}$$

Notez que l'utilisation d'une covariance diagonale constante avec une approche de maximum de vraisemblance correspond à la minimisation de l'erreur de somme des carrés.
Un préalable conjugué pratique, qui nous permet de définir facilement notre connaissance préalable (ou ignorance) de la variance du bruit, est donné en général par

$$p(\sigma^2) = Inv - Gamma(\sigma^2|\nu_0, \sigma_0^2) \tag{3.51}$$

$$\propto \big(\sigma^2\big)^{((-\nu_0/2)+1)}exp\Big(-\frac{1}{2}\nu_0\sigma_0^2\sigma^{-2}\Big) \tag{3.52}$$

Où $Inv - Gamma$ est l'inverse de la distribution de la loi Gamma avec ν degré de liberté.
De cette manière, la connaissance a priori à propos de σ est équivalente à ν_0 mesures a priori ayant une variance commune σ_0. Parfois, les distributions antérieures sont spécifiées en termes de précisions correspondantes $\tau = \sigma^{-2}$, avec des distributions Gamma données. La distribution gamma dans l'analyse bayésienne a été discutée, par plusieurs mathématiciens, par exemple, par Box et Tiao en 1973. La distribution gamma en rapport avec les PML a été discutée par Neal en 1996 [36].
Pour prédire une nouvelle sortie y^{n+1} pour une nouvelle entrée x^{n+1}, la distribution prédictive est obtenue en intégrant les prédictions du modèle par rapport à la distribution a posteriori des paramètres du modèle

$$p(y^{(n+1)}|x^{(n+1)}, D) = \int p(y^{(n+1)}|x^{(n+1)}, \theta)p(\theta|D)d\theta \tag{3.53}$$

Les attentes peuvent être évaluées par rapport à la distribution a posteriori des paramètres, par exemple pour $y^{(n+1)}$

$$\hat{y}^{(n+1)} = \int f(x^{(n+1)}, w)p(\theta|D)d\theta \tag{3.54}$$

Les distributions a posteriori dans le cas des PMC sont généralement très complexes et les intégrales sont très difficiles à évaluer. Neal [36] a introduit la mise en œuvre de l'implantation bayésienne pour les réseaux de neurones par Les méthodes Monte Carlo par chaînes de Markov (MCMC). Dans MCMC, les intégrations requises par l'approche bayésienne sont approchées numériquement à l'aide d'un échantillon de valeurs tirées de la distribution a posteriori des paramètres.
Par exemple (3.54) est approximé par

$$\widehat{y}^{(n+1)} \approx \sum_{t=1}^{N} f(x^{(n+1)}, w^{(t)}) \tag{3.55}$$

Où $w^{(t)}$ sont des échantillons des poids.
Dans les méthodes MCMC, les échantillons sont générés à l'aide d'une chaîne de Markov qui a la distribution a posteriori souhaitée comme distribution limite.

Conclusion

Les méthodes bayésiennes pour l'apprentissage des réseaux de neurones peuvent apporter plusieurs avantages, car il n'est pas nécessaire de limiter la taille du réseau pour éviter le sur ajustement, et que le nombre de neurones cachés peut tendre vers l'infini, le seul facteur qui doit limiter la taille du réseau est la capacité des ordinateurs utilisés et le temps disponible pour effectuer les calculs nécessaires, mais comme les paramètres utilisés sont issus d'une distribution de probabilité, il est nécessaire pour connaître un paramètre de calculer des intégrales faisant intervenir les distributions des autres paramètres. Il est, en général, impossible de calculer ces intégrales analytiquement, et plusieurs approches ont été proposées pour effectuer ces calculs.

CHAPITRE 4

APPLICATION DES RÉSEAUX DE NEURONES BAYŚIENS À LA RECONNAISSANCE DES VISAGES

Introduction

La reconnaissance faciale est un domaine de recherche riche en publications en raison de ses nombreuses applications. Parmi celles-ci les systèmes de contrôle d'accès, de surveillance, d'interaction homme machine dans des applications multimédia de divertissement ou à finalités éducatives, de vérification de carte de crédit et bien d'autres.
Une première distinction au sein des approches existantes peut se faire entre les méthodes à échantillons multiples et les méthodes à échantillon unique. Dans la première catégorie, le training set est composé de plusieurs prises de vue par sujet et ces dernières peuvent inclure des variations diverses telles que l'illumination, l'âge, l'orientation du visage ou l'expression. La seconde catégorie, quant à elle, s'attarde sur des problèmes de reconnaissance où il n'y a qu'un seul échantillon disponible par personne pour construire le reconnaisseur de visage.

4.1 Reconnaissance faciale comme technique biométrique

La biométrie consiste à identifier une personne à partir d'une ou de plusieurs caractéristiques physiologiques (empreintes digitales, visage, iris, contour de la main, etc.), ou comportementales (signature, démarche, etc.). Etymologiquement, la biométrie humaine est synonyme d'anthropologie physique. Une autre définition de la biométrie est donnée par Roethenbaugh : "La biométrie s'applique à des particularités ou des caractères humains uniques en leur genre et mesurables, permettant de reconnaître ou de vérifier automatiquement l'identité".
Les systèmes biométriques sont de plus en plus utilisés depuis quelques années. L'apparition de l'ordinateur et sa capacité à traiter et à stocker les données ont permis la création des systèmes biométriques informatisés. Il existe plusieurs caractéristiques physiques uniques pour un individu, ce qui explique la diversité des systèmes appliquant la biométrie, selon que l'on prend en compte :

- L'empreinte digitale
- La géométrie de la main
- L'iris
- La rétine

- Le visage
- La reconnaissance vocale
- La dynamique des signatures
- ...

Nous insisterons sur la reconnaissance faciale parmi les autres techniques biométriques, car elle constitue l'objectif de ce travail, et nous exposerons les difficultés majeures liées à la reconnaissance de visage, et qui font toujours l'objet de recherche par la communauté scientifique.

4.1.1 Systèmes biométriques basés sur la reconnaissance de visage

La reconnaissance automatique de visage s'effectue en trois étapes principales (voir figure 4.1) :

1) Détection de visages

2) Extraction et normalisation des caractéristiques du visage

3) Identification et/ou vérification

Certaines techniques de traitements d'images peuvent être communes à plusieurs étapes. Par exemple, l'extraction des caractéristiques faciales (yeux, nez, bouche) est utilisée aussi bien pour la détection que pour l'identification de visages. Par ailleurs, les étapes de détection de visage et d'extraction de caractéristiques peuvent être exécutées simultanément. Cela dépend notamment de la nature de l'application, de la taille de la base d'apprentissage, et des conditions de prise de vue (bruit, occultation, etc.). Enfin, les techniques de traitement utilisées dans chaque étape sont très critiques pour les applications biométriques, et doivent, par conséquence, être optimisées pour améliorer les performances du système global.

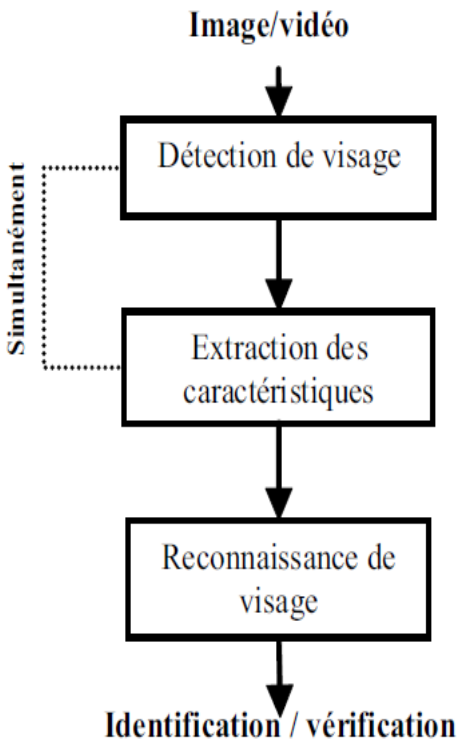

Figure 4.1 – Les étapes de la reconnaissance du visage.

Dans ce qui suit nous allons détailler chaque étape du système de reconnaissance faciale, et nous présenterons les principales difficultés rencontrées.

4.1.1.1 Détection du visage

L'efficacité des systèmes biométriques basés sur l'authentification de visage dépend essentiellement de la méthode utilisée pour localiser le visage dans l'image. Dans la littérature scientifique, le problème de localisation de visages est aussi désigné par la terminologie "détection de visages".
Plusieurs travaux de recherches ont été effectués dans ce domaine. Ils ont donné lieu au développement d'une multitude de techniques allant de la simple détection du visage, à la localisation précise des régions caractéristiques du visage, tels que les yeux, le nez, les narines, les sourcils, la bouche, les lèvres, les oreilles, etc. Cependant, les solutions proposées jusqu'à maintenant sont loin d'être satisfaisantes car elles fonctionnent uniquement dans des environnements contrôlés, et par conséquent elles ne gèrent pas la variabilité des conditions d'acquisition de la vie quotidienne, notamment :

- **La pose :** où les images d'un visage changent en fonction de l'orientation de ce dernier (frontal, 45 degrés, profil).

- **La présence ou absence des composantes structurales :** les caractéristiques faciales tels que la barbe, la moustache, et les lunettes causent une grande variabilité des composantes structurales du visage, notamment au niveau de la forme, de la couleur, et de la taille.

- **Les occultations :** les visages peuvent être partiellement occultés par d'autres objets. En effet, dans une image contenant un groupe de personnes par exemple, des visages peuvent partiellement masquer d'autres visages.

- **Les conditions d'illumination :** des facteurs tels que l'éclairage (distribution de la source de lumière, son intensité, son spectre) et les caractéristiques de l'appareil photographique affectent l'aspect d'un visage dans l'image acquise.

4.1.1.2 Extraction des caractéristiques du visage

L'extraction des caractéristiques telles que les yeux, le nez, la bouche est une étape prétraitement nécessaire à la reconnaissance faciale. On peut distinguer deux pratiques différentes : la première repose sur l'extraction de régions entières du visage, elle est souvent implémentée avec une approche globale de reconnaissance de visage. La deuxième pratique extrait des points particuliers des différentes régions caractéristiques du visage, tels que les coins des yeux, de la bouche et du nez. Elle est utilisée avec une méthode locale de reconnaissance et aussi pour l'estimation de la pose du visage.

4.1.1.3 Reconnaissance du visage

Le module de reconnaissance exploite les caractéristiques du visage ainsi extraites pour créer une signature numérique qu'il stocke dans une base de données. Ainsi, à chaque visage de la base est associée une signature unique qui caractérise la personne correspondante. La reconnaissance d'un visage requête est obtenue par l'extraction de la signature requête correspondante et sa mise en correspondance avec la signature la plus proche dans la base de données. La reconnaissance dépend du mode de comparaison utilisé : vérification ou identification.

4.1.2 Principales difficultés de la reconnaissance du visage

Pour le cerveau humain, le processus de la reconnaissance de visages est une tâche visuelle de haut niveau. Bien que les êtres humains puissent détecter et identifier des visages dans une scène sans beaucoup de peine, construire un système automatique qui accomplit de telles tâches représente un sérieux défi. Ce défi est d'autant plus grand lorsque les conditions d'acquisition des images sont très variables. Il existe deux types de variations associées aux images de visages : inter et intra sujet. La variation inter-sujet est limitée à cause de la ressemblance physique entre les individus. Par contre la variation intra-sujet est plus vaste. Elle peut être attribuée à plusieurs facteurs que nous analysons ci-dessous.

4.1.2.1 Changement d'illumination

L'apparence d'un visage dans une image varie énormément en fonction de l'illumination de la scène lors de la prise de vue (voir figure4.2). Les variations d'éclairage rendent la tâche de reconnaissance de visage très difficile. En effet, le changement d'apparence d'un visage du à l'illumination, se révèle parfois plus critique que la différence physique entre les individus, et peut entraîner une mauvaise classification des images d'entrée. L'identification de visage dans un environnement non contrôlé reste donc un domaine de recherche ouvert. Les évaluations FRVT ont révélé que le problème de variation d'illumination constitue un défi majeur pour la reconnaissance faciale.

Figure 4.2 – Exemple de variation d'éclairage.

4.1.2.2 Variation de pose

Le taux de reconnaissance de visage baisse considérablement quand des variations de pose sont présentes dans les images. Cette difficulté a été démontrée par des tests d'évaluation élaborés sur les bases FERET et FRVT. La variation de pose est considérée comme un problème majeur pour les systèmes de reconnaissance faciale. Quand le visage est de profil dans le plan image (orientation $< 30^o$), il peut être normalisé en détectant au moins deux traits faciaux (passant par les yeux). Cependant, lorsque la rotation est supérieure à 30^o, la normalisation géométrique n'est plus possible (voir figure 4.3).

4.1.2.3 Expressions faciales

Un autre facteur qui affecte l'apparence du visage est l'expression faciale(voir figure 4.4). La déformation du visage qui est due aux expressions faciales est localisée principalement sur la

Figure 4.3 – Exemples de variation de poses.

partie inférieure du visage. L'information faciale se situant dans la partie supérieure du visage reste quasi invariable. Elle est généralement suffisante pour effectuer une identification. Toutefois, étant donné que l'expression faciale modifie l'aspect du visage, elle entraîne forcément une diminution du taux de reconnaissance. L'identification de visage avec expression faciale est un problème difficile qui est toujours d'actualité et qui reste non résolu. L'information temporelle fournit une connaissance additionnelle significative qui peut être utilisée pour résoudre ce problème.

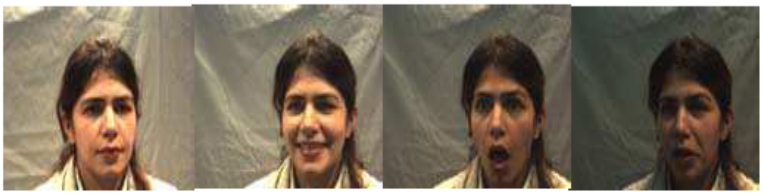

Figure 4.4 – Exemples de variation d'expressions.

4.1.2.4 Présence ou absence des composants structurels

La présence des composants structurels telle que la barbe, la moustache, ou bien les lunettes peut modifier énormément les caractéristiques faciales telles que la forme, la couleur, ou la taille du visage. De plus, ces composants peuvent cacher les caractéristiques faciales de base causant ainsi une défaillance du système de reconnaissance. Par exemple, des lunettes opaques ne permettent pas de bien distinguer la forme et la couleur des yeux, et une moustache ou une barbe modifie la forme du visage.

4.1.2.5 Occultations partielles

Le visage peut être partiellement masqué par des objets dans la scène, ou par le port d'accessoire tels que lunettes, écharpe... Dans le contexte de la biométrie, les systèmes proposés doivent être non intrusifs c'est-à-dire qu'on ne doit pas compter sur une coopération active du sujet. Par conséquent, il est important de savoir reconnaître des visages partiellement occultés.

4.2 Réduction de la dimensionnalité

Cette tâche apparaît dans le schéma global d'un processus de reconnaissance de formes, comme la sélection/extraction de variables. Cette phase est importante et incontournable, car elle conditionne le processus de classification. Ainsi, deux approches, illustrées à la figure 4.5, sont principalement utilisées pour réduire la dimension :

- Une réduction basée sur une sélection de caractéristiques qui consiste à sélectionner les caractéristiques les plus pertinentes à partir de l'ensemble de données des variables décrivant le phénomène étudié.

- Une réduction basée sur une transformation des données appelée aussi une extraction de caractéristiques et qui consiste à remplacer l'ensemble initial des données par un nouvel ensemble réduit, construit à partir de l'ensemble initial de caractéristiques.

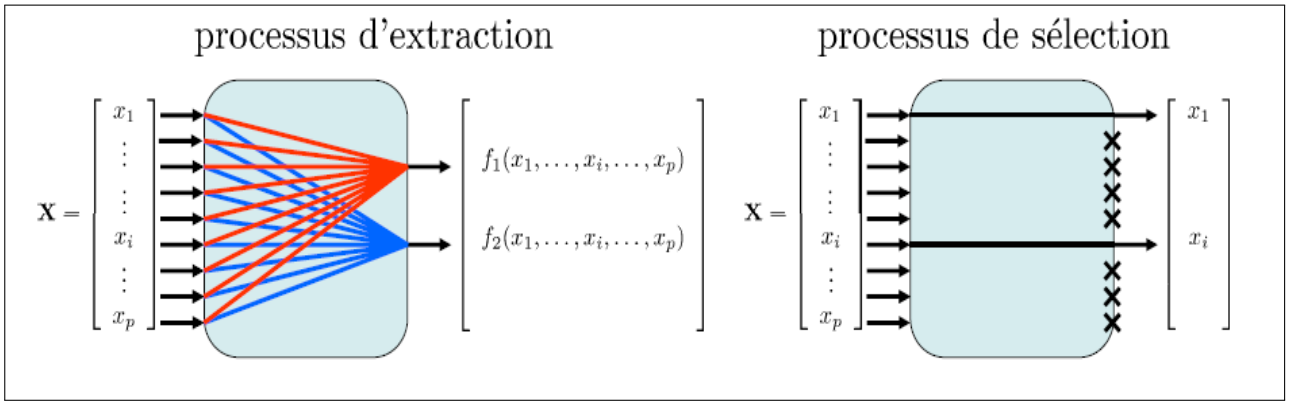

Figure 4.5 – Extraction des caractéristiques et sélection de variables

4.2.1 Réduction basée sur une transformation de données

4.2.1.1 Analyse en composantes principales ACP

L'Analyse en Composantes principales (ACP) fait partie du groupe des méthodes descriptives multidimensionnelles appelées méthodes factorielles.
L'ACP est une technique qui permet de trouver des espaces de dimensions plus petites dans lesquels il est possible d'observer au mieux les individus. Sa démarche essentielle consiste à transformer les variables quantitatives initiales, plus ou moins corrélées entre elles, en des variables quantitatives, non corrélées, combinaisons linéaires des variables initiales et appelées composantes principales. Les composantes principales sont donc de nouvelles variables indépendantes, combinaisons linéaires des variables initiales, possédant une variance maximale.
Globalement l'ACP consiste à rechercher la direction suivant laquelle le nuage de points des observations s'étire au maximum. A cette direction correspond la première composante principale. La seconde composante principale est déterminée de telle sorte qu'elle soit la plus indépendante possible de la première ; elle est donc perpendiculaire à celle-ci. Ces deux composantes forment le premier plan principal. Cette opération est réitérée de manière à trouver toutes les composantes principales expliquant le maximum de variance.
La figure 4.6 montre à gauche, un exemple de données 3D qui se trouvent dans un plan 2D et à droite les deux premières composantes principales sur ces données.

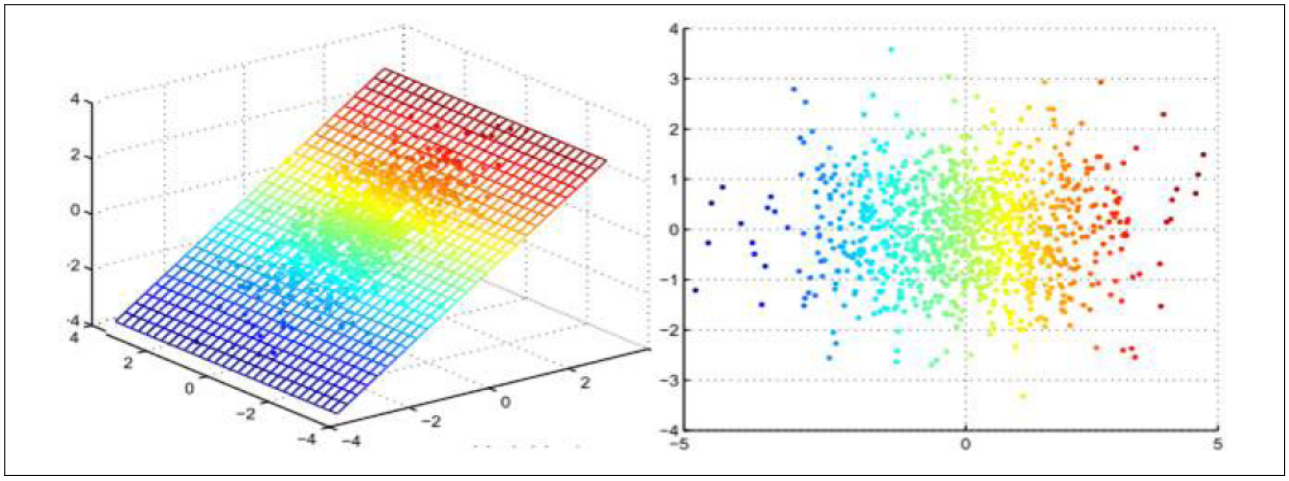

Figure 4.6 – ACP sur des données linéaires

Supposons que nous ayons un ensemble de données $X = \{x_1, x_2, ..., x_M\}$ composé de M observations où chaque observation $x_i = (x_{i1}, x_{i2}, ..., x_{iN})$ est composée de N caractéristiques.
X est associé à une matrice de données A de taille $N \times M$ où chaque colonne représente une caractéristique. En pratique, le calcul de l'ACP pour la matrice X revient à réaliser les opérations ci-dessous afin de trouver les composantes principales :

1. Calculer le vecteur $\mu = (\mu_1, \mu_2, ..., \mu_m)^T$ qui représente le vecteur moyen où μ_i est la moyenne de la ième composante des données.

2. Calculer la matrice T en soustrayant le vecteur moyen à toutes les colonnes de A dans le but d'obtenir des données centrées.

3. Calculer la matrice S (de taille $N \times N$) de covariance de T avec $(S = T.T^T)$.

4. Calculer la matrice U (de taille$N \times N$) qui est composée des coordonnées des vecteurs propres u_j de S triés par ordre décroissant des modules des valeurs propres λ_j (la première colonne de U est le vecteur propre qui correspond à la plus grande valeur propre)

5. Garder les R premières colonnes de U pour former la matrice U : $(N \times R)$ qui représente les R premières composantes principales.

L'ACP étant une méthode de réduction de dimension, il est important de savoir qu'elle ne peut pas retenir la totalité de l'information contenue dans le nuage de points initial. Enfin, l'ACP prend uniquement en compte les dépendances linéaires entre les variables et ne peut donc pas fournir une projection pertinente pour une distribution non-linéaire de points.
La figure 4.7 montre à gauche, un exemple de données non-linéaires (non réparties dans un plan) et à droite le résultat de leur projection dans un plan générale par les deux premières composantes principales calculées sur ces données.

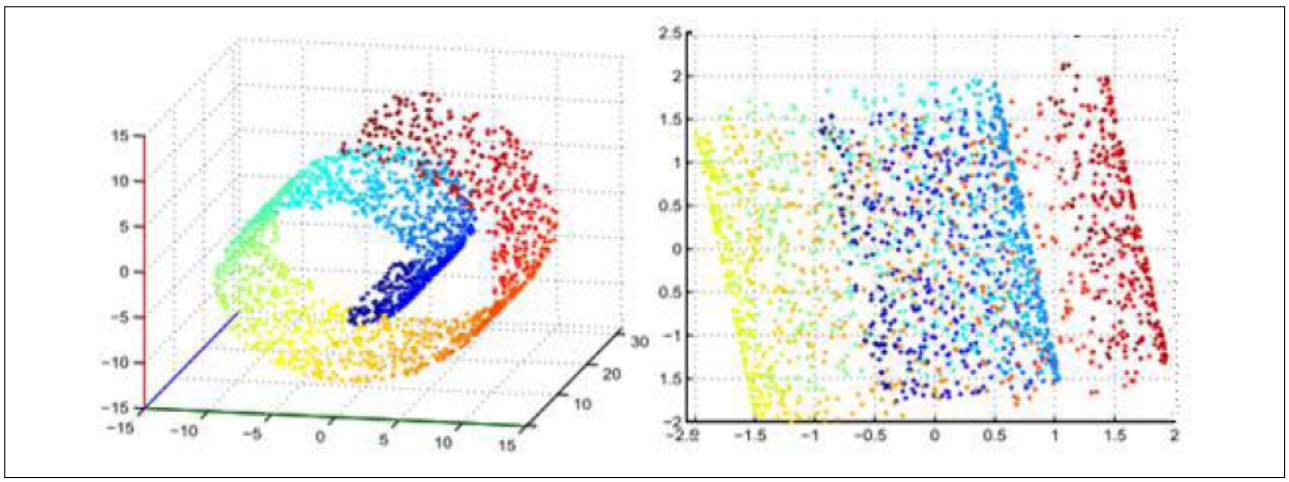

Figure 4.7 – ACP sur des données non linéaires

4.2.1.2 Analyse Linéaire Discriminante ALD

L'analyse linéaire discriminante, appelée aussi analyse discriminante linéaire de Fisher, est une méthode de réduction du nombre de dimensions proposée par Fisher en 1936 (Fisher[1936]). Cette méthode s'applique lorsque les classes des individus sont connues. L'idée de Fisher a été de créer une méthode pour choisir entre les combinaisons linéaires des variables celles qui maximisent l'homogénéité de chaque classe. En d'autres termes, cette méthode consiste à chercher un espace vectoriel de faible dimension qui maximise la variance interclasse.

4.3 Bases de données utilisées

Pour tester les performances du système de reconnaissance du visage par les réseaux de neurones bayésiens, nous avons utilisé des bases d'images standards qui contiennent un nombre précis des sujets chacun de ces derniers contient un nombre fixe d'images de même type et de même résolution.
Le choix de ces base d'image n'est pas aléatoire, car les images présentent des différentes variations telles que : Le changement des expressions, d'illumination et inclination de visages.
Dans notre projet on va se limiter sur deux bases : ORL et Yale.

4.3.1 Base ORL

ORL est une base des images standard contenant 40 sujets, chaque sujet contient 10 image de type pgm avec une résolution de 112x92 pixels, Pour certains sujets, les images ont été prises à des moments différents, en faisant varier l'éclairage, les expressions faciales et les détails du visage (lunettes ou pas de lunettes).

4.3.2 Base Yale

Yale est une base qui contient 15 sujets avec 11 images pour chacun, de type gif avec résolution de 320 x 243 pixels, ces images présentent des variations d'expression et de l'illumination puisqu'elles sont prises sur trois angles d'éclairage (Figure 4.9).

Figure 4.8 – Exemples de visage d'un seul sujet de la base ORL

4.4 Description du visage par Local Binary Patterns LBP

4.4.1 Motifs binaires locaux LBP

Les motifs binaires locaux ont initialement été proposés par Ojala en 1996 afin de caractériser les textures présentes dans des images en niveaux de gris. Ils consistent à attribuer à chaque pixel P de l'image $I(i,j)$ à analyser, une valeur caractérisant le motif local autour de ce pixel. Ces valeurs sont calculées en comparant le niveau de gris du pixel central P aux valeurs des niveaux de gris des pixels voisins.
Le concept du LBP est simple, il propose d'assigner un code binaire à un pixel en fonction de son voisinage. Ce code décrivant la texture locale d'une région est calculé par seuillage d'un voisinage avec le niveau de gris du pixel central.
Afin de générer un motif binaire, tous les voisins prendront alors une valeur "1" si leur valeur est supérieure ou égale au pixel courant et "0" autrement (Figure 4.10). Les pixels de ce motif binaire sont alors multipliés par des poids et sommés afin d'obtenir un code LBP du pixel courant. On obtient donc pour toute l'image, des pixels dont l'intensité se situe entre 0 et 255 comme dans une image à 8 bits ordinaire. Plutôt que de décrire l'image par la séquence des motifs LBP, on peut choisir comme descripteur de texture un histogramme de dimension 255.

Pour calculer un code LBP dans un voisinage de P pixels, dans un rayon R, on compte simplement les occurrences de niveaux de gris g_p plus grands ou égaux la valeur centrale.

$$LPB_{m,R} = \sum_{i=0}^{m-1} u(g_i - g_c).2^i \tag{4.1}$$

Figure 4.9 – Exemples de quelques visages de 3 sujets de la base Yale

où $u()$ est la fonction signe et où g_i et g_c sont respectivement les niveaux de gris d'un pixel voisin et du pixel central.

$$u(x) = \begin{cases} 1 & \text{si } x \geq 0 \\ 0 & \text{autrement} \end{cases} \tag{4.2}$$

La méthode LBP s'est révélée très efficace pour la classification d'images texturées comme dans les applications de reconnaissance de visage [40] et [43].

4.4.2 LBP multi échelle

Le concept du LBP multi-échelle, est fondé sur le choix du voisinage afin de calculer un code LBP pour pouvoir traiter les textures à différentes échelles [41] et [42]. Un voisinage pour un pixel central est réparti sur un cercle et construit à partir de deux paramètres : le nombre de voisins "P" sur le cercle et un rayon "R" pour définir une distance entre un pixel central et ses voisins. Soit une texture : $T = t(g_c, g_0, ..., g_{P-1})$, g_c correspond à la valeur de niveau de gris du pixel central et g_p, avec $p = 0, ..., P-1$, correspond au niveau de gris de P pixels espacés régulièrement sur un cercle de rayon R. Si les coordonnées de g_c sont égales à $(0, 0)$, alors les coordonnées de g_p sont données par l'équation suivante :

$$\{ x_g = x_c + Rcos(2\pi p/P) \quad y_g = y_c - Rsin(2\pi p/P) \} \tag{4.3}$$

Comme nous pouvons le voir sur la Figure 4.11, les coordonnées d'un voisin ne sont pas forcément situées au centre d'un pixel. Dans ce cas, le niveau de gris est déterminé par l'intermédiaire

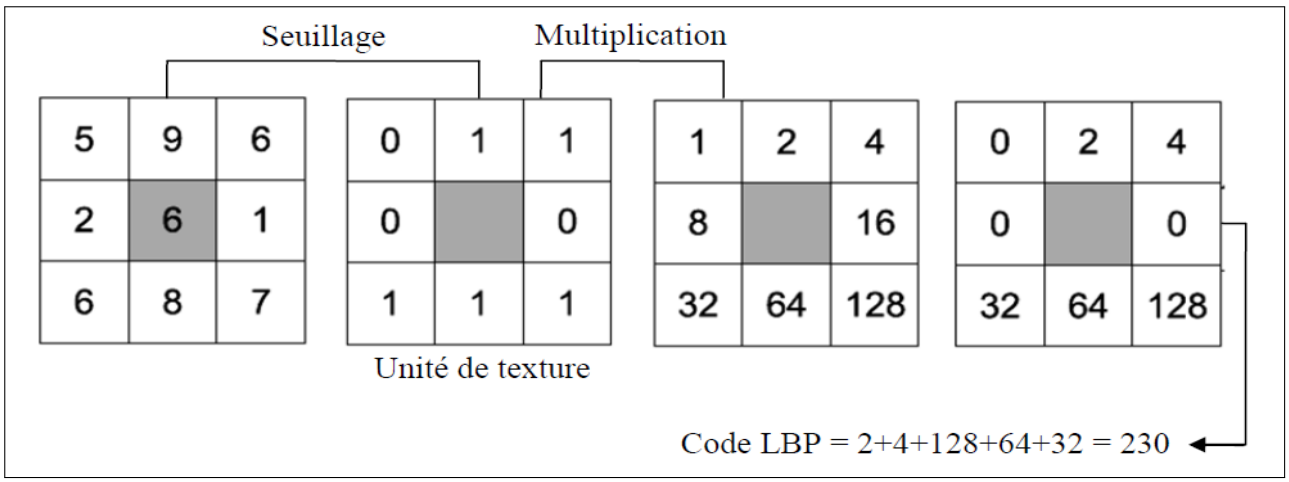

Figure 4.10 – Construction d'un motif binaire et calcul du code LBP.

d'une interpolation. La Figure 4.11, illustre différents voisinages obtenus pour différentes valeurs du couple (P, R).

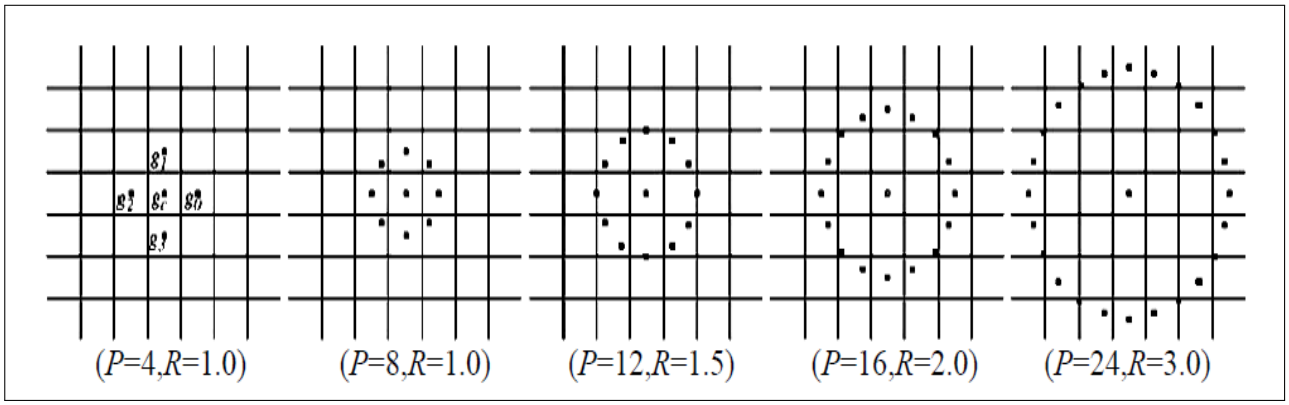

Figure 4.11 – LBP multi-échelle. Exemples de voisinages obtenus pour différentes valeurs de (P, R).

Partant de la définition de voisinage, les auteurs définissent tout d'abord un motif binaire local invariant à toute transformation monotone de l'échelle des niveaux de gris, $LBP_{P,R}$. Pour chaque pixel (x, y) $(g_c = g(x, y))$. Comme dans la méthode LBP classique, le pixel central n'est pas utilisé pour la caractérisation des textures. En effet, indépendamment du voisinage g_p, ce pixel décrit uniquement une intensité lumineuse ce qui n'est pas forcément utile [42]. Par la suite, g_c est utilisé comme un seuil de la manière suivante :

$$T = t\Big(u(g_0 - g_c), \ldots, u(g_0 - g_{P-1})\Big) \tag{4.4}$$

4.4.3 LBP invariant par rotation

Un motif binaire uniforme est définit, comme étant tout motif possédant exactement 0 ou 2 transitions (01 ou 10) dans un parcours circulaire comme le montre l'exemple de la Figure 4.12 (par exemple 10000001 ou 00011000 est un motif uniforme, mais pas 00101010). Cette notion d'uniformité est importante dans la méthode LBP pour représenter les informations de primitives structurelles comme les arêtes (coins) et les contours.
La définition de ce motif ordonné au fait que le voisinage construit soit circulairement symétrique a permis de définir un second motif, lui aussi invariant à toute transformation monotone de

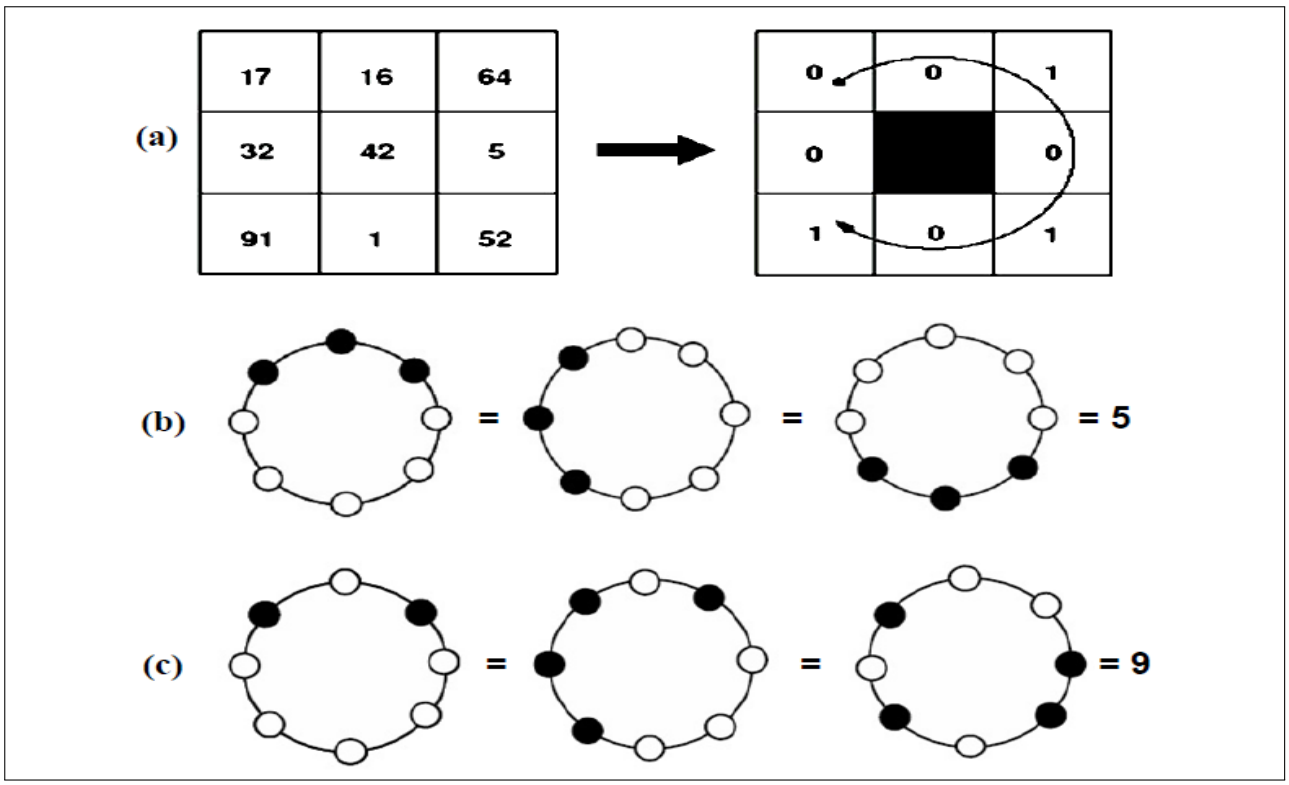

Figure 4.12 – Construction et uniformité d'un motif LBP. (a) le motif construit ici est non-uniforme. (b) et (c) Exemples de motifs respectivement uniformes et non-uniformes.

l'échelle des niveaux de gris mais également invariant à toute rotation de l'image, "$LBP^{ri}_{P,R}$". Pour tout pixel (x, y), le calcul de ce motif est donné par :

$$LBP^{ri}_{P,R}(x, y) = min\{\ ROR(LBP_{P,R}(x, y)\ ,\ i)\ \ :\ \ i \in [0, P-1]\ \} \tag{4.5}$$

où $ROR(a, i)$ correspond au résultat de i décalages circulaires successifs vers la droite des bits du nombre a (codé sur P bits). $LBP^{ri}_{P,R}$ quantifie ainsi les statistiques d'occurrence de modèles individuels invariant en rotation correspondant à certaines micro-caractéristiques de l'image.

4.4.4 Description de visage par LBP

L'opérateur LBP basique prend comme entrée un carré de 9 pixels et a pour sortie un nombre binaire 8 bits. Ce dernier est présenté à la figure 4.13. La motivation qui a poussé à utiliser cet opérateur est qu'un visage peut être vu comme un assemblage de micro-patterns dont la description par LBP est à la fois bonne, robuste face aux variations de gris et rapide à générer.
Ensuite pour inclure une information quant à la disposition spatiale des textures et éviter de se limiter à une description holistique des textures qui souffrirait des limitations connues des méthodes de ce type, l'image convertie est divisée en plusieurs sous-régions pour lesquelles autant d'histogrammes LBP seront faits. Ainsi pour 4 sous-régions, 4 histogrammes seront générales. Ces derniers seront concaténés pour former une matrice à 2 dimensions appelée 'histogramme spatialement amélioré'. Notons que les sous-régions peuvent se recouvrir et ne doivent pas nécessairement être rectangulaires.
La comparaison de deux histogrammes spatialement améliorés suppose d'une part d'établir une méthode de mesure de distance entre deux histogrammes simples et d'autre part l'utilisation de poids pour rassembler les distances obtenues pour chaque sous région. La méthode 2 en 1

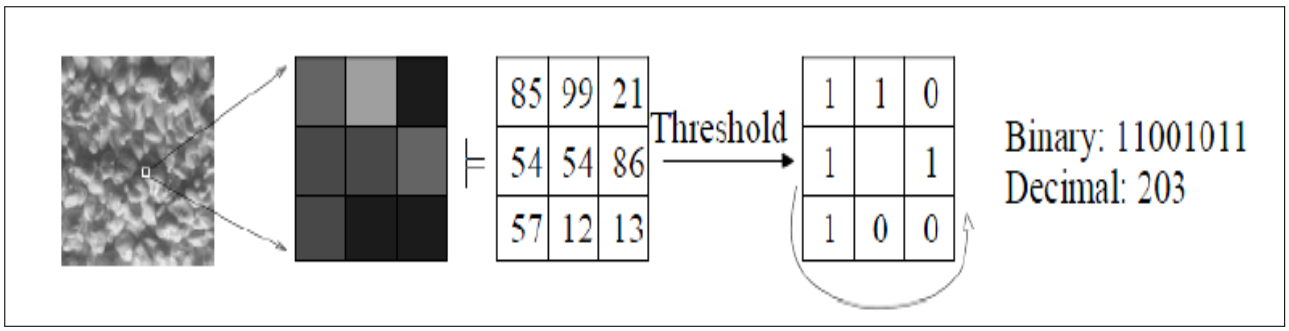

Figure 4.13 – Opérateur LBP basique. L'entrée est un carré de 9 pixels.

proposée par Ahonen et al. est la distance carré de Chi balancée :

$$\chi^2_w(x,\xi) = \sum_{j,i} w_j \frac{(x_{i,j} - \xi_{i,j})^2}{x_{i,j} + \xi_{i,j}} \tag{4.6}$$

où x et ξ sont des histogrammes spatialement améliorés normalisés à comparer, i correspond à $i$ème valeur du $j$ème sous-histogramme et w_j est le poids accordé à la sous région j.
Notons maintenant l'ensemble des éléments paramétrisables : tout d'abord le choix de l'entourage (P, R) à utiliser pour la conversion LBP, ensuite la manière dont les sous régions sont délimitées, puis le poids accordé à chacune d'elles et enfin la méthode de calcul de distance qui sera utilisée pour comparer deux histogrammes. Ces paramétrages bien effectués amélioreront les performances du classifieur final.

4.5 Description des visages par Matrice de co-occurrence

Du fait de leur richesse en information de texture, les matrices de co-occurrences sont devenues les plus connues et les plus utilisées pour extraire ces caractéristiques de textures. Elles estiment des propriétés des images relatives à des statistiques de second ordre.
Une matrice de co-occurrence mesure la probabilité d'apparition des paires de valeurs de pixels situés à une certaine distance dans l'image. Elle est basée sur le calcul de la probabilité $P(i, j, \delta, \theta)$ qui représente le nombre de fois où un pixel de niveau de couleur i apparaît à une distance relative δ d'un pixel de niveau de couleur j et selon une orientation θ donnée.

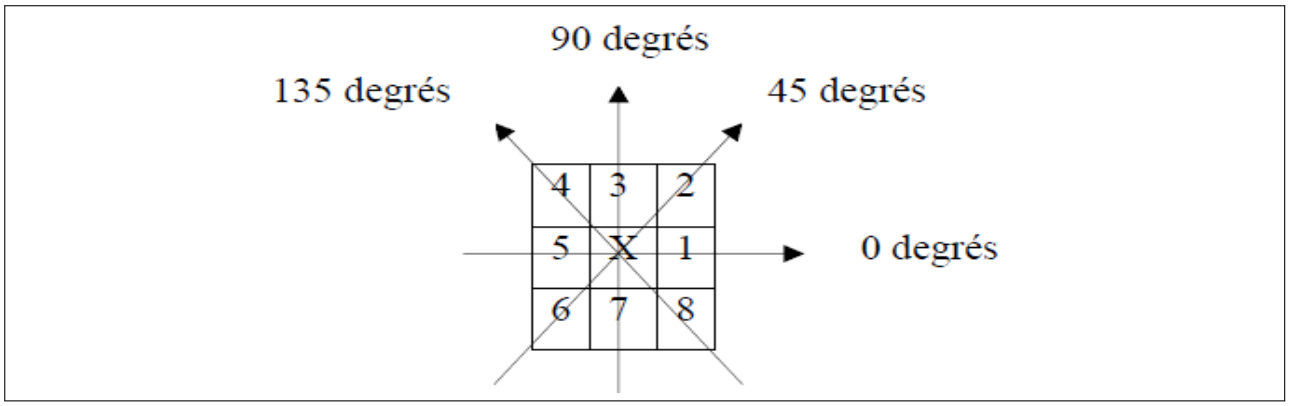

Figure 4.14 – Plus proches voisins du pixel 'x' selon 4 directions.

Les directions angulaires θ classiquement utilisées sont $0, 45, 90$ et 135 degrés. Les relations

de voisinage entre pixels, nécessaires au calcul des matrices, sont illustrées en figure 4.14 ; par exemple, les plus proches voisins de 'x' selon la direction $\theta = 135$ degrés sont les pixels 4 et 8. Les caractéristiques extraites à partir de ces matrices contiennent des informations notamment sur l'homogénéité, les dépendances linéaires entre les niveaux de gris, le contraste et la complexité de cette image.
Les matrices obtenues selon ces quatre directions sont alors calculées par les formules suivantes :

$$P(i,j,\delta,0) = Card\left\{\begin{array}{c}\Big((k,l),(m,n)\Big) \in (N\times M)^2 \ : \\ (k-m=0,\ |l-n|=\delta,\ I_{k,l}=i,\ I_{m,n}=j)\end{array}\right\}$$

$$P(i,j,\delta,45) = Card\left\{\begin{array}{c}\Big((k,l),(m,n)\Big) \in (N\times M)^2 \ : \\ (k-m=\delta,\ l-n=-\delta) \vee (k-m=\delta,\ l-n=-\delta),\ I_{k,l}=i,\ I_{m,n}=j\end{array}\right\}$$

$$P(i,j,\delta,90) = Card\left\{\begin{array}{c}\Big((k,l),(m,n)\Big) \in (N\times M)^2 \ : \\ (|k-m|=\delta,\ l-n=0,\ I_{k,l}=i,\ I_{m,n}=j)\end{array}\right\}$$

$$P(i,j,\delta,135) = Card\left\{\begin{array}{c}\Big((k,l),(m,n)\Big) \in (N\times M)^2 \ : \\ (k-m=\delta,\ l-n=-\delta) \vee (k-m=-\delta,\ l-n=-\delta),\ I_{k,l}=i,\ I_{m,n}=j\end{array}\right\}$$

où (k,l) sont les coordonnées d'un pixel de niveau de couleur $i \in [0, n^{max}-1]$ et (m,n) celles du pixel de niveau de couleur $j \in [0, n^{max}-1]$.

La plupart des images sont codées sur 256 niveaux de gris, par conséquent, la taille des matrices de co-occurrence est de 256×256. On s'aperçoit ainsi que ces matrices comptabilisent une très grosse quantité d'informations difficile à exploiter directement. C'est pourquoi, un certain nombre d'auteurs comme Zucker [39] ont essayé d'extraire de l'information de ces matrices afin de mettre en évidence la structure des textures. Mais c'est Haralick et al [35] qui ont proposé les premiers 14 paramètres, caractérisant les textures, issus de ces matrices. Voici 6 paramètres considérés comme étant les plus utilisés et les plus pertinents :

1) L'énergie :

$$ENE = \sum_i \sum_j \Big(P_{ij}(\delta,\theta)\Big)^2 \tag{4.7}$$

Ce paramètre mesure l'uniformité de la texture. Il atteint de fortes valeurs lorsque la distribution des niveaux de gris est constante ou de forme périodique. Dans ce dernier cas, les valeurs élevées d'énergie sont obtenues pour les matrices $P(\delta,\theta)$ lorsque (δ,θ) correspond à la période.

2) Le contraste :

$$CST = \sum_i \sum_j \Big((i-j)^2 P_{ij}(\delta,\theta) \Big) \tag{4.8}$$

La valeur en est d'autant plus élevée que la texture présente un fort contraste. Ce paramètre est fortement non corrélé à l'énergie.

3) L'entropie :

$$ENT = \sum_i \sum_j \Big(log(P_{ij}(\delta,\theta))\, P_{ij}(\delta,\theta) \Big) \tag{4.9}$$

Ce paramètre mesure le désordre dans l'image. Contrairement à l'énergie, l'entropie atteint de fortes valeurs lorsque la texture est complètement aléatoire (sans structure apparente). Elle est fortement corrélée (par l'inverse) à l'énergie.

4) La variance :

$$VAR = \sum_i \sum_j \Big((i-\mu)^2\, P_{ij}(\delta,\theta) \Big) \tag{4.10}$$

La variance mesure l'hétérogénéité de la texture. Elle augmente lorsque les niveaux de gris différent de leur moyenne. La variance est indépendante du contraste.

5) La corrélation :

$$COR = \sum_i \sum_j \Big(\frac{(i-\mu)(j-\mu)P_{ij}(\delta,\theta)}{\sigma^2} \Big) \tag{4.11}$$

COR mesure la dépendance linéaire (relativement à (δ,θ)) des niveaux de gris de l'image. La corrélation n'est corrélée ni à l'énergie, ni à l'entropie.

6) Le moment inverse :

$$IDM = \sum_i \sum_j \frac{P_{ij}(\delta,\theta)}{1+(i-j)^2} \tag{4.12}$$

IDM (Inverse Difference Moment) mesure l'homogénéité de l'image. Ce paramètre est corrélé à une combinaison linéaire des variables ENE et CST.

De nombreuses études ont été menées afin de caractériser, classifier, modéliser les textures à l'aide de ces paramètres.
La méthode d'extraction de ces paramètres basée sur le calcul des matrices de co-occurrence est une des méthodes les plus proches de la notion de texture. Elles mettent effectivement en avant les relations qui existent entre les pixels de l'image en faisant intervenir l'aspect local (les niveaux de gris) et l'aspect spatial ((δ,θ)).

4.6 Apprentissage par les réseaux de neurones bayésiens

4.6.1 Procédure avant apprentissage

Notons qu'on va faire la reconnaissance des visages des 40 personnes représentant la base des données ORL.
Après avoir lire l'image qui contient le visage, et la transformée en une matrice tel que chaque composante de la matrice représente le niveau de gris du pixel associé dans l'image, on calcule la matrice de co-ocurrence associée a cette matrice, et a partir de cette matrice de co-ocurrence on calcule les 6 caractéristiques pertinentes du visage,qui seront par la suite l'entrée du perceptron multicouche.
Les étapes avant apprentissage sont résumées dans la figure 4.15

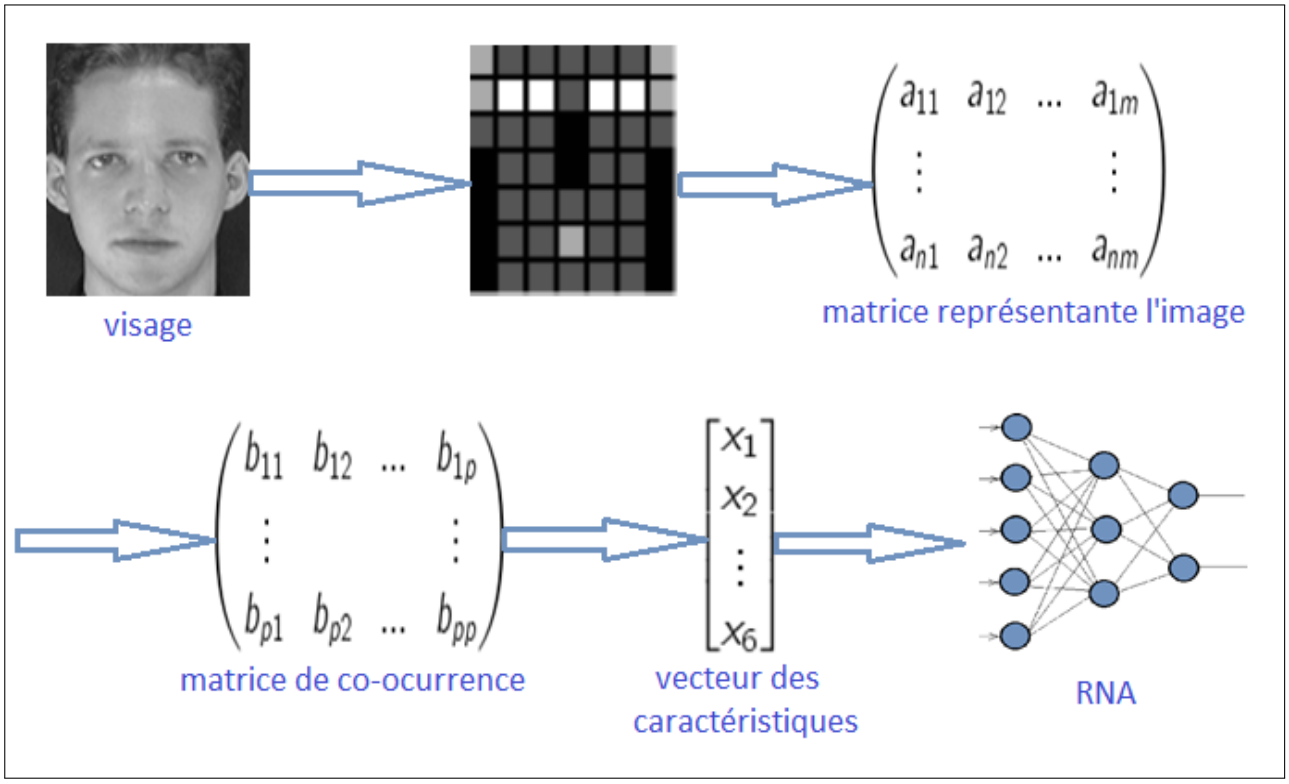

Figure 4.15 – Les étapes et la procédure avant apprentissage

4.6.2 Codage et architecture du réseau utilisé

- La base ORL contient 40 personnes différentes, donc on a besoin de 40 neurones dans la couches de sortie, de telle sorte que chaque neurone représente une personne.

- La sortie désirée associée a un vecteur des caractéristiques d'un visage du personne *i* est le vecteur nulle sauf dans la case *i* contient 1.

- Le vecteur des caractéristiques est de taille 6, alors on a besoin de 6 neurones dans la couche d'entrée, de telle sorte que chaque neurone représente une caractéristique :

. x_1 : représente le contraste.
. x_2 : représente l'entropie.
. x_3 : représente la variance.
. x_4 : représente la corrélation.
. x_5 : représente l'énergie.
. x_6 : représente le moment inverse.

- L'architecture du réseau utilisé est représentée dans la figure 4.16

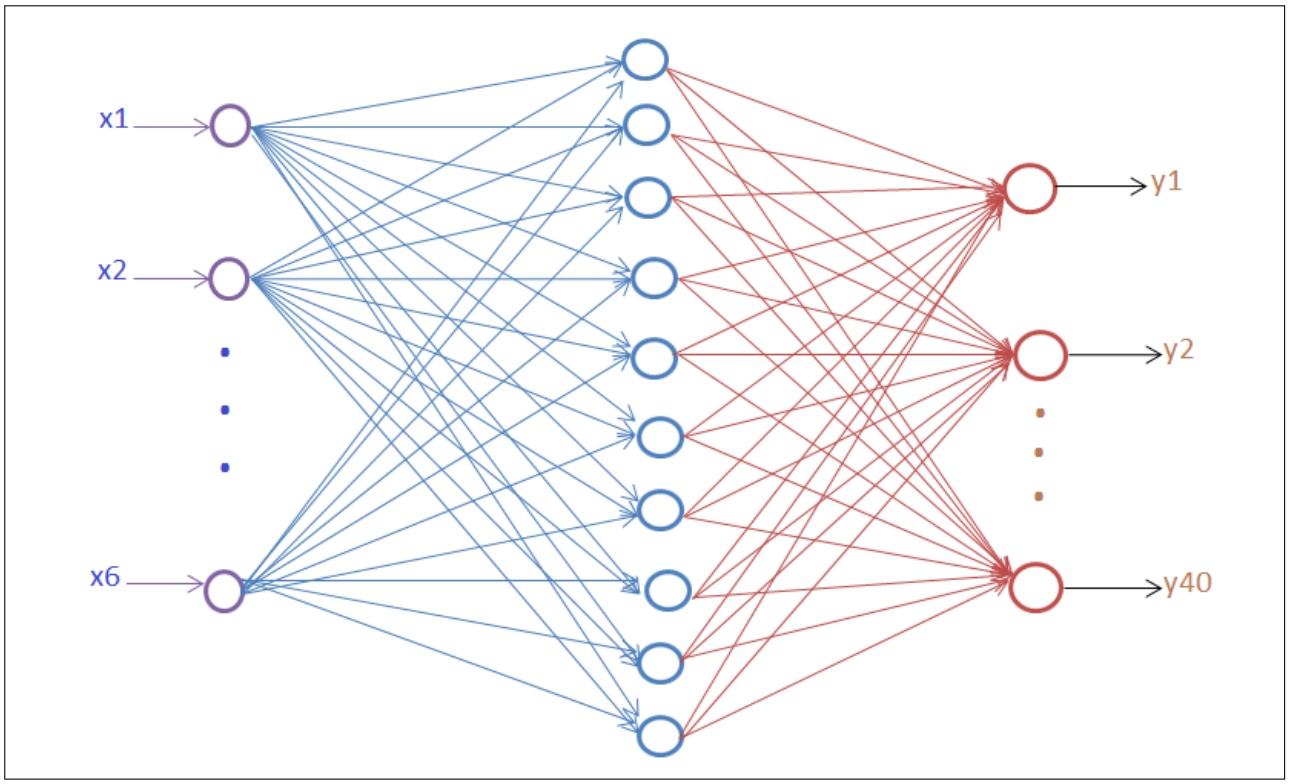

Figure 4.16 – Architecture du réseau utilisé dans l'apprentissage de la base ORL

4.6.3 Implémentation et résultats numériques

Connaître le taux de reconnaissance

Dans cette partie de notre programme on fait le test sur un ensemble des images qui n'appartiennent pas à l'ensemble des images d'apprentissage. Pour calculer le taux de reconnaissance, on divise le nombre des images du test reconnue sur le nombre total des images de test. On refait ces calcules pour des pourcentages différents des images d'apprentissage pour voir leurs influences sur le taux de reconnaissance.
Après avoir tourné l'algorithme de rétropropagation du gradient dans un premier temps et la procédure d'évidence après, pour le perceptron multicouche dont l'architecture est décrite avant. On a aboutit à les résultats représentés dans le tableau suivant :

Pourcentages des images d'apprentissage (%)	10	20	30	40	50	60	70	80	90	95
taux de reconnaissance (%) (Rétro-propagation de gradient)	20	32.6	40.83	50	52	53.5	60	67.74	71.98	80.13
taux de reconnaissance (%) (RNB)	19.35	35	40.11	53	55.13	58	63	69	75.42	85

commentaires

- Lors du phase d'apprentissage le pourcentage des images d'apprentissages a un effet sur le taux de reconnaissance , généralement lorsque on augmentent le pourcentage des images d'apprentissages le taux de reconnaissance augmente aussi, ce qui montre l'importance d'augmenter le nombre d'image d'apprentissage.

- D'après le tableau nous remarquons que dans la plupart des cas, le tau de reconnaissance pour les réseaux de neurones bayésiens est plus élevé que celle-ci pour l'apprentissage par rétro-propagation du gradient, ce qui montre l'importance de l'intégration des méthodes bayésiennes dans l'apprentissage des réseaux de neurones.

Conclusion

Dans ce chapitre nous avons présenté la reconnaissance faciale comme étant une technique biométrique ainsi que les principales difficultés de cette technique. Nous avons aussi détaillé deux méthodes de description des visages qui sont la description par les motifs binaires locaux et la description par la Matrice de co-occurrence. Et enfin nous avons appliqué les réseaux de neurones bayésiens dans la reconnaissance des visages.

CONCLUSION GÉNÉRALE ET PERSPECTIVES

Les réseaux de neurones artificiels est une technologie alternative pour simuler des complexes problèmes. Ils sont utilisés dans divers applications : robotique, reconnaissance des formes, traitement des images, reconnaissance des visages, reconnaissance de la parole, reconnaissance de locuteur, problème d'optimisation etc.

Les réseaux de neurones sont généralement optimisés par des méthodes d'apprentissage de type probabiliste, en particulier bayésien. Ils sont placés d'une part dans la famille des applications statistiques, qu'ils enrichissent avec un ensemble de paradigmes permettant de créer des classifications rapides, et d'autre part dans la famille des méthodes de l'intelligence artificielle auxquelles ils fournissent un mécanisme perceptif indépendant des idées propres de l'implémenteur, et fournissant des informations d'entrée au raisonnement logique formel.

L'utilisation du théorème de Bayes permet de proposer une formulation bayésienne de l'apprentissage, qui est très générale, et qui peut s'appliquer à la recherche de tous types de modèles pour peu que ceux-ci soient représentés par un ensemble de paramètres sur lesquels on peut faire une hypothèse de distribution. Nous avons décrit dans ce travail le principe de façon intuitive, et nous avons montré ensuite l'application dans le cadre des réseaux de neurones.

Les perspectives de ce travail sont nombreuses : dans un premier temps, nous souhaitons améliorer l'Igorithme d'évidence et intégrer d'autre méthodes bayésiennes dans l'apprentissage des réseaux de neurones et faire une comparaison entre ces méthodes. Dans un deuxième temps, nous souhaitons améliorer le module d'extraction automatique de visage et de ses caractéristiques. Enfin nous souhaitons appliquer les différentes méthodes bayésiennes développées, dans l'apprentissage et la reconnaissance des visages et comparer les résultats trouvés pour citer le modèle le plus adéquat a ce problème.

BIBLIOGRAPHIE

[1] A.A. Suratgar, M.B. Tavakoli, and A. Hoseinabadi, *"Modified Levenberg-Marquardt Method for Neural Networks Training"*, Water Resources Management, Vol.6, pp.1745–1747, 2007.

[2] A. Dupas, *"Opérations et Algorithmes pour la Segmentation Topologique d'Images 3D"*, Thèse de doctorat, Université de Poitiers, 2009.

[3] A.E. Shivdas, *"Face Recognition Using Artificial Neural Network"*, International Journal of Research in Management, Science and Technology, Vol.2, pp.2321-3264, 2014.

[4] A. Vehtari and J. Lampinen, *"Bayesian Neural Networks with Correlating Residuals"*, Neural networks, Vol.3, pp.1662-1665, 2009.

[5] A. Vehtari, S. Sarkka and J. Lampinen, *"On MCMC sampling in Bayesian MLP neural networks"*, Neural networks, Vol.1, pp.317-322, 2000.

[6] B.M. Wilamowski, S. Iplikci, O. Kaynak and M.Ö. Efe, *"An Algorithm for Fast Convergence in Training Neural Networks"*, Neural networks, Vol.3, pp.1778-1782 , 2001.

[7] C.M. Bishop, *"Exact Calculation of the Hessian Matrix for the Multi-layer Perceptron"*, Neural Computation, pp.494-501 , 1992.

[8] C.M. Bishop, *"Neural networks for pattern recognition"*, Oxford : Oxford University Press, 1995.

[9] C.P. Robert, *"Le choix bayésien : Principes et pratique"*, Springer-Verlag France, Paris, 2006.

[10] D. Arora and D. Kundu, *"Face Recognition Using Neural Network"*, International Journal of Computer Science and Telecommunications, pp. 18-20, 2015.

[11] D. Husmeier, W.D. Penny and S.J. Roberts, *"An empirical evaluation of Bayesian sampling with hybrid Monte Carlo for training neural network classifiers"*, Neural Networks, Vol.12, pp.677-705, 1999.

[12] D.J.C. MacKay, *"A practical Bayesian framework for back-propagation networks"*, Neural Computation, vol.4, pp.448-472, 1992.

[13] D.J.C. MacKay, *"Bayesian interpolation"*, Neural Computation, vol.4, pp.415-447, 1992.

[14] D.J.C. MacKay, *"Information theory, inference, and learning algorithms"*, Cambridge : Cambridge University Press, 2003.

[15] D.J.C. MacKay, *"Modelling phase transformation in steels - Bayesian non linear modelling with neural networks"*, Cambridge : Cambridge University Press, 1995.

[16] D.J.C. MacKay, *"The evidence framework applied to classification networks"*, Neural Computation, Vol.5, pp.720-736, 1992.

[17] F.D. Foresee and M.T. Hagan, *"Gauss-Newton Approximation to Bayesian Learning"*, Neural networks, Vol.3, pp.1930-1935 , 1997.

[18] F. Vivarellia and C.K.I. Williams, *"Comparing Bayesian neural network algorithms for classifying segmented outdoor images"*, Neural Networks, Vol.14, pp.427-437, 2001.

[19] H. Boughrara, M. Chtourou,and C.B. Amar, *"MLP neural network based face recognition system using constructive training algorithm"*, International conference on Multimedia computing and system, IEEE, pp. 233–238, 2012.

[20] H. CHOUAIB, *"Sélection de caractéristiques : méthodes et applications"*,Thèse de doctorat, Université Paris Descartes, 2011.

[21] H. Majdoulayne, *"Extraction de caractéristiques de texture pour la classification d'images satellites"*, Thèse de doctorat, Université de Toulouse, 2009.

[22] H. Ramchoun, M.A. Janati Idrissi, Y. Ghanou and M. Ettaouil, *"Multilayer Perceptron : Architecture Optimization and Training"*, International Journal of Interactive Multimedia and Artificial Intelligence, Vol.4, pp.26-30, 2016.

[23] I.T. Jolliffe, *"Principal Component Analysis"*,Springer-Verlag, 1986.

[24] J. Lampinen and A. Vehtari, *"Bayesian approach for neural networks-review and case studies"*, Neural Networks, Vol.14, pp.257-274, 2001.

[25] K. Delac, M. Grgic and S. Grgic, *"Independent Comparative Study of PCA, ICA, and LDA on the FERET Data Set"*, International Journal of Imaging Systems and Technology, Vol.15, pp.252-260, 2005.

[26] L. HOUAM, *"Contribution à l'analyse de textures de radiographies osseuses pour le diagnostic précoce de l'ostéoporose"*, Thèse de doctorat, Université de Guelma et Université

d'Orléans, 2013.

[27] Lijun Li and Shuling Dai, *"Bayesian Neural Network Approach to Hand Gesture Recognition System"*, Guidance, Navigation and Control Conference, IEEE, pp.2019-2023, 2014.

[28] M. Agarwal, N. Jain, M. Kumar and H. Agrawal, *"Face Recognition using Eigen Faces and Artificial Neural Network"*, International Journal of Computer Theory and Engineering, Vol.4, pp.1793-8201, 2010.

[29] M. ETTAOUIL and Y. GHANOU, *"Neural architectures optimization and Genetic algorithms"*, Wseas Transactions On Computer, Vol.8, pp.526-537, 2009.

[31] M. ETTAOUIL, M. LAZAAR and Y. GHANOU, *"Architecture optimization model for the multilayer perceptron and clustering"*,Journal of Theoretical and Applied Information Technology, Vol.47, pp.64-72, 2013.

[30] M. Feuilloy, *"Étude d'algorithmes d'apprentissage artificiel pour la prédiction de la syncope chez l'homme"*, Thèse de doctorat, Université d'Angers, 2009.

[32] M.M. Kasar, D.Bhattacharyya and T. Kim, *"Face Recognition Using Neural Network : A Review"*, International Journal of Security and Its Applications, Vol.10, pp.81-100, 2016.

[33] N. Jindal and V. Kumar, *"Enhanced Face Recognition Algorithm using PCA with Artificial Neural Networks"*, International Journal of Advanced Research in Computer Science and Software Engineering, Vol.3, pp. 864-872, 2013.

[34] P. Lauret, E. Fock, R.N. Randrianarivony and J.F. Manicom-Ramassamy, *"Bayesian neural network approach to short time load forecasting"*, Energy conversion and management, Vol.49, pp.1156-1166, 2008.

[35] R.M. Haralick, *"Statistical and structural approachs to texture"*, IEEE, Vol.67, pp.786–804, 1979.

[36] R.M. Neal, *"Bayesian learning for neural networks"*, Lecture Notes in Statistics. New York : Springer ; 1996.

[37] S.G. ABABSA, *"Authentification d'individus par reconnaissance de caractéristiques biométriques liées aux visages 2D/3D"*, Thèse de doctorat, Université Evry Val d'Essonne, 2008.

[38] S. Monjoly, *"Outils de prédiction pour la production d'électricité d'origine Eolienne"*, Thèse de doctorat, Université des Antilles et de la Guyane, 2013.

[39] S.W. Zucker and D. Terzopoulos, *"Finding structure cooccurrence matrices for texture analysis"*, Computer Vision Graphics and Image Processing, Vol.12, pp.286–308, 1980.

[40] T. Ahonen, A. Hadid, and M. Pietikainen, *"Face Description with Local Binary Patterns : Application to Face Recognition"*, IEEE transactions on pattern analysis and machine intelligence, Vol.28, pp.2037-2041, 2006.

[41] T. Ojala, M. Pietikainen, and T. Maenpaa, *"A Generalized Local Binary Pattern Operator for Multiresolution Gray Scale and Rotation Invariant Texture Classification"*, International Conference on Advances in Pattern Recognition, Springer Berlin Heidelberg, pp.399-408, 2001.

[42] T. Ojala, M. Pietikainen, and T. Maenpaa, *"Multiresolution gray-scale and rotation invariant texture classification with local binary patterns"*, IEEE Transactions on pattern analysis and machine intelligence, Vol.24, pp.971-987, 2002.

[43] T. Xiaoyang and B. Triggs, *"Enhanced Local Texture Feature Sets for Face Recognition Under Difficult Lighting Conditions"*, IEEE transactions on image processing, Vol.19, pp.1635-1650, 2010.

[44] V.P. Singh, D. Khalili, *"Daily Outflow Prediction by Multi Layer Perceptron with Logistic Sigmoid and Tangent Sigmoid Activation Functions"*, Water Resources Management, Vol.24, pp.2673–2688, 2010.

[45] V.P. Singh, D. Khalili, *"Modified Levenberg-Marquardt Method for Neural Networks Training"*, Water Resources Management, Vol.6, pp.2673–2688, 2010.

[46] W.D. Penny and S.J. Roberts, *"Bayesian neural networks for classification : how useful is the evidence framework ?"*, Neural Networks, Vol.12, pp.877-892, 1999.

[47] Y. Le Fablec, *"Prévision de trajectoires d'avions par réseaux de neurones"*, Thèse de doctorat, Université de Toulouse, 1999.

[48] Z. EN-NAIMANI and M. ETTAOUIL, *"Architecture Optimization Model of Probabilistic Neural Network"*, International Journal of Computer Science Issues, V.13, pp.1-9, 2016.

[49] Z. EN-NAIMANI M. LAZAAR and M. ETTAOUIL, *"Architecture Optimization Model for the Probabilistic Self-Organizing Maps and Speech Compression"*, International Journal of Computational Intelligence and Applications, V.15, p. 1650007, 2016.

[50] Z. Hammal, N. Eveno, A. Caplier and P.Y. Coulon, *"Extraction des traits caractéristiques du visage à l'aide de modèles paramétriques adaptés"*, traitement du signal, pp.59-72, 2005.

[51] www.face-rec.org