REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE
UNIVERSITE MOHAMED BOUDIAF - M'SILA

FACULTE DE TECHNOLOGIE
DEPARTEMENT D'ELECTRONIQUE
N° d'ordre:87/ESEM13/2018

DOMAINE : SCIENCES ET TECHNOLOGIE
FILIERE : ELECTRONIQUE
OPTION : ELECTRONIQUE DES SYSTEMES EMARQUEES

Mémoire présenté pour l'obtention Du diplôme de
Master Académique

Par

AICHE Ishaq & KETFI Meryem

Intitulé

MACHINES D'APPRENTISSAGE PUISSANTES POUR LA RECONNAISSANCE DES FORMES.

Soutenu le 17 juin 2018 devant le jury composé de:

Mr. KHENNOUF Salah	Université M'SILA	Président.
Dr. LADJAL Mohamed	Université M'SILA	Encadreur.
Mr. BRIK Youcef	Université M'SILA	Co-Encadreur.
Mr. DJERIOUI Mohamed	Université M'SILA	Examinateur.

Promotion : Juin 2018

Remerciements

Avant tout, nous remercions ALLAH le tout puissant qui nous a donnés la force et qui nous a permis d'arriver à ce stade-là. Nous remercions aussi nos parents pour leurs sacrifices en témoignage de tous nos affections au long de nos études.

Nous tenons à remercier en particulier nos encadreurs monsieur ***Dr. LADJAL Mohamed*** *et Monsieur* ***BRIK Youcef*** *de nous avoir aidé par leurs conseils, leurs remarques pertinentes et par leur collaboration effective pour la réalisation de ce mémoire.*

Nous tenons également à exprimer toute notre reconnaissance pour les membres du jury qui ont accepté d'honorer notre soutenance
Monsieur ***KHENNOUF Salah*** *en qualité de président du jury*
Monsieur ***DJERIOUI Mohamed*** *en tant qu'examinateur.*
Nous remercions également Monsieur ***ET-Tahir Zemmouri*** *qui nous a honoré par sa présence le jour de notre soutenance*

Nous remercions également ***Dr. SEGHIOUR Abdelatif*** *pour son aide, Ses conseils qui nous ont permis d'avancer dans ce travail*

Un grand merci aux membres actuels de la société de production d'électricité (SPE-M'sila), en particulier ***Mr. TELLI Abdelwahab*** *le chef de contrôle de l'économie pour son soutien, son humour, sa patience et sa disponibilité. Nos remerciements vont également à tous les enseignants du département d'électronique sans oublier le staff administratif.*

Dédicace

A mes chers parents.

A mes chers frères et soeurs.

Toute la famille aiche.

Tous mes amis.

Tous les collègues de notre promotion (2018).

Ishaq.

Dédicace

Je dédie ce modeste travail à mes très chères parents.

À ma petite famille, mon mari et mes enfants.

Toute la famille ketfi et Amroune.

Meryem.

Sommaire

Liste des Figures

Liste des Tableaux

Abréviations

- **ACP** : Analyse en Composante Principale.
- **AFD** : Analyse Factorielle Discriminante.
- **ConvNets** : Convolutional Neural Networks.
- **DBN** : Deep Belief Network.
- **FCM** : Fuzzy C-Means.
- **FPM** : Fuzzy Pattern Matching.
- **KKT** : Karush-Kuhn-Tucker.
- **KNN** : k-Nearest Neighbor.
- **k-PPV** : k-Plus Proches Voisins.
- **LOO** : Leave-One-Out.
- **MSE** : Mean Square Error.
- **MRS** : Minimisation du Risque Structurel.
- **MNIST** : Mixed National Institute of Standards and Technology.
- **MLP** : MultiLayer Perceptron.
- **OAA** : One Against All.
- **OAO** : One Against One.
- **OCR** : Optical Character Recognition.
- **OD** : Oxygène Dissous.

- **pH** : Potentiel Hydrogène.
- **PQ** : Programmation Quadratique.
- **RBF** : Radial Basis Function.
- **RdF** : Reconnaissance des Formes.
- **ReLU** : Rectified Linear Unit.
- **RGB** : Red, Green, Blue.
- **RNA** : Réseau de Neurones Artificiels.
- **SMO** : Sequential Minimal Optimisation.
- **SPE** : Société de la Production d'Electricité.
- **SVM** : Support Vector Machines.
- **TDS** : Totale Dissolved Solids.
- **VC** : Vapnik Chervonenkis.
- **VQP** : Vector Quantization and Projection.

Introduction générale

La simulation des fonctions humaines par la machine est un domaine de recherche très actif depuis l'avènement des ordinateurs. Une de ces fonctions est l'analyse de l'information visuelle, dite tout simplement l'analyse de la vision. La vision, plus que tout autre sens, nous fournit des informations sur l'environnement et guide la plupart de nos activités et nos interactions. En vision par ordinateur, l'un des principaux enjeux consiste à prendre une image (ou un signal) de la scène qui nous entoure (ou d'un capteur spécifique) pour la/le traiter et en dégager les informations pertinentes (en lien avec la tâche à accomplir). Parmi les différents aspects de l'information visuelle, la forme joue un rôle crucial dans différentes tâches de la vie quotidienne. La forme représente la caractéristique la plus pertinente de l'objet et elle peut être complétée par des informations de couleur ou de texture. En effet, la forme est l'un des plus importants attributs visuels d'un objet. La caractérisation de la forme fournit un indice puissant de l'identité et de la fonctionnalité de l'objet, et conduit à la reconnaissance de formes. La reconnaissance de formes reste à nos jours l'un des domaines ouverts de recherche et d'expérimentation. En vision par ordinateur, la reconnaissance de formes a été largement étudiée au cours des dernières années dans la littérature scientifiques. Elle est motivée par un large éventail d'applications importantes telles que la robotique, la biologie, l'internet, l'analyse de documents, les arts visuels, l'industrie et la sécurité, etc. Cependant, malgré toutes les recherches intensives, les techniques de reconnaissance disponibles restent au-dessous des capacités visuelles humaines. Nous pouvons dire que les techniques développées pour des applications particulières ont rencontré un succès limité sur des domaines restreints [Merhy et al., 2017].

Le travail présenté dans le cadre de ce mémoire a pour objectif de l'étude et la mise en œuvre des modèles d'apprentissage statistique appliqués dans le domaine de la reconnaissance

des formes en classification. Une étude en simulation est effectuée pour valider, évaluer, et comparer les performances de ces modèles dans un but de choix décisif adapté à deux problèmes de reconnaissance des formes dans deux domaines différents afin de pouvoir généraliser la puissances de ces machines sur des données de différente nature. La première application que nous voulons traiter est celle de la surveillance de la qualité des eaux usagées dans une station de production d'électricité [Mamar, 2008]. Par contre, la deuxième consiste à classifier automatiquement des chiffres pour objet de convertir des images qui sont compréhensibles par l'homme, en un code interprétable par un ordinateur .

Les énormes services offerts par les systèmes bancaires, les systèmes de tri postal, et les systèmes de gestion des formulaires ne sont pas assurés 100 % du fait de la très grande variabilité des styles d'écriture de différentes personnes [Duda et al., 2012]. La difficulté de la reconnaissance des chiffres est due principalement aux variations des styles d'écriture. Dans cette mémoire, nous allons traiter deux problématiques majeures rencontrées ; la surveillance et la classification des eaux industriels est un point essentiel dans l'industrie, à cause de la maintenance, le diagnostic préventif et le refroidissement des machines. Ces nécessite à de traiter dans ce contexte nous allons proposer un classifieur basé sur l'intelligence artificiel. par ailleurs, en reconnaissance des chiffres qui est la proposition d'un système de reconnaissance des chiffres permettant de s'adapter aux changements des styles d'écriture, aux différents angles d'orientation et différents fonts d'écriture pour en augmenter les performances.

Le manuscrit est séparé en trois chapitres organisés comme suit : Le premier chapitre est consacré au domaine de la reconnaissance des formes (RdF). Sa définition, sa formulation et ses principes sont décrits. Les différentes phases pour construire un système par RdF sont aussi illustrées. La dernière phase du processus de RdF consiste en une étape de classification.

Le deuxième chapitre présente les méthodes et les techniques d'apprentissage statistiques appliquées à la classification. Premièrement, nous allons rappeler les concepts fondamentaux qui sont à la base des machines à vecteurs de support, k plus proches voisins et réseaux de neurones artificiels. Un exemple de réseaux de neurones à convolution et Auto-encodeur pour l'apprentissage profond (deep learning) est présenté.

Enfin, Dans le troisième chapitre, nous allons présenter les deux bases des données, les

différents paramètres caractéristiques de l'eau, et la difficulté de la reconnaissance des chiffres synthétiques. Nous allons mettre en œuvre les techniques classiques (K-plus proches voisins(k-ppv), Machines à Vecteurs de Supports (SVM) et Réseaux de Neurones Artificiels (ARN)) et les machines d'apprentissage profondes (Réseaux de Neurones à convolutions (CNN) et Auto-encodeur) appliquées au contrôle de la déminéralisation de l'eau et de la reconnaissance des chiffres synthétiques. Une discussion des résultats de simulation obtenus pour le choix de la technique la plus adaptée à l'application conclue cette étude.

Finalement, une conclusion générale retrace les différentes étapes réalisées et souligne les perspectives envisagées.

Chapitre 1

La reconnaissance des formes

1.1 Introduction

Les fonctions de RdF s'appuient généralement sur un modèle de fonctionnement normal et/ou défaillant. Ce modèle est construit en utilisant une connaissance a priori sur le fonctionnement du système. Lorsque cette connaissance est suffisante pour construire un modèle mathématique ou analytique du système, les méthodes à base de modèle qualitatif et/ou quantitatif sont privilégiées. Lorsque cette connaissance est minimale et insuffisante pour construire un modèle analytique, les méthodes de Reconnaissance des Formes (RdF) constituent une solution efficace pour construire un modèle du système. Ce modèle est construit par apprentissage en utilisant un historique basé sur des observations du fonctionnement du système [Hartert, 2010].

La RdF regroupe l'ensemble des méthodes permettant la classification des formes (vecteurs de caractéristiques de l'état de fonctionnement du système) dans des classes.

Chaque classe est associée à un mode de fonctionnement du système et elle est représentée par un modèle. Les modèles de classes sont utilisés pour la classification d'une nouvelle forme à une des classes existantes [Hartert, 2010].

Nous allons présenter dans ce chapitre le système et le principe général des méthodes de reconnaissance des formes.Différentes phases pour construire un système par RdF sont aussi illustrés.

1.2 Principe général des méthodes de reconnaissance des formes

Le suivi de fonctionnement continu des systèmes industriels, tels que les robots, les systèmes hydrauliques, les machines de tri,... etc permet d'améliorer leur productivité et de faire décroître leur coût de production. Quand une faute est détectée, le système de suivi exécute une procédure de correction pour ramener le procédé dans des conditions normales d'opération. Plusieurs méthodes peuvent être utilisées pour réaliser le suivi de ces systèmes.

La sélection d'une de ces méthodes dépend de plusieurs facteurs comme :

- La dynamique du système (discrète, continue ou hybride),
- Les contraintes d'environnement (en ligne pour une réaction souhaitée en temps réel, ou hors ligne si la réaction peut être prise après un certain temps),
- La représentation des informations (quantitative et/ou qualitative),
- La complexité du système (grande ou simple),
- L'information disponible sur le système (structurelle, analytique, connaissances heuristiques, etc.),

Quand les connaissances a priori du comportement d'un procédé ne sont pas suffisantes pour construire un modèle analytique et quand seules des mesures sur son état de fonctionnement sont disponibles, les méthodes de Reconnaissance des Formes (RdF) sont particulièrement intéressantes. En effet, ces méthodes sont adaptées à la surveillance des modes de fonctionnement des systèmes, au diagnostic des fautes, ou à l'accomplissement du pronostic des défaillances, et la surveillance des eaux potables et industrielles. En RdF, le système est considéré comme une boite noire, c'est-à-dire qu'aucune équation mathématique n'est nécessaire pour modéliser son fonctionnement. Les méthodes de RdF utilisent exclusivement un ensemble de mesures et/ou de connaissances heuristiques du fonctionnement d'un procédé pour passer de l'espace d'observation à l'espace de décision, appelé espace de représentation. Les résultats de classification dépendent alors seulement, de la méthode elle-même,

des connaissances a priori, des paramètres caractérisant le système, et de la qualité des données [Hartert, 2010].

1.3 Système de reconnaissance des formes

Rappelons que sur des problèmes complexes, la modélisation du procédé est souvent difficile à mettre en œuvre. On privilégie alors l'approche par reconnaissance de formes. Celle-ci sera présentée comme une solution alternative à l'approche de surveillance par modèle puisque les modes de fonctionnement sont modélisés mais pas de manière analytique. On ne dispose pas de modèle issu de considérations sur les comportements du système ou de ses composantes, mais d'une base de données composée à partir d'un ensemble de mesures (individus) de ses modes de fonctionnement [Dubuisson, 2001, Bishop et al., 1995a, Boutleux, 1996, Boudaoud, 1997]. En fonction de l'application considérée, un individu est un ensemble de mesures réalisées sur un système physique ou un ensemble d'informations collectées lors de l'observation d'un phénomène. Dans la suite, un individu est représenté par le terme "forme" [Mamar, 2008].

1.3.1 Les approches de reconnaissance de formes

La reconnaissance de formes se base sur la définition d'algorithmes permettant de classer des formes en les comparants à des formes-types. Elle intervient dans de nombreux domaines tels que la reconnaissance vocale, la reconnaissance de caractères, l'automatisation industrielle, le diagnostic médical, la classification de documents, etc. De manière générale on distingue deux types de reconnaissance des formes :

- Les approches syntaxiques, structurelles et la mise en correspondance des formes (Template Matching) utilisent les hypothèses sur les distributions des données à l'intérieur des classes. Les procédures de classification, dans ce cas, sont construites à l'aide d'hypothèses probabilistes (Ex. classement Bayesien). Elles se basent sur une représentation des formes à l'aide de grammaires et nécessitent des moyens de calculs importants. De ce fait, ce type d'approche n'est pas approprié aux applications en temps réel [Jain et al., 2000] ;

- La reconnaissance de formes statistique, contrairement à la précédente, s'appuie sur une représentation numérique des formes en se basant sur des méthodes paramétriques ou non paramétriques. Les méthodes non paramétriques ne posent pratiquement aucune hypothèse sur la forme des distributions (Ex. K-plus proches voisins). Parmi les méthodes les plus utilisées, on cite les méthodes connexionnistes qui constituent un sous-ensemble des méthodes statistiques [Mamar, 2008].

1.3.2 Principe et formulation

On considère N formes, chacune définie par un ensemble de p paramètres regroupés dans un vecteur appelé vecteur forme x. L'espace $\Re^p$ défini par ces paramètres est appelé l'espace de représentation. Les formes-types ou prototypes constituent des points représentatifs de l'espace de représentation. Dans le cas réel, les observations sont bruitées et sont rarement confondues avec l'un des prototypes. Cela signifie que chaque forme-type est représentée par une zone géométrique désignée par le terme "classe" en reconnaissance de formes [Mamar, 2008].

Le principe est qu'on observe des formes de M classes différentes, $\Omega = \{\omega_i, i = 1, ..., M\}$. L'ensemble Ω définit "l'espace de décision". L'objectif est de construire des frontières réalisant une partition de façon à affecter un nouveau vecteur forme à l'une des classes $\omega_1, ..., \omega_M$. Cette association désigne l'opération de classement ou discrimination [Mamar, 2008].

L'ensemble des données issues du système de perception sert à construire le vecteur forme. La notion d'état de fonctionnement du procédé permet de construire l'ensemble d'apprentissage, celui-ci est composé de vecteurs formes associés à chaque mode de fonctionnement ou classe [Mamar, 2008].

De nombreux travaux, ont permis de montrer l'intérêt de ce type d'approche dans plusieurs domaines d'applications. Le domaine industriel est devenu l'un des premiers champs d'application pour le diagnostic par reconnaissance des formes [Mamar, 2008].

1.4 Construction d'un système de reconnaissance de formes

Un système de reconnaissance de formes se décompose généralement en cinq étapes séquentielles, (figure 1.1). La qualité de chaque étape du processus dépend de la qualité des étapes précédentes. Néanmoins, pour faciliter la compréhension, nous les regroupons en trois phases principales : phase d'analyse, phase de choix de la méthode de décision et enfin la phase d'exploitation [Mamar, 2008].

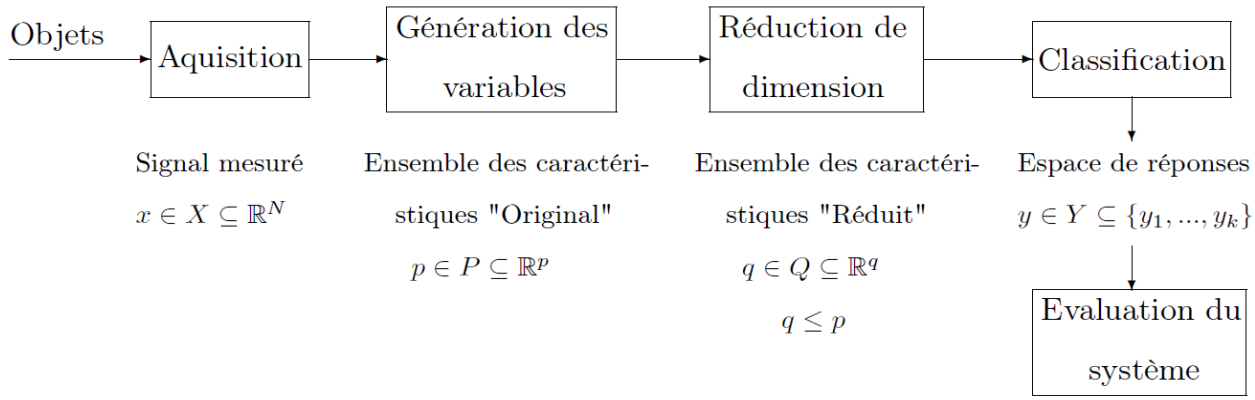


Figure 1.1 – Système de reconnaissance de formes.

1.4.1 Phase d'analyse

Cette première phase consiste à déterminer les paramètres et les techniques de prétraitement nécessaires pour permettre une bonne classification. La collecte d'informations sur le procédé est effectuée grâce au module d'acquisition, durant l'étape de perception. Suivant le domaine d'application, ces données peuvent être de nature qualitative, quantitative ou les deux à la fois.

1.4.1.1 Détermination de l'espace de représentation

Il s'agit de construire le vecteur forme x. Les données collectées par le module de perception sont des vecteurs réalisations de variables qui correspondent à des mesures réalisées sur un système physique ou à des informations collectées lors d'une observation d'un phénomène. Ces variables ne sont pas toutes aussi informatives. Elles peuvent correspondre à du bruit, peuvent

être peu significatives, corrélées ou redondantes, aberrantes ou simplement inexploitables. De ce fait, une étape de génération de variable s'impose.

L'étape de génération est définie comme la transformation initiale qui vient après la mesure du signal, produisant un ensemble original de paramètres dénoté $p \in P \subseteq \Re^p$. On utilise en général des méthodes d'analyse de données ou de traitement du signal pour effectuer cette transformation. Chaque observation effectuée à un instant donné sera caractérisée par l'ensemble de p variables $x = (x_1, x_2, ..., x_p)$.

L'objectif fondamental est d'accentuer les informations importantes du signal acquis. Cela implique une transformation du vecteur représentant le signal du domaine temporel vers un domaine où les informations contenues dans le signal seront mieux représentées.

L'étape de caractérisation des états de fonctionnement du système consiste à traiter les mesures recueillies lors de la phase d'acquisition pour extraire une représentation stable et concise, nécessaire à la mise en forme des caractéristiques du fonctionnement normal ou anormal du système. Elle représente le fruit de l'expérience acquise sur le processus et conditionne le succès du système de reconnaissance de formes.

Les données résultantes sont ensuite regroupées dans un tableau numérique de dimension $(N \times p)$ qui correspond à un ensemble de N formes pour lesquelles on connaît la valeur de p variables quantitatives. Les N formes $(x_1, x_2, ..., x_N)$ recueillies sur le système constituent l'ensemble d'apprentissage. La détermination de l'espace de représentation constitue l'étape clé dans la construction du module. La dimension p des variables disponibles à la fin de cette étape peut être très importante. Une forme peut se distinguer mieux des autres en éliminant des variables qui ne sont pas significatives ou en changeant de représentation. Ainsi, travailler dans un espace de représentation réduit, facilitera largement la tâche de la classification. En mode d'apprentissage, l'étape de réduction de variables permet de définir l'espace de représentation le plus approprié pour la classification. C'est dans cet espace de représentation que sont définies les formes à reconnaître.

1.4.1.2 Réduction de la dimension de l'espace de représentation

Il s'agit de réduire la dimension de l'espace de représentation afin, d'une part, de diminuer le temps de classification d'une nouvelle observation et, d'autre part, d'éviter la dégradation des performances à cause d'une dimension trop élevée de l'espace de représentation [Dash and Liu, 1997, Blum and Langley, 1997]. Cette opération de réduction de dimension peut être obtenue par une sélection ou une extraction de variables.

La sélection de variables consiste à ne conserver, parmi l'ensemble des p variables originales, que les plus pertinentes ou les plus informatives, au sens d'un critère de séparabilité entre les classes et d'un critère de condensation des points d'une même classe, comme par exemple le rapport de Fisher. Tandis que l'approche d'extraction, contrairement à la première, utilise toute l'information des variables initiales en les projetant le mieux possible dans un espace de dimension plus petite. En combinant entre elles les variables initiales, on définit de nouvelles variables par transformations linéaires ou non linéaires. Les méthodes d'extraction linéaires les plus utilisées sont l'Analyse en Composante Principale (ACP) et l'Analyse Factorielle Discriminante (AFD). En revanche si cette corrélation est non linéaire, des méthodes d'extraction non linéaire telle que le réseau Vector Quantization and Projection (VQP) sont alors utilisées. D'autres méthodes de projection non linéaires particulièrement adaptées à la visualisation des données, comme la projection de Sammon, sont également utilisées pour l'extraction des variables [Mamar, 2008].

1.4.1.3 Détermination de l'espace de décision

La reconnaissance de formes est réalisé en utilisant une méthode de classification qui associe une nouvelle observation à une classe correspondant à un mode de fonctionnement. Les méthodes de classification se divisent généralement en deux familles : les méthodes supervisées, et les méthodes non supervisées [Jain and Dubes, 1988, Duda et al., 2001]. L'espace de décision est défini par l'ensemble des classes possibles. Si l'on dispose d'un ensemble de formes étiquetées dont la classe d'appartenance est connue, la classification est dite supervisée. L'étiquetage permet de définir (ou d'apprendre) un ensemble fini de M classes d'objets en général mutuellement exclusives. L'objectif est alors de définir des règles

d'affectation pour une forme inconnue x à l'une des M classes apprises. Ainsi, on parle de prédiction en statistique et de classement en reconnaissance de formes. Nous utiliserons, par la suite, les termes discriminateur ou classifieur pour désigner une règle d'affectation [Mamar, 2008].

Dans le cas contraire, si la classe d'origine de chaque observation de l'ensemble d'apprentissage - l'espace de décision - n'est pas parfaitement connue, l'apprentissage des classes se fait en mode non supervisé. Il existe de nombreuses méthodes de classification non supervisée, nommées les méthodes de coalescence telles que les méthodes hiérarchiques et les méthodes des nuées dynamiques [Diday, 1971]. Le principe général de ces méthodes est de détecter dans l'ensemble d'apprentissage une structure de classes disjointes au sens d'un critère de similarité, de telle façon que les observations d'une même classe soient semblables et que celles de classes différentes soient dissemblables.

1.4.2 Phase de choix d'une méthode de décision

Une fois l'espace de décision défini, il s'agit de construire des frontières de décision entre les classes. Les méthodes de décision utilisées définissent une règle de décision pour la classification des nouvelles observations aux différentes classes de l'ensemble d'apprentissage. La performance de ces méthodes est estimée en utilisant un indice de performance. En général on choisit la probabilité de mauvaise classification. La règle de décision peut être construite en utilisant une approche statistique ou une approche analytique [Mamar, 2008].

Dans une approche statistique, on associe à chaque vecteur forme x une densité de probabilité conditionnelle $f(x/\omega_i)$ par rapport à chaque classe de w. Les concepts issus de la théorie statistique de la décision et de l'analyse discriminante peuvent être utilisés pour établir les frontières de décision entre les classes. En fonction de la connaissance, complète ou incomplète, dont on dispose sur la distribution de $f(x/\omega_i)$ on trouve différentes approches [Mamar, 2008] :

- lorsque la connaissance est complète, modèle et paramètres de étant supposés connus, la règle de décision optimale est basée sur la théorie Bayesienne de la décision [Dubuisson, 2001] [Fukunaga, 2013].

- lorsque la connaissance est incomplète c'est-à-dire dans la plupart des cas, le modèle de $f(x/\omega_i)$ inconnu, différentes stratégies peuvent être utilisées :
  - si la forme de $f(x/\omega_i)$ est supposée connue, on estime les paramètres de $f(x/\omega_i)$.
  - si la forme de $f(x/\omega_i)$ est inconnue, on utilise des méthodes non paramétriques pour l'estimation de $f(x/\omega_i)$ telles que les noyaux de Perzen [Dubuisson, 2001].
- ....

Sinon on estime les probabilités a posteriori en utilisant par exemple la méthode des k plus proches voisins $(k - PPV)$ et ses extensions, ou les réseaux neuromimetriques, etc. Que ce soit pour les méthodes paramétriques ou non paramétriques, l'estimation de la densité conditionnelle $f(x/\omega_i)$ se fait sur un ensemble d'apprentissage.

Il existe d'autres méthodes de classification qui divisent directement l'espace de représentation en des zones de décision en utilisant des indicateurs, des sorties binaires, telles que les réseaux de neurones basés sur la notion de distance ou les réseaux Learning Vector Quantization ou encore les classifieurs linéaires tels que les réseaux perceptron.

Dans l'approche analytique, la fonction discriminante est déterminée en estimant les paramètres d'une fonction mathématique de manière à séparer au mieux les classes c'est-à-dire à minimiser la probabilité de mauvaise classification. Le choix du modèle de la fonction mathématique dépend de la complexité de la frontière de décision. En pratique, on commence par des fonctions linéaires puis si les performances de classification ne sont pas satisfaisantes, on passe à des modèles plus complexes, quadratiques, réseaux neuromimetriques, etc [Mamar, 2008].

C'est à l'issue de ces deux premières phases qu'est élaboré le système qui décidera à quelle classe ω_i sera affectée une nouvelle observation x. Nous nous sommes intéressés aux méthodes de classification non paramétrique, plus adaptées aux cas industriels, pour lesquelles la densité de probabilité est complètement inconnue. Nous comptons environ 500 types de classifieur qui peuvent être utilisés lors de l'étape de la classification. Il est difficile de proposer un classement général des méthodes de classification supervisées et une même méthode peut être présentée sous divers points de vue [Mamar, 2008].

1.4.3 Phase d'exploitation

Évaluation du système de RdF

Le système de reconnaissance de formes permet de classer chaque nouvelle observation recueillie sur le système dans une des classes connues, en appliquant la règle de décision élaborée dans la phase précédente. La détermination de cette classe permet de connaître le mode de fonctionnement actuel du système. L'élaboration d'un système RdF suppose la connaissance exhaustive de tous les modes de fonctionnement du système, hypothèse rarement vérifiée, ce qui nécessite une adaptabilité du système de décision lors de l'apparition de nouveaux modes de fonctionnement.

Une fois l'apprentissage effectué, il est indispensable d'évaluer la qualité de la règle de classement induite. L'idée est d'appliquer une phase de test pour estimer l'erreur réelle de classement (taux de mauvais classement). Dans la pratique, l'échantillon S de taille N est divisé en deux ensembles : un ensemble d'apprentissage et un ensemble test. Selon la répartition entre les deux ensembles, il existe de nombreuses méthodes pour estimer l'erreur de classement :

- La ré-substitution : les deux ensembles d'apprentissage et de test sont identiques et correspondent à l'ensemble de l'échantillon. Si l'on note n_e le nombre d'erreurs commises, on aura $Erreur = \frac{n_e}{N}$;
- La méthode Holdout : on partitionne l'échantillon S en un ensemble d'apprentissage et un ensemble test. La répartition entre les deux ensembles doit être aléatoire ; en général dans des proportions 1/2, 1/2 pour chacun des deux ensembles ou 2/3 pour l'ensemble d'apprentissage et 1/3 pour l'ensemble test. Si m est la taille de l'ensemble test, on aura $Erreur = \frac{n_e}{m}$;
- La D-validation croisée [Burman, 1989] : l'échantillon est partitionné en D parties de tailles (approximativement) égales. L'apprentissage se fait alors sur les $(D-1)$ sous-ensembles et le test sur le D^{me} sous-ensemble restant. On réalise alors d'apprentissages en laissant à chaque fois une des parties de côté pour le test. Si l'on note $n_e(d)$ le

nombre d'erreurs de classement commises sur le d-ème sous-ensemble, l'estimation de l'erreur est la moyenne arithmétique des $n_e(d)$, $Erreur = \frac{1}{N} \sum_{d=1}^{D} n_e(d)$.

- La méthode du Leave-one-out : cette technique est un cas particulier de la validation croisée pour lequel $D = N$. La procédure de classement est répétée N fois sur $(N-1)$ instances ce qui la rend très coûteuse en temps de calcul, mais recommandée si l'ensemble E est de petite taille. En effet, elle conduit à un minimum de biais dans l'estimation de l'erreur ;

- La méthode de ré-échantillonnage (Bootstrap) : étant donné un échantillon E de taille N, on tire avec remise un ensemble d'apprentissage de même taille N (un élément de N peut ne pas appartenir à l'ensemble d'apprentissage, ou y figurer plusieurs fois), l'ensemble test est N. L'erreur est alors la moyenne des erreurs obtenues pour un certain nombre d'itérations de l'algorithme d'apprentissage.

Dans le cas supervisé, la méthode de ré-substitution est connue pour être facile et rapide à mettre en œuvre. Cependant, l'ensemble de test étant identique à l'ensemble d'apprentissage, l'estimation risque de souffrir d'un biais optimiste quasi systématique. On ne teste alors pas la capacité en généralisation mais plutôt la faculté d'apprendre éventuellement par cœur. La méthode **"Holdout"** est connue pour être pessimiste et biaisée surtout lorsque la taille des ensembles d'apprentissage et/ou de test est faible. Un partitionnement différent des ensembles d'apprentissage et test donne un résultat d'estimation différent, ce qui se traduit par une variance importante. A l'inverse, la technique du **"Leave-One-Out (LOO)"** est sans biais mais souffre de l'influence du choix de la base d'apprentissage. Les méthodes de **"Bootstrap"**, quant à elles, fournissent un bon estimateur de l'erreur de variance moins importante que la méthode du "Leave-one-out", mais sont encore plus coûteuses en temps de calcul. Elles sont très utiles pour les échantillons de petite taille. La technique de validation croisée réalise un bon compromis entre le biais et la variance [Burman, 1989].

1.5 Environnement statique et environnement dynamique

Selon le type de système étudié, les formes obtenues sur un système peuvent être statiques, ou dynamiques lorsque leurs caractéristiques évoluent avec le temps. Une forme statique est représentée par un point dans l'espace de représentation tandis qu'une forme dynamique est représentée par une trajectoire multidimensionnelle. Dans ce dernier cas, l'espace de représentation possède une dimension supplémentaire qui est le temps. De la même façon, les classes peuvent être statiques ou dynamiques. Les classes statiques sont représentées par des zones restreintes contenant des formes similaires dans l'espace de représentation. Pour ces classes, l'ordre d'arrivée des formes n'est pas lié à leur valeur d'appartenance et les paramètres du classifieur restent inchangés [Hartert, 2010].

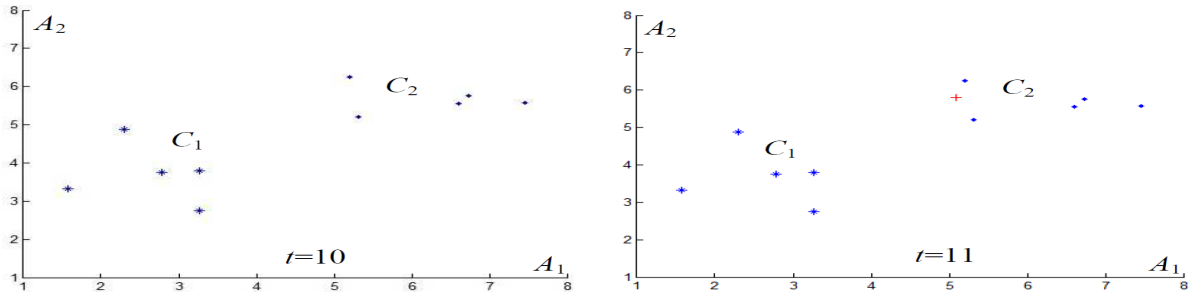
De nombreuses méthodes de RdF existent pour traiter le cas des données statiques. On peut citer la méthode des K-Plus Proches Voisins, les Séparateurs à Vaste Marge (SVM), les méthodes Bayésiennes, la méthode d'Analyse en Composantes Principales (ACP), la méthode Fuzzy Pattern Matching (FPM), la méthode Fuzzy C-Means, ainsi que les nombreuses déclinaisons de ces méthodes, etc.

1.6 Reconnaissance statique des formes statiques

Le RdF correspond au cas le plus classique de reconnaissance des formes. Les données statiques sont des données qui conservent les mêmes caractéristiques statistiques au cours du temps. Le problème correspond donc ici à la classification de nouvelles formes dont les caractéristiques ne changent pas par rapport à celles connues. Ces formes doivent être classées dans des classes statiques ne se déplaçant pas dans l'espace de représentation. Le classifieur utilise comme ensemble d'apprentissage ces formes et classes statiques, et il reste inchangé avec le temps [Hartert, 2010]. Toutefois, la base de données concernant un système est souvent incomplète, tous les modes de fonctionnement et notamment les modes défaillants ou dangereux ne peuvent pas être provoqués pour obtenir des données informatives. L'apparition des nouvelles données (bruitées ou non) va permettre d'affiner les fonctions d'appartenance

estimées par le classifieur. Dans ce cas, les petites adaptations de la classe sont réalisées, incrémentalement ou non, et elles permettent d'améliorer les résultats de classification. Dans certains cas, lorsqu'on peut réaliser un traitement hors ligne, il peut être plus intéressant de recommencer complètement la phase d'apprentissage en réapprenant un nouveau lot de données plus informatives.

Du point de vue statique, une forme représente l'observation du système en fonction de plusieurs paramètres à un instant donné. L'espace de représentation contient l'ensemble des formes connues observées à différents instants. Sur la figure 1.2, on peut voir une représentation de deux classes statiques à un instant $t = 10$ (avec 5 formes dans chaque classe) et à un autre instant $t = 11$ (où une nouvelle forme est apparue dans la classe $C2$). L'espace est composé de deux attributs $A1$ et $A2$, correspondant aux paramètres observés sur le système. On peut par exemple, utiliser un critère de similarité basé sur la distance entre les formes pour discriminer les classes. L'ordre et le temps d'arrivée des formes ne sont pas des informations utiles pour ce type de classification.



(a) forme à t=10 (b) forme à t=11

Figure 1.2 – Les deux classes possèdent chacune 5 formes à t=10 (a). A l'instant suivant t=11, une seule nouvelle forme '+' est apparue dans l'une des deux classes (b).

Sur la figure 1.2, les deux attributs sont notés $A1$ et $A2$. L'exemple d'espace de représentation montré sur cette figure peut être utilisé par n'importe qu'elle méthode statistique de reconnaissance des formes.

1.7 Notion de Classificateur

Soit la représentation d'un objet quelconque au moyen d'un vecteur de caractéristiques $X = [x_1 x_2 ... x_d]^T$. Tous les vecteurs qui représentent l'ensemble des objets peuvent être

positionnés dans l'espace Euclidien R^d, où ils correspondent chacun à un point. Ceux-ci peuvent alors être regroupés en amas, chacun de ces amas étant associé à une classe particulière. Un exemple pour un problème à deux classes est illustré à la figure 1.3. Le rôle d'un classificateur est de déterminer, parmi un ensemble fini de classes, à laquelle appartient un objet donné [Gosselin, 1996].

Un classificateur doit être capable de modéliser au mieux les frontières qui séparent les classes les unes des autres. Cette modélisation fait appel à la notion de *fonction discriminante*, qui permet d'exprimer le critère de classification de la manière suivante : " Assigner la classe ω_i à l'objet représenté par le vecteur X si, et seulement si, la valeur de la fonction discriminante de la classe ω_i est supérieure à celle de la fonction discriminante de n'importe quelle autre classe ω_i ".

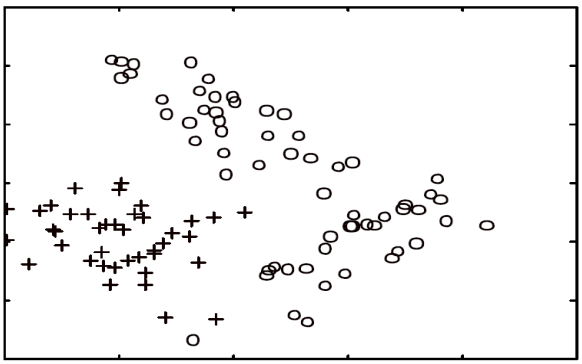


Figure 1.3 – Représentation d'objets appartenant à deux classes distinctes dans un espace à deux dimensions.

Les classificateurs peuvent être séparés en trois catégories distinctes :

- Les classificateurs paramétriques, qui sont entièrement définis par un ensemble fini de paramètres qu'il suffit de calculer,
- Les classificateurs non paramétriques, qui ne dépendent d'aucun paramètre en particulier,
- Les classificateurs dits « neuronaux », qui intègrent des fonctions discriminantes à la suite d'un apprentissage par l'exemple.

1.8 Les Classificateurs Paramétriques

1.8.1 Le Classificateur Euclidien

Il s'agit probablement de l'un des plus simples classificateurs qui puissent être conçus. La classe dont le vecteur de caractéristiques moyen est le plus proche, au sens de la distance Euclidienne, du vecteur de caractéristiques de l'objet à classifier est assignée à ce dernier.

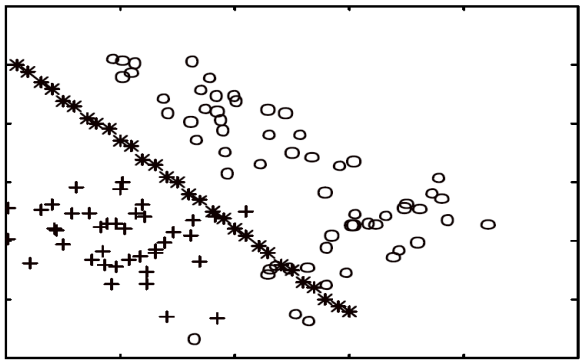


Figure 1.4 – Frontière fournie par le classificateur Euclidien dans le cas d'un problème à deux classes.

1.8.2 Le Classificateur Quadratique

Comme le nom l'indique, les frontières de décision fournies par ce modèle de classificateur sont quadratiques (figure 1.5).

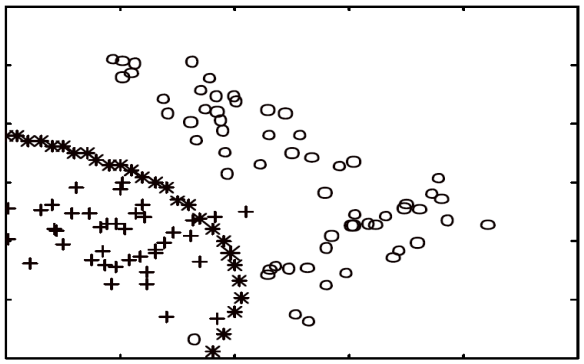


Figure 1.5 – Frontière obtenue à l'aide du classificateur quadratique.

1.8.3 Le Classificateur Gaussien

Les fonctions discriminantes utilisées ici sont basées sur une estimation paramétrique des fonctions de répartition des vecteurs de caractéristiques. Ce classificateur suppose que les

éléments de chaque classe possèdent une distribution Gaussienne multi variable.

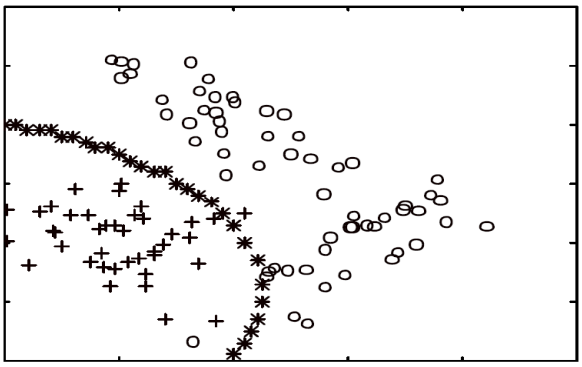


Figure 1.6 – Frontière générée par le classificateur Gaussien.

1.9 Les Classificateurs Non Paramétriques

1.9.1 La Méthode du Plus Proche Voisin

Ce classificateur est une extrapolation du classificateur Euclidien décrit précédemment. Au lieu d'utiliser le vecteur de caractéristiques moyen M_i comme unique prototype d'une classe, la méthode du plus proche voisin fait intervenir tous les exemplaires des vecteurs de caractéristiques disponibles. La distance Euclidienne entre chacun de ceux-ci et celui de l'objet à classifier est calculée, et la classe assignée à l'objet est celle du prototype le plus proche de celui-ci [Gosselin, 1996].

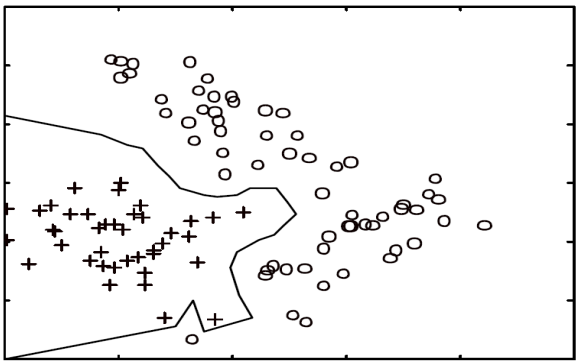


Figure 1.7 – Frontières fournies par le classificateur du Plus Proche Voisin.

1.9.2 La Méthode des k plus Proches Voisins

Un des inconvénients majeurs de la méthode du Plus Proche Voisin est que celle-ci présente une sensibilité élevée aux abords des frontières entre classes. Le plus proche voisin d'un objet

peut être d'une classe incorrecte, alors que la majorité de ses voisins ne le sont pas. Afin de contrer cet effet, la classe assignée à un objet peut être celle qui est la plus représentée parmi les k plus proches prototypes trouvés. La méthode porte dans ce cas le nom de « k Plus Proches Voisins ».

1.10 Les Classificateurs Neuronaux

La reconnaissance du fait que le cerveau fonctionne de manière entièrement différente de celle d'un ordinateur conventionnel a joué un rôle très important dans le développement des réseaux de neurones artificiels. Les travaux effectués pour essayer de comprendre le comportement du cerveau humain ont menés à représenter celui-ci par un ensemble de composants structurels appelés neurones, massivement interconnectés entre eux. Le cerveau humain en contiendrait plusieurs centaines de milliards, et chacun de ceux-ci serait, en moyenne, connecté à dix mille autres. Le cerveau est capable d'organiser ces neurones, selon un assemblage complexe, non-linéaire et extrêmement parallèle, de manière à pouvoir accomplir des tâches très élaborées. Par exemple, n'importe qui est capable de reconnaître des visages, alors que c'est là une tâche quasiment impossible pour un ordinateur classique [Gosselin, 1996].

C'est la tentative de donner à l'ordinateur les qualités de perception du cerveau humain qui a conduit à une modélisation électrique de celui-ci. C'est cette modélisation que tentent de réaliser les réseaux de neurones artificiels.

Haykin en propose la définition suivante [Haykin, 1994] : « Un réseau de neurones est un processus distribué de manière massivement parallèle, qui a une propension naturelle à mémoriser des connaissances de façon expérimentale et de les rendre disponibles pour utilisation. Il ressemble au cerveau en deux points :

1. La connaissance est acquise au travers d'un processus d'apprentissage ;

2. Les poids des connections entre les neurones sont utilisés pour mémoriser la connaissance.

1.11 Conclusion

Dans ce chapitre, le système et le Principe général des méthodes de reconnaissance des formes ont été présentés. Il s'agissait des méthodes de reconnaissance statique des formes statiques. Le choix d'une catégorie des méthodes de reconnaissance des formes se fait en fonction de différents éléments tels que les données (formes, classes, nombre des paramètres observés, etc.) et l'environnement du système (statique, dynamique).

Chapitre 2

Les Machines d'apprentissage

2.1 Introduction

La résolution des problèmes par la construction des machines capables d'apprendre à partir des entrées et des sorties, caractérise l'approche fondamentale de la théorie d'apprentissage (Machine Learning). Le problème typique de la théorie de l'apprentissage statistique se résume dans le contexte où des données engendrées par une distribution de probabilité (phénomène physique) [Ladjal, 2013].

Nous allons présenter dans ce chapitre des généralités sur les Machines à vecteur support(SVMs), la Méthode des k plus proches voisins (k-PPV) (k-nearest neighbor), leurs principes de fonctionnement, et aussi les fonctions noyaux de chaque méthode. Une étude générale sur les réseaux de neurones artificiels, le passage du modèle biologique au modèle artificiel, le protocole d'apprentissage seront présentés. Nous allons présenter un exemple usuel de RNA tel que perceptron multi couches (PMC). Nous parlerons également des Machines d'apprentissage puissantes tels que les autoencodeur et le réseau de neurone à convolution, qui se basent sur l'apprentissage profond.

2.2 Les k plus proches voisins (k-Nearest Neighbor)

2.2.1 Principe général

C'est une des méthodes de classification des plus simples qui donne généralement de bons résultats. Après plus de 35 ans, les classificateurs utilisant la règle des Plus Proches Voisins (PPV) sont encore largement étudiés et utilisés pour résoudre des problèmes de reconnaissance des formes. L'idée de base est d'inférer la classe d'un nouvel exemple en choisissant celle de l'exemple qui lui est le plus proche dans la base d'apprentissage. Ceci correspond à choisir pour un exemple x la classe $y = \omega_i$ de son plus proche voisin φ [Lebrun, 2006] :

$$D_{ppv}(x) = \omega_j | j = \arg \min_{1 \leq i \leq m} d(x, x_i) \tag{2.1}$$

Avec $d(x, x_i)$ une métrique permettant de mesurer la distance entre deux exemples. Pour éviter d'être trop sensible a des données bruitées, il est souvent réalisé la recherche des k plus proches voisins. Notons $N_{vision}(x, k, \omega_i, Z_m)$ la fonction retournant le nombre d'exemples de la classe parmi les k plus proches voisins de x. La classe sélectionnée est celle qui est majoritaire parmi ces k voisins, ce qui correspond à la fonction de décision suivante :

$$D_{k-ppv}(x) = \arg \max_{\omega_i \in y} d(x, x_i) \tag{2.2}$$

Si $k = 1$, alors l'individu est affecté à la classe du plus proche voisin de l'ensemble T. Une variante de la règle de la majorité consiste à prévoir un seuil s au-dessus du quel une décision de rejet est prise. Ainsi, on peut rencontrer des cas où l'individu n'est affecté à aucune classe. Soit l'exemple de la figure 2.1 avec deux dimensions correspondant aux attributs $e1$ et $e2$, et avec $k = 3$.

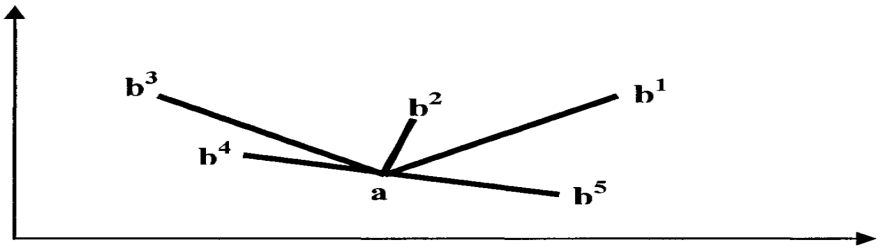


Figure 2.1 – Méthode des 3-pvv.

Dans cet exemple les trois plus proches voisins de a son $b4$, $b2$ et $b5$, donc a sera affecté à la classe majoritaire parmi ces trois points. Méthode des k-ppv a l'avantage d'être très simple à mettre en œuvre et d'utiliser directement l'ensemble d'apprentissage T. Elle ne fait aucune hypothèse a priori sur les données.

La qualité de la discrimination par cette méthode dépend du choix du nombre k de voisins considérés. Il est cependant souvent nécessaire de faire varier ce nombre k pour obtenir les meilleurs résultats possibles. Un autre problème important de la méthode des k-ppv est qu'elle nécessite un espace mémoire très important pour stocker les données et pour faire les différents calculs dans la phase de classification. De plus, elle a l'inconvénient d'utiliser les distances pour déterminer les voisins de l'individu à affecter, ce qui peut poser des problèmes si les dimensions à agréger ne sont pas homogènes. Afin de remédier à l'inconvénient de l'utilisation de distances, on a recours à l'utilisation des relations de ressemblances floues. [Perny, 1998]

2.2.2 L'algorithme KNN

L'algorithme KNN parmi les plus simples algorithmes d'apprentissage artificiel. Dans un contexte de classification d'une nouvelle observation x, l'idée fondatrice simple est de faire voter les plus proches voisins de cette observation. La classe de x est déterminée en fonction de la classe majoritaire parmi les k plus proches voisins de l'observation x. La méthode KNN est donc une méthode à base de voisinage, non-paramétrique ; Ceci signifiant que l'algorithme permet de faire une classification sans faire d'hypothèse sur la fonction qui relie la variable dépendante aux variables indépendantes [Hechenbichler and Schliep, 2004].

Algorithme 1-NN

La méthode du plus proche voisin est une méthode non paramétrique où une nouvelle observation est classée dans la classe d'appartenance de l'observation de l'échantillon d'apprentissage qui lui est la plus proche, au regard des covariables utilisées. La détermination de leur similarité est basée sur des mesures de distance.

Formellement, soit L l'ensemble de données à disposition ou échantillon d'apprentissage

$$L = \{(y_i, x_i)\}, i = 1, ..., n_L \tag{2.3}$$

Où $y_i \in \{1, ..., c\}$ dénote la classe de l'individu i et le vecteur $X_i = (x_{i1}, ..., x_{ip})$ représente les variables prédicatrices de l'individu i. La détermination du plus proche voisin est basée sur une fonction distance arbitraire $d(.,.)$.

La distance euclidienne ou dissimilarité entre deux individus caractérisés par p covariables est définie par :

$$d((x_1, x_2, ...x_p), (u_1, u_2, ..., u_p)) = \sqrt{(x_1 - u_1)^2 + (x_2 - u_2)^2 + ... + (x_p - u_p)^2} \tag{2.4}$$

Ainsi, pour une nouvelle observation (y, x) le plus proche voisin $(y_{(1)}, x_{(1)})$ dans l'échantillon d'apprentissage est déterminé par :

$$d(X, X_{(1)}) = min_i(d(X, X_1)) \tag{2.5}$$

et $\widehat{y} = y_{(1)}$, la classe du plus proche voisin, est sélectionnée pour la prédiction de y. Les notations $x_{(j)}$ et $y_{(j)}$ représentent respectivement le j^{me} plus proche voisin de x et sa classe d'appartenance. Parmi les fonctions distance types, la distance euclidienne est définie comme suit :

$$d(X_i, X_j) = (\sum_{s=1}^{p} (x_{is} - x_{js})^2)^{\frac{1}{2}} \tag{2.6}$$

et plus généralement la distance de Minkowski :

$$d(X_i, X_j) = (\sum_{s=1}^{p} (x_{is} - x_{js})^q)^{\frac{1}{q}} \tag{2.7}$$

Quelques règles sur le choix de k

Le paramètre k doit être déterminé par l'utilisateur : k et N. En classification binaire, il est utile de choisir k impair pour éviter les votes égalitaires. Le meilleur choix de k dépend du jeu de donnée. En général, les grandes valeurs de k réduisent l'effet du bruit sur la classification et

donc le risque de sur-apprentissage, mais rendent les frontières entre classes moins distinctes. Il convient donc de faire un choix de compromis entre la variabilité associée à une faible valeur de k contre un 'oversmoothing' ou 'surlissage' (i.e. gommage des détails) pour une forte valeur de k. Un bon k peut être sélectionné par diverses techniques heuristiques, par exemple, de validation-croisée. Nous choisirons la valeur de k qui minime l'erreur de classification.

2.2.3 Classification par la méthode le plus proche voisin

Le plus proche voisin (PPV) est une technique simple non paramétrique et très efficace, elle a utilisé dans plusieurs domaines, comme la reconnaissance des formes, classification des modèles, catégorisation de texte et classification pour les grandes données. Parmi, les algorithmes les plus utilisés dans les applications d'apprentissage automatique est la technique k-NN. Cette technique est connue par sa simplicité et des excellents résultats obtenus vis-à-vis différent problèmes rencontrer. Cet algorithme fonctionne en utilisant un vecteur d'entrée avec k échantillons d'apprentissage les plus proches dans l'espace des caractéristiques. Pour effectuer la classification, l'algorithme identifie la classe la plus commune parmi les k voisins les plus proches. L'algorithme est besoin à un apprentissage pour définir les voisins en fonction de la distance de l'échantillon de test, et une étape de test pour déterminer la classe à laquelle appartient cet échantillon de test [Vitola et al., 2017].

Le nombre de voisins peut être changé pour ajuster l'algorithme de K-NN. Dans ce contexte, beaucoup de voisins peuvent être trouvé pour d'ajustement. Il y a six classificateurs différents de k-NN disponibles dans MATLAB, peut être utilisé pour classifier des données. De plus, ces classificateurs sont basés sur des distances différentes comme le Fine, Medium et Coarse, les algorithmes de k-NN se servent de la distance euclidienne pour déterminer les voisins les plus proches. Selon MATLAB, chaque classificateur fonctionne comme suit :

- Fine k-NN : Un classificateur NN qui fait des distinctions finement détaillées entre les classes avec le nombre d'ensemble de voisins à un.
- Medium K-NN : Un classificateur NN avec moins de distinctions qu'un Fine k-NN avec le nombre d'ensemble de voisins à 10.

- Coarse K-NN : Un NN entre les classes, avec le nombre d'ensemble de voisins à 100.
- Cosine k-NN : Un classificateur NN qui utilise la distance de cosinus. La distance de Cosinus entre deux vecteurs u et v est définie comme :

$$1 - \frac{u.v}{|u|\,|v|'} \tag{2.8}$$

- C'est-à-dire un moins le ratio du produit intérieur de u et v sur le produit des normes de u et v.
- Cubic k-NN : Un classificateur NN qui utilise la distance cubique. La distance cubique entre deux vecteurs dimensionnels u et v de n est définie comme :

$$\sqrt[3]{\sum_{i=1}^{n} |u_i - v_i|^3} \tag{2.9}$$

- Weighted k-NN : Le classificateur NN qui utilise la pondération de distance. La distance Euclidienne pondérée entre deux vecteurs n-dimensional u et v est définie comme :

$$\sqrt{\sum_{i=1}^{n} w_i (x_i - y_i)^2} \tag{2.10}$$

Où

$$0 < w_i < 1 \quad et \quad \sum_{i=1}^{n} w_i = 1 \tag{2.11}$$

Le k-NN technique a été utilisé avec succès dans la détection de défaut dans les sondes de gaz, détection et classification de défaut pour les lignes de transmission à haute tension DC et prévision d'état du trafic.

2.3 Machines à Vecteurs de Supports

Les machines à vecteurs de support SVMs ou séparateurs à vaste marge constituent une technique d'apprentissage supervisé destinée à résoudre des problèmes de classification [Aboulaiche and Meskine, 2010]. La reconnaissance des mots manuscrits se fait dans un contexte supervisé où on cherche à estimer une fonction $f(x)$ à partir des exemples x, comme l'indique la figure 2.2.

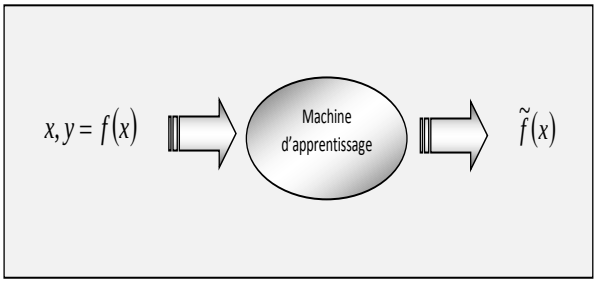


Figure 2.2 – Apprentissage supervisé d'une machine.

L'origine des SVMs remonte en 1975 lorsque Vapnik et Chervonenkis proposèrent le principe du risque structurel et la dimension de VC (VC : Vapnik Chervonenkis) pour caractériser la capacité d'une machine d'apprentissage. A cette époque ce principe n'a pas trouvé place car il n'existait pas encore un modèle de classification solidement appréhendé pour être utilisé avec cette théorie. Il a fait attendre jusqu'en 1982 pour que Vapnik propose un premier classifieur basé sur la Minimisation du Risque Structurel (MRS) baptisé SVM. Ce modèle était toutefois linéaire et l'on ne connaissait pas encore le moyen d'introduire des frontières de décision non-linéaires. En 1992 Boser et al [Boser et al., 1992] proposèrent d'introduire des noyaux non-linéaires pour étendre les SVMs aux cas non-linéaires. Par la suite, Cortes et Vapnik [Cortes and Vapnik, 1995] ont introduit des paramètres de relâchement qui tolèrent quelques erreurs d'apprentissage pour mettre fin à l'algorithme d'optimisation du SVM qui peut autrement, tourner indéfiniment. Dans ce qui suit nous présentons les classifieurs SVMs mais nous passons d'abord en revue quelques notions sur la théorie de l'apprentissage statistique.

2.3.1 Théorie de l'Apprentissage Statistique

La tâche d'un classifieur consiste à calculer une fonction de décision reliant les exemples à classifier x (appelés espace d'entrée), à leur classes y (appelées espace de sortie) où y est pris comme étant $\{+1, -1\}$ ce qui correspond au cas binaire.

Soit : $\{f_\alpha, \alpha \in \Re\}$, $f_\alpha : \Re^d \to \{\pm 1\}$ l'ensemble de fonctions, tel que $(x_1, y_1), ..., (x_i, y_i) \in \Re^d \times \{\pm 1\}$, sont des données d'apprentissage générées aléatoirement selon une distribution de probabilité $p(x, y)$ inconnue. La tâche d'apprentissage de f_α qui approxime le mieux cette distribution consiste à minimiser le risque réel donné par :

$$R(\alpha) = \int \frac{1}{2} |f_\alpha(x) - y| \, dP(x, y) \tag{2.12}$$

Ne connaissant pas $P(x, y)$ il est difficile d'estimer le risque $R(\alpha)$. il est possible toutefois de considérer une fonction du risque empirique de la forme :

$$R_{emp}(\alpha) = \frac{1}{2l} \sum_{i=1}^{l} |f_\alpha(x_i) - y_i| \tag{2.13}$$

A ce niveau, on peut légitimement se demander si la minimisation du risque empirique implique-t-elle la minimisation du risque réel ?

A partir de l'équation 2.13, nous constatons que pour une base de données $\{x_i, y_i\}$, le risque empirique est un nombre fixe pour un choix particulier de α. Ce dernier peut avoir une infinité de valeurs ce qui explique la complexité de l'apprentissage. Ainsi, la restriction de la recherche des f fonctions dans un intervalle limité de α, est très importante. En effet, si on lève cette restriction, il sera très facile de construire un classifieur donnant un risque empirique nul (n'importe quelle fonction f telle que $f(x_i) = y_i$, mais une telle fonction ne minimisera pas forcément le risque R (C'est l'apprentissage par cœur ou sur-apprentissage).

En 1979 Vapnik a mis au point le principe de la minimisation du risque structurel (MRS) qui évite le sur-apprentissage des données. La minimisation du risque structurel (MRS) procède par la construction d'une famille de classes telle que $f_1 \subset f_2 \subset ... \subset f_i$ avec une dimension de VC croissante. Les fonctions f_i sont choisies telles qu'une borne supérieure sur l'erreur de généralisation soit minimisée. Pour expliquer ce concept, soit un ensemble de l points ayant chacun une étiquette $+1$ ou -1 exprimant sa classe. La dimension de VC notée par h, constitue le nombre maximal de points pouvant être éclatés par une classe de fonctions et permet ainsi, de relier le risque réel au risque empirique. Avec une probabilité $1 - \mathrm{x}$, l'inégalité 2.14 est vraie .Le côté droit de cette inégalité est appelé borne du risque structurel alors que son second terme constitue la confiance de VC.

$$R(\alpha) \leq R_{emp}(\alpha) + \sqrt{\frac{h(\ln(2l/h) - \ln(\eta/4)}{l}} \tag{2.14}$$

L'erreur de généralisation est donc bornée par la somme du risque empirique et un terme

lié à la dimension de VC de la machine d'apprentissage. La minimisation de cette borne garantit la minimisation de l'erreur de généralisation.

Donc, étant données plusieurs fonctions de séparation, en choisissant un x suffisamment petit, et en prenant la machine qui minimise le côté droit de l'inégalité, on est en train de choisir la machine qui permet d'avoir la plus petite borne supérieure sur l'erreur de généralisation.

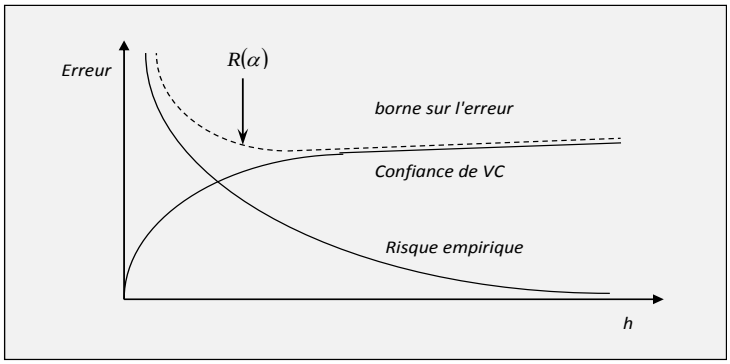


Figure 2.3 – Représentation graphique de l'équation 2.14.

2.3.2 Risque structurel

La MRS divise la classe de fonctions en plusieurs structures imbriquées de plus en plus complexes, d'où l'appellation du risque structurel (figure 2.4). L'apprentissage vise à trouver parmi ces structures celle qui réalise le meilleur compromis entre le risque empirique et la capacité de la machine. Ainsi, l'apprentissage démarre de la plus petite structure vers la plus grande mais il s'arrête lorsque la somme du risque empirique et le terme de confiance, est minimale (figure 2.3).

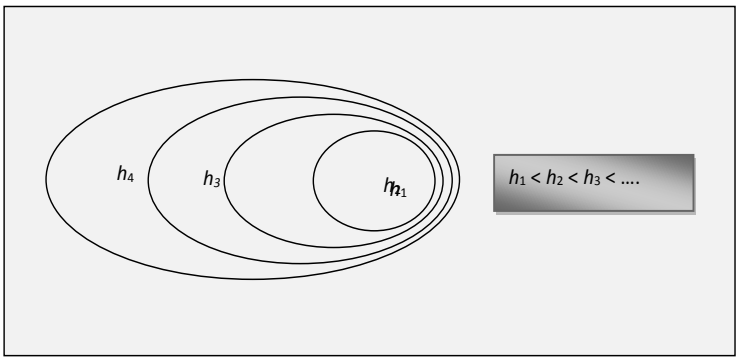


Figure 2.4 – Structures imbriquées d'une classe de fonctions.

2.3.3 Principe des SVMs

Les SVMs sont conçus pour mettre en œuvre le principe du risque structurel. Grâce à leur base théorique solide, ces classifieurs ont requis un très grand succès dans les différents domaines de traitement de données. En outre, ils répondent à deux problèmes majeurs de la théorie de l'apprentissage statistique qui sont :

- Le contrôle de la capacité du classifieur.
- Le sur-apprentissage de données.

Aussi, deux autres avantages sont particulièrement offerts par les SVMs. D'une part, contrairement aux machines d'apprentissage classiques dont le nombre de paramètres à définir est important, les SVMs possèdent une formulation élégante qui laisse très peu de place aux paramètres utilisateurs. D'autre part, puisque l'erreur ne dépend pas de la dimension de l'espace d'entrée, les SVMs sont bien adaptés pour le traitement des données de très haute dimension telles que les textes et les images. L'apprentissage d'un SVM consiste à chercher l'hyperplan permettant une séparation optimale entre deux classes de données. L'erreur est minimisée tout en maximisant la marge de séparation et en contrôlant la dimension VC du classifieur.

SVMs linéaires

Données linéairement séparables

Soit un ensemble de données d'apprentissage : $\{x_i, y_i\}, i = 1, ..., l$ avec : $x_i \in R^d$ et $y_i \in \{-1, +1\}$. Supposons qu'on dispose d'un hyperplan séparant les données positives des données négatives. Les x_i qui appartiennent à l'hyperplan vérifient la relation : $x_i.w + b = 0$. Où w est la normale de l'hyperplan tandis que $|b| / \|w\|$ est la distance perpendiculaire entre l'hyperplan et l'origine (figure 2.5).

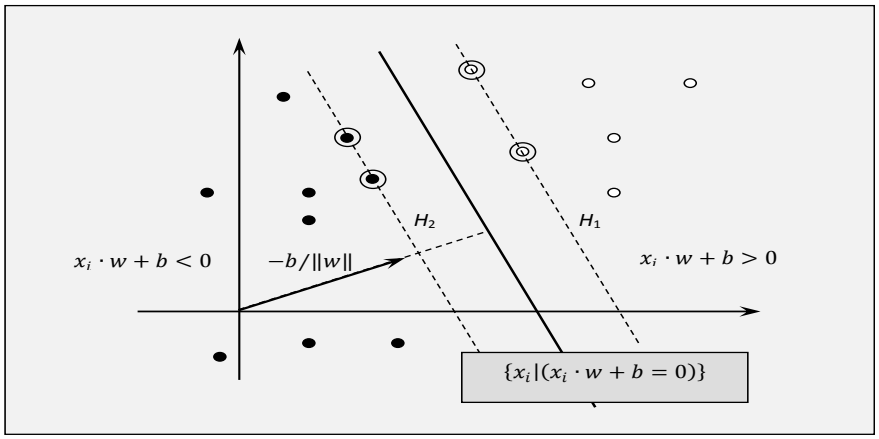


Figure 2.5 – Hyperplan de séparation linéaire pour des données linéairement séparables (Les vecteurs de support sont encerclés).

L'algorithme des vecteurs de support cherche simplement l'hyperplan séparateur permettant la plus large marge entre les exemples des deux classes. Ceci est formulé par les inégalités suivantes :

$$x_i.w + b \geq +1, y_i = +1 \tag{2.15}$$

$$x_i.w + b \leq -1, y_i = -1 \tag{2.16}$$

Et l'on écrira aussi :

$$y_i(x_i \cdot w + b) - 1 \geq 0 \tag{2.17}$$

Les points vérifiant l'égalité de 2.15, appartiennent à l'hyperplan $H_1 : x_i \cdot w = +1$ dont la normale w est de distance $|1 - b| / \|w\|$ par rapport à l'origine. Similairement, les points vérifiant l'égalité de 2.16, appartiennent à l'hyperplan $H_2 : x_i \cdot w + b = -1$, dont la normale est w, tandis que sa distance par rapport à l'origine est $|-1 - b| / \|w\|$. Notons que H_1 et H_2 sont parallèles et qu'aucune donnée d'apprentissage ne se trouve entre eux. Les distances entre ces hyperplans et l'hyperplan séparateur sont donc : $d^+ = d^- = 1/\|w\|$ ce qui donne une marge égale à $2/\|w\|$.

Ainsi, la maximisation de la marge se fait en minimisant $\|w\|^2$ qui est un problème d'optimisation souvent reformulé en Lagrangien pour deux raisons de simplification. La

première est que la contrainte dans 2.17 sera remplacée par une contrainte sur les multiplieurs du Lagrangien plus simple à traiter. En plus, dans cette reformulation, seules les données d'apprentissage apparaissent sous forme d'un produit scalaire. Cette propriété cruciale permet de généraliser ce problème au cas non linéaire. Le Lagrangien primal , correspond à la différence entre la fonction objective ($\frac{1}{2}\|w\|^2$) et la multiplication de la contrainte dans 2.17 par des multiplieurs α_i.

$$L_p = \frac{1}{2}\|w\|^2 - \sum_{i=1}^{l} \alpha_i y_i (x_i \cdot w + b) + \sum_{i=1}^{l} \alpha_i \tag{2.18}$$

L'objectif est la minimisation du Lagrangien par rapport à w et b sous la contrainte que les dérivées premières par rapport à tous les multiplieurs α_i, disparaissent et que $\forall i, \alpha_i \geq 0$. Ce problème étant convexe et puisque la fonction objective est elle-même convexe, on travaille plutôt sur sa forme duale. Alors, l'optimisation du SVM revient donc à maximiser le Lagrangien dual (L_D) par rapport aux α_i, sous les contraintes que ses dérivées premières par rapport à w et b disparaissent.

En imposant que les gradients de L_p par rapport à w et b disparaissent, nous obtenons :

$$w = \sum_i \alpha_i y_i x_i \tag{2.19}$$

$$\sum_i \alpha_i y_i = 0 \tag{2.20}$$

En substituant les équations 2.19 et 2.20 dans 2.21, nous obtenons le Lagrangien Dual L_D :

$$L_D = \sum_i \alpha_i \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j x_i x_j \tag{2.21}$$

Après l'optimisation, les points dont les α_i sont strictement supérieurs à 0, appelés vecteurs de support, appartiennent à l'un des hyperplans H_1 ou H_2 . Ces points sont les plus proches de la frontière de décision et forment le plan séparateur. Les autres points n'interviennent pas dans le calcul de la fonction de décision. Par ailleurs, les conditions de Karush-Kuhn-Tucker (KKT) résument les contraintes que le processus d'optimisation doit vérifier.

- Conditions de Karush-Kuhn-Tucker

Les conditions de KKT pour le problème d'optimisation du SVM sont les suivantes :

$$\frac{\partial}{\partial b}L_p = w_v - \sum\nolimits_i \alpha_i y_i x_{iv} = 0, v = 1, ..., d \tag{2.22}$$

$$\frac{\partial}{\partial b}L_p = -\sum\nolimits_i \alpha_i y_i = 0 \tag{2.23}$$

$$y_i(x_i \cdot w + b) - 1 \geq 0, i = 1, ..., l \tag{2.24}$$

$$\alpha_i \geq 0, \forall i \tag{2.25}$$

$$\alpha_i(y_i(x_i \cdot w + b) - 1) = 0, \forall i \tag{2.26}$$

d : Dimension des données.

Ces conditions sont nécessaires et suffisantes pour que les w et b et α_i ainsi obtenus soient une solution du problème. En résumé, l'optimisation d'un SVM revient à trouver une solution aux conditions KKT. Enfin, la fonction de décision est donnée par :

$$g(x) = sign(\sum\nolimits_{i=1}^{Vs} \alpha_i y_i x_i^T + b) \tag{2.27}$$

Vs : Nombre de vecteurs de support.

Avec : $(\sum_{i=1}^{Vs} \alpha_i y_i x_i^T = w)$ alors que le seuil b peut être calculé en utilisant l'équation 2.26 appliquée pour un exemple dont le α n'est pas nul.

Données non linéairement séparables

Lorsque les données ne sont pas linéairement séparables (figure 2.6), [Cortes and Vapnik, 1995] ont proposé de relâcher les contraintes 2.15 et 2.16 en introduisant des variables $\xi_i \geq 0$, telles que :

$$x_i \cdot w + b_i \geq +1 - \xi_i, y_i = +1 \tag{2.28}$$

$$x_i \cdot w + b \geq -1 + \xi_i, y_i = -1 \tag{2.29}$$

$$\xi_i \geq 0, \forall i \tag{2.30}$$

Pour qu'une erreur se produise, le ξ_i doit être supérieur à 1. La somme $\sum_i \xi_i$ constitue une borne supérieure sur le nombre des fausses classifications de la base d'apprentissage et contrôle ainsi le risque empirique. Cependant, elle peut être considérable si plusieurs ξ_i atteignent des valeurs très grandes lorsque les classes à séparer chevauchent énormément, par exemple à cause du bruit. Dans ce cas, il n'y a aucun garant que l'hyperplan calculé constitue la solution optimale. Pour contrôler le coût sur les erreurs, le terme $(\frac{1}{2}\|w\|^2)$ de la fonction objective est remplacé par $\frac{1}{2}\|w\|^2 + c(\sum_i \xi_i)$. La seule différence par rapport à l'hyperplan optimal est que dans ce cas, les $\rightarrow i$ ont une borne supérieure fixée par un paramètre C appelé paramètre de régularisation. Ce dernier contrôle le compromis entre les erreurs sur les données d'apprentissage et la largeur de la marge. Plus C est grand, plus la pénalité sur l'erreur est grande.

Notons encore que les ξ_i sont nuls pour les exemples dont les multiplieurs du Lagrangien se trouvent dans l'intervalle $0 < \alpha_i < C$.

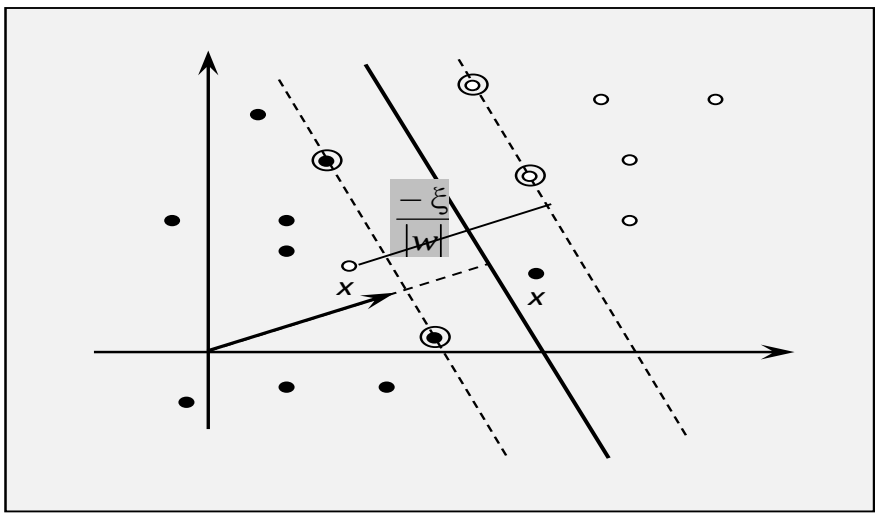


Figure 2.6 – Hyperplan de séparation linéaire pour des données non linéairement séparables.

SVMs non linéaires

Naturellement, en reconnaissance de formes les données nécessitent une fonction de décision non linéaire pour être séparées. Pour généraliser les SVMs aux cas non linéaires, ont introduit des noyaux de Merce permettant de trouver une hyper-courbe qui marquerait la frontière de décision entre les exemples positifs et négatifs. La figure 2.7 illustre le passage de l'espace d'entrée vers l'espace augmenté.

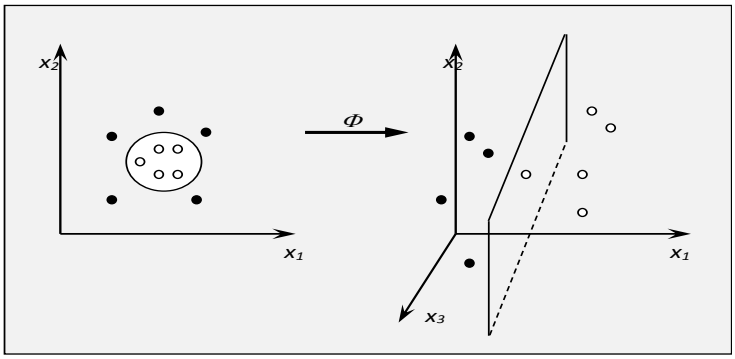


Figure 2.7 – Exemple d'une application Φ rendant les données linéairement séparables.

Les auteurs ont montré que le problème d'optimisation reste typiquement le même puisque seul le produit scalaire $x_i \cdot x_j$ est substitué par le produit des projections dans l'espace H : $\Phi(x_i) \cdot \Phi(x_j)$. La dimension de cet espace peut être infinie ce qui fait qu'une connaissance explicite de Φ n'est pas triviale. Pour contourner ce problème, les noyaux (ou Kernel) de Mercer telles que $\mathrm{K}(x_i, x_j) = \Phi(x_i) \cdot \Phi(x_j)$ sont employés pour calculer le produit entre les projections à partir de l'espace de données. Le Lagrangien dual de la fonction objective à maximiser devient alors :

$$Maximiser \quad \sum_{i=1}^{l} \alpha_i - \frac{1}{2} \sum_{i,j}^{l} \alpha_i \alpha_j y_i y_j \mathrm{K}(x_i, x_j) \tag{2.31}$$

Sous les contraintes 2.15 et 2.16. Nous obtenons ainsi un SVM fonctionnant dans un espace de dimension infinie où toutes les considérations des sections précédentes sont vérifiées puisque nous maintenons toujours la séparation linéaire, mais dans un espace différent. La fonction de décision devient :

$$g(x) = \sum_{i=1}^{Vs} \alpha_i y_i \mathrm{K}(\mathrm{x}_i, \mathrm{x}) + b \tag{2.32}$$

2.3.4 Algorithmes d'optimisation du SVM

A cause de sa taille considérable, le problème de Programmation Quadratique d'un SVM ne peut pas être facilement résolu avec les techniques standards. La matrice de gram contient un nombre d'éléments égal au carré du nombre total de données. Depuis plusieurs années, deux algorithmes ont été proposés pour ce fait. Il s'agit de l'algorithme procédant par tronçon ou Chunking et l'algorithme de décomposition. Plus récemment, Platt a proposé un algorithme beaucoup plus rapide baptisé optimisation minimale et séquentielle (ou *Sequential Minimal Optimisation SMO*).

- Optimisation minimale et séquentielle : SMO

Le SMO est un algorithme simple et rapide proposé pour résoudre le problème de PQ du SVM sans la nécessité de stocker une grande matrice en mémoire. A l'instar des autres algorithmes, il décompose le problème d'optimisation en plusieurs sous-problèmes mais à chaque étape il optimise le plus petit problème possible. En effet, à chaque étape les valeurs de deux multiplieurs du Lagrangien sont optimisées. Ceci dit, l'avantage principal du SMO réside dans le fait que l'optimisation de deux multiplieurs peut être effectuée analytiquement. En plus, le SMO ne nécessite pas un espace mémoire important. Cet algorithme se compose principalement de trois éléments :

1. Une méthode analytique pour résoudre le problème de PQ.
2. Des heuristiques pour choisir les multiplieurs à optimiser.
3. Une méthode pour calculer le seuil b.

2.3.5 Implémentations multi-classes des SVMs

A l'origine, les SVMs ont été conçus pour la séparation de deux classes. Cependant, plusieurs approches permettent d'étendre cet algorithme aux problèmes multi-classes. En effet, il existe deux approches classiques permettant l'implémentation multi-classes des SVMs.

2.3.5.1 Un contre un (OAO : one against one)

Dans cette approche chaque SVM sépare deux classes de données. Pour un problème de classes, on doit construire M(M − 1)/2 SVMs. On définit f_{ij} la fonction de décision séparant la classes i de la classes j par : $f_{ij} = -f_{ij}(p)$ durant la phase de classification, on calcule le score de chaque classe tel que :

$$S_i\,(p) = \sum\nolimits_{j\neq i, j=1}^{M} sign\ \left(f_{ij}\,(p)\right) \tag{2.33}$$

Enfin, p est affecté à la classe ayant le plus grand score :

$$\mathrm{argmax_{i=1,....M} S_i(p)} \tag{2.34}$$

2.3.5.2 Un contre tous (OAA : one against all)

C'est la première approche proposée pour l'implémentation multi-classes des SVMs. Le principe consiste à déterminer pour chaque classe un hyperplan séparant celle-ci de toutes les autres classes. Ainsi, pour M classes on doit déterminer M fonction de décision. Tous les exemples appartenant à la classe considérée sont étiquetés positivement $(+1)$ et tous les exemples n'appartenant pas à la classe sont étiquetés négativement (-1). La figure 2.8 décrit le schéma synoptique de cette séparation multi-classes. Par ailleurs, pour chaque exemple de test, on calcule M fonctions de décision. Ensuite, l'exemple est affecté à la classe qui correspond à la classe ayant la plus grande valeur de sortie.

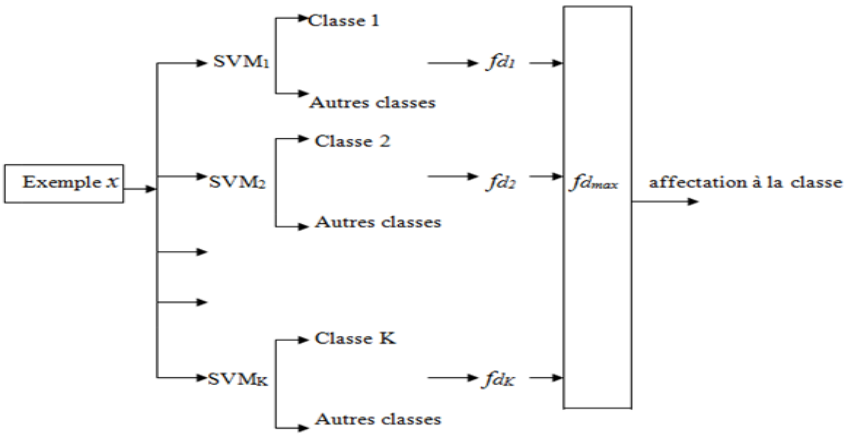


Figure 2.8 – Schéma synoptique de l'implémentation un contre tous.

2.4 Réseaux de neurones

2.4.1 Définition

Qu'est-ce qu'un réseau de neurones ? [Sarzeaud, 1995]

Tout d'abord, ce que l'on désigne habituellement par réseau de neurones est en fait un réseau de neurones artificiels basé sur un modèle simplifié de neurone. Ce modèle permet certaines fonctions du cerveau, comme la mémorisation associative, l'apprentissage par exemple, le travail en parallèle. Mais le neurone artificiel est loin de posséder toutes les capacités du neurone biologique. Les réseaux de neurones biologiques sont ainsi beaucoup plus compliqués que les modèles mathématiques et informatiques.

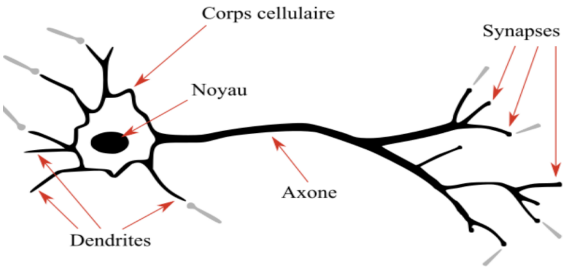


Figure 2.9 – Schéma d'un neurone biologique.

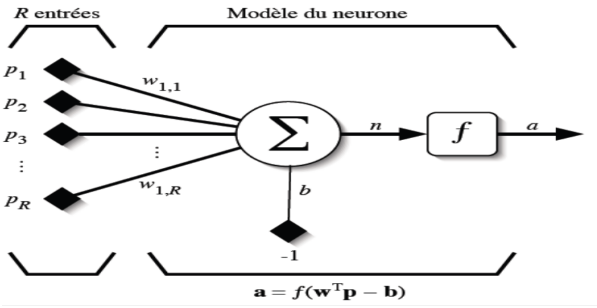


Figure 2.10 – Perceptron.

Il n'y a pas de définition universellement acceptée de « réseau de neurones ». On considère généralement qu'un réseau de neurones est constitué d'un grand ensemble d'unités. (ou neurones), ayant chacune une petite mémoire locale. Ces unités sont reliées par des canaux de communication (les connexions, aussi appelées synapses d'après le terme biologique correspondant), qui transportent des données numériques. Les « unités » peuvent uniquement agir sur leurs données locales et sur les entrées qu'elles reçoivent par leurs connexions.

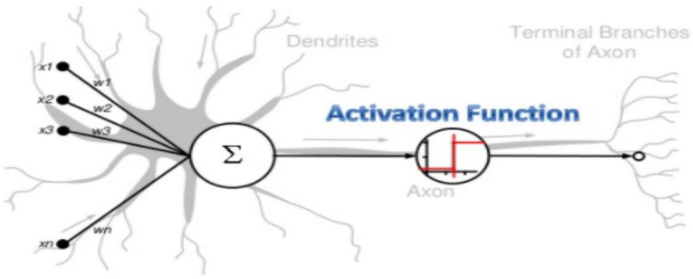


Figure 2.11 – Réseaux de neurones artificiels RNAs.

2.4.2 Modèle mathématique d'un neurone artificiel

Un neurone formel est une fonction algébrique non linéaire et bornée, dont la valeur dépend des paramètres appelés coefficients ou poids , comme il est montre sur la Figure 2.12. Par similitude avec le neurone biologique, la dynamique de neurone formel est modélisée par deux étapes [Mohamed et al., 2015] :

- Un opérateur de sommation, qui élabore le potentiel a, pour calculer le seuil d'activation θ_i.
- Un opérateur non linéaire qui donne la limite d'activation de neurone (fonction de transfert).

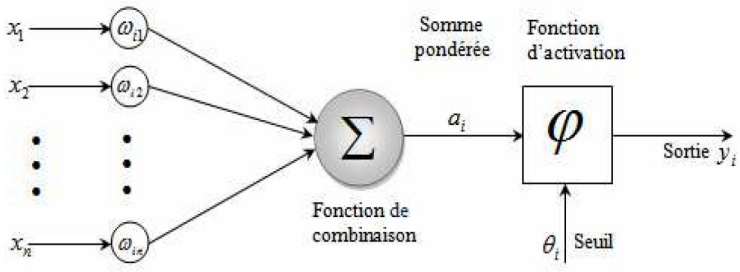


Figure 2.12 – Modèle non-linéaire d'un neurone formel.

Dans cette (Figure 2.12), x_j sont les entrées du neurone i (sortie du neurone amont j), w_{ij} c'est la valeur du poids synaptique des liaisons dirigées au neurone i vers le neurone j . Cependant, une densité positive indique une conséquence animatrice du neurone j émetteur (j) vers le neurone récepteur (i) et une densité négative représente un effet inhibiteur.

Le variable θ_i représente le seuil du neurone i qui donne la limite d'activation de neurone.

La fonction d'activation renvoie les résultats de modélisation entre la matrice des entrées et la matrice des poids synaptiques. Autrement dit, elle calcule la somme pondérée par l'équation suivante :

$$a_i = \sum\nolimits_{j=1,n} w_{ij} x_j \tag{2.35}$$

Plusieurs fonctions sont utilisées pour l'activation [Bishop et al., 1995b] : La fonction sigmoïde :

$$g(a) = \frac{1}{1 + \exp(-a)} \tag{2.36}$$

La fonction de Heavyside :

$$\begin{cases} & g(a) = 0 \quad si \quad a < 0 \quad \sin on \\ une \quad guassienne : & \\ & g(a) = \exp(\frac{-a^2}{2}) \end{cases} \tag{2.37}$$

A partir de résultats tirer de la somme pondérée, une fonction de seuillage φcalcule les paramètres de l'état de neurone, transmise aux neurones suivant, donnée comme suite :

$$y_i = \varphi(a_i - \theta_i) \tag{2.38}$$

Unité Linéaire Rectifiée(ReLU) :

$$f(x) = \max(0, x) \tag{2.39}$$

Softmax :

$$y_i(x) = \frac{\exp(a_i(x))}{\sum\limits_{k=1}^{K} \exp(a_k(x))} \tag{2.40}$$

Où

$$0 \leq y_i \leq 1 \quad et \quad \sum_{k=1}^{K} y_k = 1 \tag{2.41}$$

Les composantes de base d'un neurone formel est bien entendu le neurone artificiel. Dans le cas général un neurone formel est constitué :

- Entrées pondérées par les poids,

- Cellule de calcul l'entrée totale,
- Cellule de seuillage,
- Sortie

Chacun de ces paramètres possède une fonction ou une caractéristique bien spécifique.

2.4.3 Procédure de développement d'un réseau de neurones

Le cycle classique de développement d'un réseau de neurones peut être décomposé en sept étapes [Bouziane et al., 2008] :

- **Collecte des données**

  L'objectif de cette étape est de recueillir des données suffisantes pour constituer une base représentative.

- **Analyse des données**

  Il est souvent préférable d'effectuer une analyse des données de manière à déterminer et différencier celles ci ; cette détermination des caractéristiques a des conséquences sur la taille du réseau, ses performances et sur le temps de développement (temps d'apprentissage).

- **Séparation des bases de données**

  Il est nécessaire de disposer de deux bases de données : une base pour effectuer l'apprentissage et une autre pour tester le réseau obtenu et déterminer ses performances.

- **Choix du réseau de neurones**

  Il existe un grand nombre de types de réseaux de neurones, avec pour chacun des avantages et des inconvénients qui rendent le choix très important.

- **Mise en forme des données**

  De manière générale, les bases de données doivent subir un prétraitement afin d'être adaptées aux entrées et sorties du réseau en question.

- **Apprentissage**

  Tous les modèles de réseaux de neurones requièrent un apprentissage. Plusieurs types d'apprentissage peuvent être adaptés à un même type de réseau de neurones. Les critères de choix sont souvent la rapidité de convergence ou les performances de généralisation.

- **Validation**

  Il est nécessaire de tester le neurone sur une base de données différente de celles utilisées pour l'apprentissage. La validation permet à la fois d'apprécier les performances du système neuronal et de détecter le type de données qui pose problème.

2.4.4 Topologies de réseaux de neurones

Il existe plusieurs topologies de réseaux de neurones [Bishop et al., 1995b] :

Les réseaux multicouches : Ils sont organisés en couches, chaque neurone prend généralement en entrée tous les neurones de la couche inferieure. Ils ne possèdent pas de cycles ni de connexions intra-classe. On définit alors une « couche d'entrée », une « couche de sortie », et n « couches cachées ». Ce type de réseau est très répandu, du fait de son apprentissage aisé.

Les réseaux à connexions locales : On reprend la même structure en couche que précédemment, mais avec un nombre de connexions limité : un neurone n'est pas forcement connecté à tous les neurones de la couche précédente.

Les réseaux à connexion récurrentes : On a toujours une structure en couche, mais avec des retours ou des connexions possibles entre les neurones d'une même couche.

Enfin dans les réseaux a connexions complètes, tous les neurones sont interconnectés (Cf. Le modèle de Hopfield et la machine de Boltzmann, mis à part l'autoconnexion).

2.4.5 Perceptron multi-couches

Les Multilayer Perceptron (MLP) appartiennent aux réseaux multicouches [Bishop et al., 1995b] : ils ne possèdent donc pas de boucle de retour, ils sont < Feed-forward >. Les MLP possèdent une fonction d'activation de type sigmoïde ou de heavyside. Le MLP est une extension multicouche du perceptron, qui est un réseau à une couche, assez limité. Il utilise un algorithme d'apprentissage très répandu car facile a implémenter : la retro- propagation

du gradient, qui utilise une erreur quadratique moyenne.

2.4.6 Apprentissage des réseaux de neurones

L'apprentissage a pour but d'augmenter les performances des réseaux en se basant sur une connaissance acquise des expériences passées (prototypes). Les poids d'un RNA représentent une mémoire distribuée, cette modification affecte les poids qui relient les neurones entre eux [Saadat, 2016].

En générale, l'apprentissage se réalise sur un intervalle longue, pendant laquelle chaque prototype d'entrée (et éventuellement de sortie désirée), est présente au réseau, plusieurs fois. L'apprentissage se fait en quatre étapes Ces étapes sont répétées jusqu'à la fin d'apprentissages :

- **Etape 1** : Initialisation des poids du réseau a des petites valeurs aléatoires.
- **Etape 2** : Présentation du prototype d'entrée au RNA. Des valeurs de sortie correspondants sont ensuite calculées après propagation d'activation.
- **Etape 3** : Calcul de l'erreur : ce terme s'emploie plus volontiers pour un apprentissage supervisé ou l'erreur tient compte de la différence entre l'activation des neurones de sortie et de la sortie désiré (lie au prototype d'entrée).
- **Etape 4** : Calcul du vecteur de correction : a partir des valeurs d'erreurs, on détermine alors la correction a apporter aux poids des connexions et aux seuils des neurones.

En effet, les approches d'apprentissage se partagent principalement en trois catégories :

1. Apprentissage supervisé ;
2. Apprentissage non supervisé ;
3. Apprentissage par renforcement ou semi – supervisé.

2.4.6.1 Apprentissage supervisé

L'apprentissage supervisé ou l'apprentissage associatif, comme il est montre sur la Figure 2.13, le réseau adaptatif w compare les sorties y en utilisant une fonction d'activation

f(d, y) des entrées u. Les différentes réponses sont connues a priori. On dispose d'une base d'apprentissage qui contient un ensemble d'observation sous forme des couples entrées/sorties associées, les poids sont modifiés en fonction des sorties désirées.

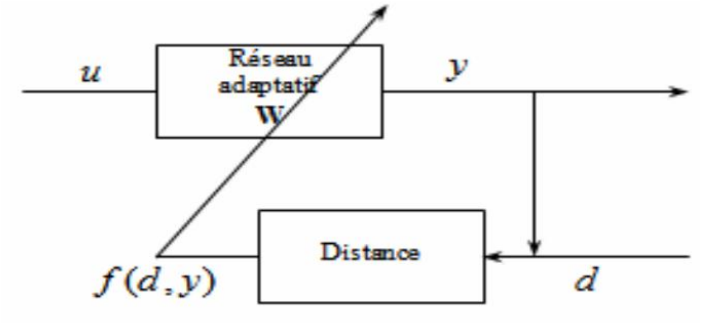


Figure 2.13 – Apprentissage supervisé.

2.4.6.2 Apprentissage non supervisé (auto organisation)

L'apprentissage auto organisation, comme il est montre sur la figure 2.14, est base sur la détermination des probabilités possible d'élaboration de modèle adéquate pour le système a modélisé. L'apprentissage auto organisation utilise les entrées de système seulement, sans le comportement de référence.

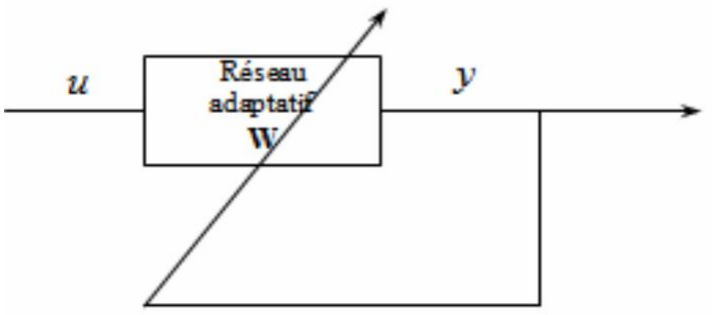


Figure 2.14 – Apprentissage auto organisation

Cet apprentissage se fait sur la base d'informations locales existant aux niveaux des neurones. Il découvre les propriétés collectives qui existe entre les données et sur lesquelles le réseau s'organise.

2.4.6.3 Apprentissage par renforcement (semi-supervisé)

Ce type d'apprentissage est moins classique que les deux derniers (ils sont les principaux). Il a d'une part en commun avec l'apprentissage supervisé la présence d'un critère qui juge l'évolution de l'apprentissage et d'autre part il ne nécessite que des entrées sans définir les sorties désirées. Comme dans l'activation non supervisé, le réseau ajuste les poids synaptiques suivant un critère de performance. Celui-ci renforce les poids du réseau si le critère y est favorable et les punit dans le cas contraire.

2.4.7 Les règles d'apprentissage

2.4.7.1 La règle de Hebb

On associe un poids à chacune des entrées du neurone, ces entrées ayant des liens avec d'autres neurones. La règle d'apprentissage, non supervisé de Hebb s'applique à ces poids. Lorsqu'un neurone est activé, il renforce les poids de ses connexions avec les neurones d'entrée qui sont simultanément activés, elle peut être modélisée comme suit [Bouziane et al., 2008] :

$$w_{ij}{}^{(t)} = w_{ij}{}^{(t-1)} + \delta x_i x_j \tag{2.42}$$

Avec x_i activation du neurone i , δ une valeur positive, et $w_{ij}{}^{(t)}$ valeur de w_{ij} à l'instant t.

2.4.7.2 La règle de Widrow-Hoff (delta)

La règle d'apprentissage de Widrow-Hoff est une règle de correction des poids permettant de rapprocher l'activation du neurone d'une valeur connue. Le principe de cette règle d'apprentissage supervisé est que le neurone connaît la valeur désirée en sortie et va chercher à adapter ses poids pour se rapprocher de cette valeur. La règle d'apprentissage de Widrow-Hoff permet à un neurone de corriger ses différents poids w_{ij} suivant cette équation :

$$w_{ij}{}^{(t)} = w_{ij}{}^{(t-1)} + \delta e_i x_j \tag{2.43}$$

Avec la valeur de sortie de i (valeur désirée de i). x_j l'activation du neurone j ; δ est une valeur positive, $w_{ij}{}^{(t)}$ la valeur de w_{ij} à l'instant t et e_i l'erreur commise par le neurone i.

2.4.7.3 La retro propagation du gradient

Principe

La retro-propagation du gradient consiste à propager « à l'envers » (de la couche de sortie vers la couche d'entrée) l'erreur obtenue sur les exemples de la base d'apprentissage. On utilise pour cela l'erreur quadratique, i.e. le carré de la différence entre ce qu'on obtient et ce qu'on désire.

Si on calcule la dérivée partielle de l'erreur quadratique par rapport aux poids des connexions (d'ou le « gradient »), il est possible de déterminer la contribution des poids à l'erreur générale, et de corriger ces poids de manière a se rapprocher du résultat souhaité. La correction se fait par itération en corrigeant plus ou moins fortement les poids par l'intermédiaire d'un coefficient x.

A l'issue d'un certain nombre d'itérations, lorsque qu'on est satisfait du classement des exemples de notre base d'apprentissage, on fixe les poids qui constituent ainsi des frontières entre les classes.

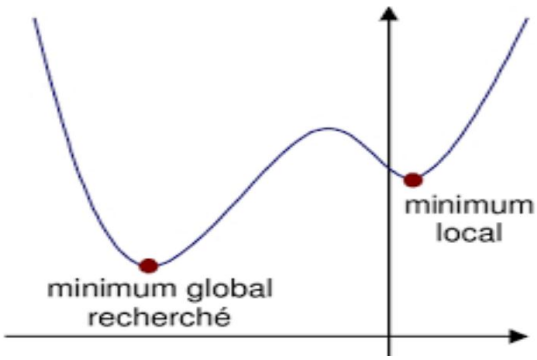


Figure 2.15 – Minimum local et minimum global recherché.

La rétro propagation de base utilise le gradient de l'erreur globale, elle a l'inconvénient de s'arrêter dans le premier minimum local rencontré.

2.5 Réseaux de neurones profonds (Deep learning)

Dans un algorithme classique d'apprentissage statistique, la première difficulté, une fois les données collectées, est de trouver des descripteurs pertinents permettant de représenter les données et de contenir de l'information utile pour la tâche souhaitée. Ainsi, pour chaque

modèle considéré, plusieurs types de descripteurs peuvent être étudiés avant de trouver une description satisfaisante des données. L'utilisation de descripteurs géométriques ou de moments statistiques sont des méthodes couramment utilisées pour obtenir des descripteurs [Jaumard-Hakoun, 2016].

Les risques sont que l'ensemble des descripteurs soit incomplet ou bien au contraire redondant. Un autre problème concerne la collecte de données, qui peuvent être de qualités variables. En outre, les échantillons de la base d'apprentissage doivent être représentatifs des données à partir desquelles le modèle est construit.

Un réseau de neurones artificiels correspond à une association en un graphe d'objets élémentaires appelés neurones formels. L'architecture de ce graphe (par exemple en couches), son niveau de complexité (par exemple la présence ou non de boucles de rétroaction), les fonctions d'activation des neurones (par exemple sigmoïde) sont des exemples de critères permettant de distinguer les réseaux de neurones. L'analogie avec un réseau de neurones biologique peut se faire en considérant les entrées d'un neurone comme des dendrites, les connexions avec les autres neurones comme des synapses, la fonction d'activation comme un noyau qui active la sortie en fonction des stimulations en entrée et la sortie du neurone comme un axone. L'apprentissage profond ou Deep Learning (par opposition au shallow learning, apprentissage peu profond) est un apprentissage réalisé sur un réseau de neurones avec plusieurs couches cachées. Le principe du Deep Learning repose sur un apprentissage hiérarchique couche par couche. Entre chaque couche interviennent des transformations non linéaires et chaque couche reçoit en entrée la sortie de la couche précédente. Dans le Deep Learning, l'extraction de descripteurs est pilotée directement à partir les données. Autrement dit, le Deep Learning repose donc sur un paradigme d'apprentissage que l'on pourrait qualifier de « supervisé par les entrées » – où les sorties attendues du modèle sont les entrées elles-mêmes. Dans ce paradigme, l'apprentissage dépend d'une fonction de coût (comme dans les apprentissages supervisés), sans avoir pour autant à fournir de données de sortie au modèle (comme dans les apprentissages non-supervisés).

L'information contenue dans des données peut être représentée de différentes manières. Par exemple, une image peut être codée comme un vecteur de valeurs d'intensité par pixel, ou bien

un ensemble de contours, de régions avec une forme particulière. Certaines représentations permettent un meilleur apprentissage de certaines tâches à partir d'exemples. Un des atouts du Deep Learning est de remplacer la détermination manuelle de descripteurs par des algorithmes d'extraction de descripteurs hiérarchiques.

Il existe plusieurs manières de construire un réseau de neurones profond, notamment le DBN (Deep Belief Network). La méthode la plus répandue afin d'entraîner efficacement un réseau de neurones profond est d'utiliser un algorithme glouton (algorithme qui recherche, étape par étape, un minimum local) d'apprentissage couche par couche par le biais de machines de Boltzmann Restreintes. Plus précisément, il s'agit d'entraîner de façon non supervisée chaque couche afin d'extraire les descripteurs principaux à partir de la distribution des données d'entrée. La première couche cachée correspond donc à une représentation de ces entrées. Cette représentation est ensuite utilisée comme entrée pour la couche suivante.

La méthode de Deep Learning peut être utilisée comme initialisation des poids et biais avant l'utilisation d'un algorithme supervisé comme la rétro-propagation du gradient (cette méthode permet de calculer le gradient de l'erreur pour chaque neurone d'un réseau de neurones, de la dernière couche vers la première. Dans l'apprentissage d'un réseau profond, la rétro-propagation joue alors de rôle de fine-tuning). L'utilisation d'une telle stratégie d'apprentissage de réseaux profonds est plutôt efficace. Il a été montré qu'initialiser les poids d'un perceptron multicouche avec un réseau profond (type Deep Belief Network, ou DBN) donnait de meilleurs résultats qu'une initialisation aléatoire.

2.5.1 Réseaux de neurones à convolutions 2D

Les réseaux de neurones à convolutions (qui seront désignés ci-après par ConvNets, pour Convolutional Neural Networks) peuvent être vus comme des réseaux de neurones MLPs particulièrement adaptés au traitement des signaux 2D. Ces réseaux ont été inspirés par les travaux de Hubel et Wiesel sur le cortex visuel chez les mammifères [Hubel and Wiesel, 1962], notamment au niveau de leur architecture ainsi que certaines de leurs propriétés comme le partage des poids. Les premiers ConvNets datent des années 1980 avec les travaux sur le Necognitron de K. Fukushima [Fukushima, 1979], mais c'est dans les années 1990 que

ces réseaux seront popularisés avec les travaux de Y. Le Cun et al. Sur la reconnaissance de caractères [LeCun et al., 1990]. Les auteurs ont proposé une série de réseaux ConvNets, baptisés LeNet (de 1 à 5), qui se basent sur trois idées architecturales clés :

- Des champs récepteurs locaux associés à des convolutions qui permettent de détecter des caractéristiques élémentaires sur l'image, formant ainsi une carte de caractéristiques.
- Un principe appelé partage des poids, qui consiste à apprendre les mêmes paramètres (ou poids) d'une convolution (et par conséquence extraire les mêmes caractéristiques) pour toutes les positions sur l'image.

Ce principe représente l'idée clé des ConvNets, puisqu'il permet de réduire considérablement la complexité en diminuant le nombre de paramètres à apprendre, et d'avoir ainsi des architectures multi-couches qui opèrent sur des entrées de grande dimension tout en étant de taille réaliste (ce qui n'était pas réalisable avec les MLPs). De plus, le partage des poids permet d'améliorer les performances en termes de généralisation du réseau, et d'être cohérent avec les études faites sur le cortex visuel [Hubel and Wiesel, 1962].

- Des opérations de sous-échantillonnage qui permettent de réduire la sensibilité aux translations, ainsi que de réduire le coût du traitement.
- Les réseaux LeNet consistent en une succession de couches qui comportent des cartes de caractéristiques et des cartes de sous-échantillonnage (cf. Figure 2.16).

L'architecture illustrée sur la Figure 2.16 correspond à un réseau de neurones qui comporte 7 couches cachées, en plus d'une couche d'entrée. Ces couches cachées peuvent être classées en deux catégories : (*i*) Les 4 premières couches sont des cartes 2D (de caractéristiques ou de sous-échantillonnage), et (*ii*) les 3 dernières sont des neurones "classiques" (similaires à celles d'un MLP), où chaque neurone est connecté à tous les neurones de la couche précédente. Sur la Figure 2.16, les cartes de caractéristiques sont représentées en gris et notées C_i, les cartes de sous-échantillonnage sont représentées en bleu et notées S_i, et les neurones sont représentés par des ronds blancs et notés N_i. Les couches C_i opèrent sur leurs entrées des convolutions 2D dont les noyaux sont les poids à apprendre, et alimentent les couches suivantes comme

pour un MLP classique. Les couches S_i appliquent un moyennage spatial (généralement de facteur 2) sur leurs entrées, puis multiplient le résultat par un poids. La succession de ces deux types de couches (la partie C1, S1, C2 *et* S2 sur la Figure 2.16) sert à extraire les informations saillantes à partir de l'image d'entrée, et à les encoder dans un vecteur au niveau de N1.

Les 3 dernières couches sont un MLP classique qui sert quant à lui à classer les données encodées. L'objectif est de construire automatiquement, à partir de l'image brute en entrée, une représentation de plus en plus haut-niveau de couche en couche. On parle alors d'apprentissage "profond" (deep learning - en anglais).

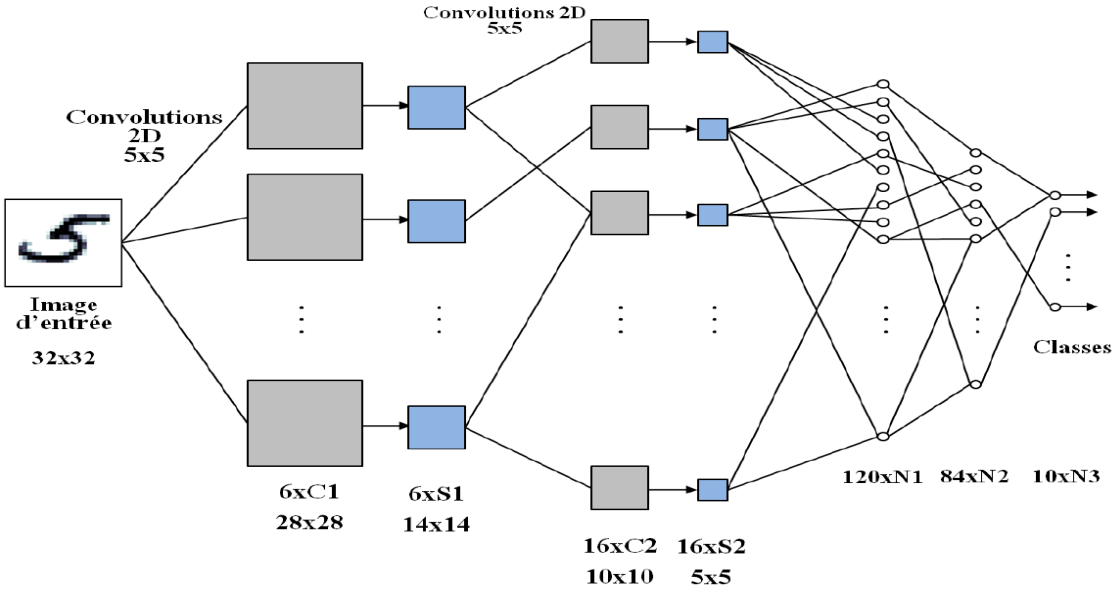


Figure 2.16 – Architecture du réseau de neurones à convolutions LeNet-5.

Les cartes de caractéristiques sont représentées en gris et notés par des C_i, les cartes de sous-échantillonnage sont représentés en bleu et notés par des S_i], et les neurones sont représentés par des ronds blancs et notés par des N_i. Les ConvNets 2D ont été appliqués avec succès à plusieurs problématiques de traitement d'images. A titre d'exemples, nous pouvons citer la reconnaissance de caractères, l'analyse de visages le suivi de gestes, la vision robotique, la détection de texte, ou encore la détection de logos TV. De plus, plusieurs solutions commerciales sont basées sur des ConvNets 2D, comme le système proposé par Google pour la détection de visages dans les images StreetView de l'application Google Earth, ou encore un OCR (Optical character recognition) proposé par Microsoft Research. Pour un

état de l'art complet des applications des ConvNets 2D en vision par ordinateur, le lecteur intéressé pourra se référer à l'article de Y. Le Cun et al [Baccouche, 2013].

2.5.2 Auto-encodeur

2.5.2.1 Principe des auto-encodeurs

Soit $D = \{(x_i\ , y_i) \in X \times Y\}_{i=1,.....,n}$ un ensemble de données entrée-sortie qu'on souhaite approximer avec un réseau de neurones. En général, une architecture neuronale comprend une couche d'entrée, une ou plusieurs couches cachées et une couche de sortie [Tian et al., 2010]. Dans la stratégie d'apprentissage des architectures profondes proposée ces dernières années, l'initialisation des poids se fait via un auto-associateur. Un auto-associateur comprend deux modules : un encodeur et un décodeur comme l'illustre la figure 2.17. L'encodeur sert à projeter les données de manière non-linéaire dans un espace de même dimension que le nombre d'unités de la couche cachée. Cette projection génère une nouvelle représentation h des données susceptibles de préserver les informations utiles pour la tâche d'apprentissage supervisé. Par exemple pour des données $x \in R^m$ (la couche d'entrée aura m unités) et la première couche cachée avec p unités, l'auto-associateur produit un code h $\in R^p$ dont la K^{me} coordonnée a pour expression :

$$h^{(k)} = f(a_k), k = 1, ..., p \quad avec \quad a_k = b_k + w_k^T \mathrm{x} \tag{2.44}$$

Dans cette équation, $w_k \in R^m$ représente le vecteur de poids, b_k le terme de biais. La fonction $f(a)$ introduit la non-linéarité dans la projection. Le module de décodage a pour but de reconstruire l'entrée x à partir du vecteur h. Cette étape de décodage est destinée à vérifier si l'encodeur a capturé l'information utile contenue dans les données. La reconstruction est basée sur l'équation suivante :

$$\hat{x}^{(j)} = g(c_j), \quad j = 1, ...m \quad ou \quad c_j = \bar{b}_j + \bar{w}_j^T h \tag{2.45}$$

Comme précédemment, $\bar{b}_j$représente le biais et $\bar{\mathrm{w}}_j$ le vecteur de poids du décodeur. La fonction g(c) introduit ici aussi une non-linéarité. Les fonctions d'activation $f$et g sont

choisies en fonction du problème traité mais elles sont souvent des fonctions tanh ou sigmoïde. L'apprentissage de l'auto-encodeur procède par minimisation d'un critère fonction de l'erreur de reconstruction $x - \hat{x}$. En général, des fonctions de coût de types moindres carrés ou entropie croisée sont usitées. Dans certaines situations, pour éviter des cas pathologiques où l'auto-encodeur réalise une simple recopie des données d'entrée (c'est-à-dire $w^T\bar{w} = I$ *ou* I est la matrice identité de taille appropriée), on impose la contrainte $w = \bar{w}^T$. Dans la suite, pour faciliter et rendre claire la présentation, les poids relatifs à l'encodeur seront désignés de manière générique par $w_{encodeur}$ et ceux relatifs au décodeur seront nommés $w_{decodeur}$.

2.5.2.2 Pré-apprentissage des Réseaux de Neurones Profonds (RNP) par auto-associateurs

Le pré-apprentissage des Réseaux de Neurones Profonds (RNP) s'effectue couche par couche. Pour chacune d'entre elles, un auto-associateur est entrainé afin de fournir les valeurs initiales des poids. L'espace projeté du précédent auto-associateur servant d'entrées au suivant. Ainsi, $q - 1$ distincts auto-associateurs doivent être entrainés pour le pré-apprentissage d'un RNP avec couches cachées. Les poids de la première couche du ième auto-associateurs servent d'initialisation au poids de la ième couche du RNP. Ceci est l'algorithme usuel de pré apprentissage des RNP.

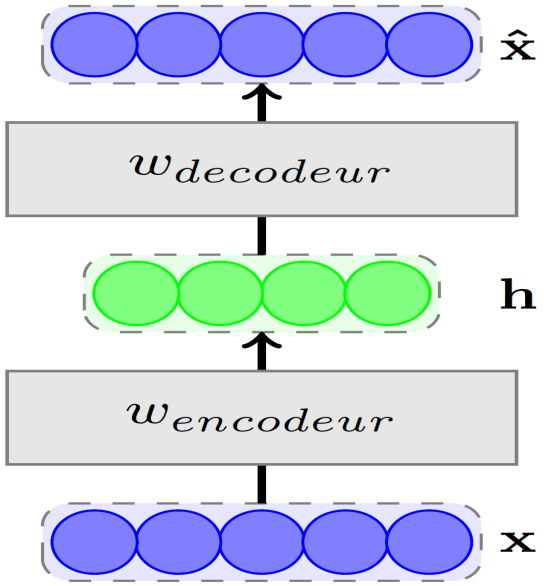


Figure 2.17 – Illustration du principe d'autoassociateur. X représente l'entrée, h le code généré par l'encodeur alors que $\widehat{x}$ définit le vecteur reconstruit par le décodeur.

2.6 Conclusion

Dans ce chapitre, nous avons introduit les principaux types d'apprentissage automatique utilisés pour la tâche de classification des données. La classification a été décrite comme la tâche qui consiste à assigner une donnée à une classe des données, en utilisant un modèle de classification (i.e. classifieur) qui lie les instances à leurs classes respectives. Afin d'apprendre ce modèle, les différents types d'apprentissage, leurs avantages et inconvénients ont été présentés.

Chapitre 3

Simulation et Résultats

3.1 Introduction

Ce chapitre est consacré à l'évaluation expérimentale des méthodes présentées comme étant une solution du problème de la classification des eaux déminéralisées utilisées dans l'industrie, et la reconnaissance automatique des chiffres .

L'objectif est d'évaluer et de valider les performances de chaqu'une des méthodes présentées. Les exigences principales d'efficacité sont formulées dans deux points essentiels. A savoir, les tests de spécification qui vérifient que le programme réalise bien la tâche pour laquelle il est conçu, et les tests de performance qui vont servir à mesurer l'efficacité avec laquelle cette tâche est remplie. Afin de mener une étude comparative permettant un choix décisif de la méthode la plus adaptée à l'application désirée, les techniques exploitées seront évaluées en fonction de leurs paramètres afin d'améliorer le taux de reconnaissance et le temps d'apprentissage, tout en minimisant l'erreur d'entraînement.

Une discussion des résultats conclura cette étude de simulation pour choisir la technique convenable. Deux exemples d'application sont prévus dans ce cadre. Le contrôle de déminéralisation de l'eau, et la reconnaissance des chiffres en tant que problème de classification. En fin de chapitre, la technique la plus performante sera retenue pour une extension multi-classe pour la reconnaissance des chiffres où pour deux classes pour la qualité de l'eau [Ladjal, 2013].

3.2 Problématique

Dans un système de reconnaissance de formes, le module le plus important est celui de la classification. Cette phase consiste à affecter chaque attribué à une classe appropriée selon une règle de décision spécifique. Pour optimiser la règle de décision, le classifieur a besoin d'un entrainement pour ajuster ses paramètres « Apprentissage ». Dans ce chapitre, on va proposer une étude comparative entre quatre (04) machines d'apprentissage, notamment, KNN, SVM, RNA, et RN profond [Duda et al., 2012]. Les tests sont appliqués sur deux différentes bases de données pour évaluer les performances de chaque machine (une base réelle pour contrôler la qualité des eaux, et l'autre synthétique pour la reconnaissance des chiffres synthétiques).

3.3 Classification des eaux industriels

3.3.1 Description de la base de données

Après avoir effectué un stage à la station de déminéralisation des eaux dans la société de la production d'électricité (SPE) dont l'objectif est de ressembler une base de données. Nous avons trouvé que cette station utilise quatre capteurs importants, permettant de mesurer le PH, la Conductivité, le TDS et l'oxygène dissous. La dimension de notre base de données est de 4×50 (4 paramètres pour 50 échantillons), ces données sont collectées durant les trois années précédentes. Les normes des paramètres citées dans le tableau ci-dessous permettent de décider si l'eau est déminéralisée ou non pour voir si elle peut être utilisée pour le refroidissement des machines de la société (les Turbines à gaz et les alternateurs) ou non.

Tableau 3.1 – Les normes des paramètres pour la déminéralisation des eaux.

PH	Conductivité	Oxygène dissous mg/l	TDS
$6.4 < PH < 9.4$	< 2800	< 8.35	< 1000

3.3.2 Caractéristiques des eaux [Ladjal, 2013]

Plusieurs paramètres peuvent utiliser pour la classification des eaux nous allons présenter les quatre paramètres utilisé dans ce travail.

3.3.2.1 Le potentiel hydrogène (pH)

Le pH mesure l'activité chimique des ions hydrogènes H+ en solution. Ce paramètre a une importance principale pour la qualité d'une eau donnée. Il indique si l'eau présente une acidité (ou alcalinité) ou une basicité trop élevée risquant de créer des conditions peu favorables à la vie au moyen d'une échelle universellement graduée de 0 à 14. Un pH inférieur à 7 correspond à une eau acide et un pH supérieur à 7 indique que l'eau est alcaline (ou basique). Un pH de 7 est donc dit neutre. Des pH faibles augmentent notamment le risque de présence de métaux sous une forme ionique toxique. Des pH élevés augmentent les concentrations d'ammoniac.

3.3.2.2 Conductivité

La conductivité est une mesure de la qualité de l'eau pouvant passer un courant électrique. Il s'agit d'une mesure indirecte de la présence des matières inorganiques solides dissoutes tels que : le chlorure, le nitrate, le sulfate, le phosphate, le sodium, le magnésium, le calcium, le fer et l'aluminium, à l'aide d'un conductivimètre. La plupart de ces sels minéraux en solution sont de bons conducteurs. On obtient une valeur en micro-siemens par centimètre (μs/cm).

3.3.2.3 Matières dissoutes totales (TDS)

Les matières dissoutes totales représentent la quantité des substances inorganiques (le sodium, le chlorure et les sulfates) dissoutes dans l'eau. Une eau à forte teneur en matières totales dissoutes acquiert un goût désagréable.

3.3.2.4 L'Oxygène Dissous

Les concentrations en oxygène dissous constituent, avec les valeurs de pH, l'un des plus importants paramètres de la qualité de l'eau. La concentration en OD varie de manière journalière et saisonnière car elle dépend de nombreux facteurs tels que la pression partielle en oxygène de l'atmosphère, la température de l'eau, la salinité, la pénétration de la lumière et l'agitation de l'eau..

Nous allons présenter quelques techniques classiques de classification supervisée : les algorithmes étudiés sont présentés dans un ordre de difficulté croissant : l'algorithme « k-

plus proches voisins », « Les machines à vecteurs de support (SVM) », « Les Réseaux de Neurones Artificiels (RNA) ». Et les machines basées sur l'apprentissage approfondi « deep learning ».

3.4 Application des techniques sur les eaux industrielles

3.4.1 La technique de classification « les k- plus proche voisine k-PPV »

L'algorithme des k-plus proches voisins (en anglais k-nearest neighbors K-NN) est une méthode d'apprentissage à base d'instances. Méthode très intuitive qui classe les exemples non étiquetés sur la base de leur similarité avec les exemples de la base d'apprentissage. Plusieurs stratégies ont été présentés pour évaluer la précision : fixer la distance Euclidienne et changer le nombre de voisinage K et vice versa. Les résultats sont présentés par les figures citées ci-dessous :

On fixe la distance Euclidienne, et on change le nombre de voisine K :

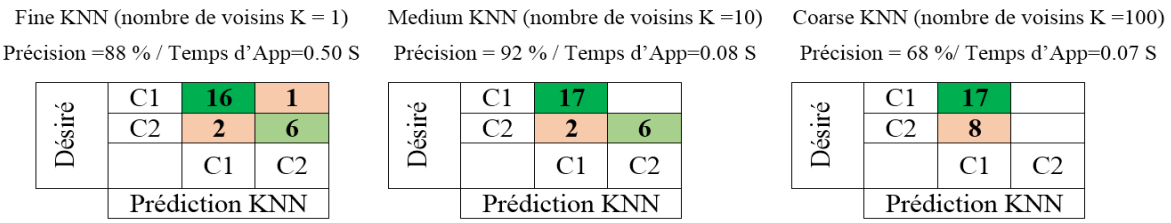


Figure 3.1 – Les Matrices des confusions.

On fixe le nombre de voisine K=10, et on change la distance :

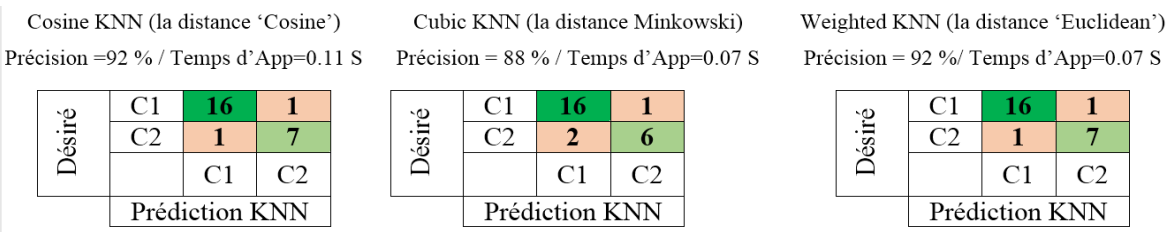


Figure 3.2 – Les Matrices des confusions.

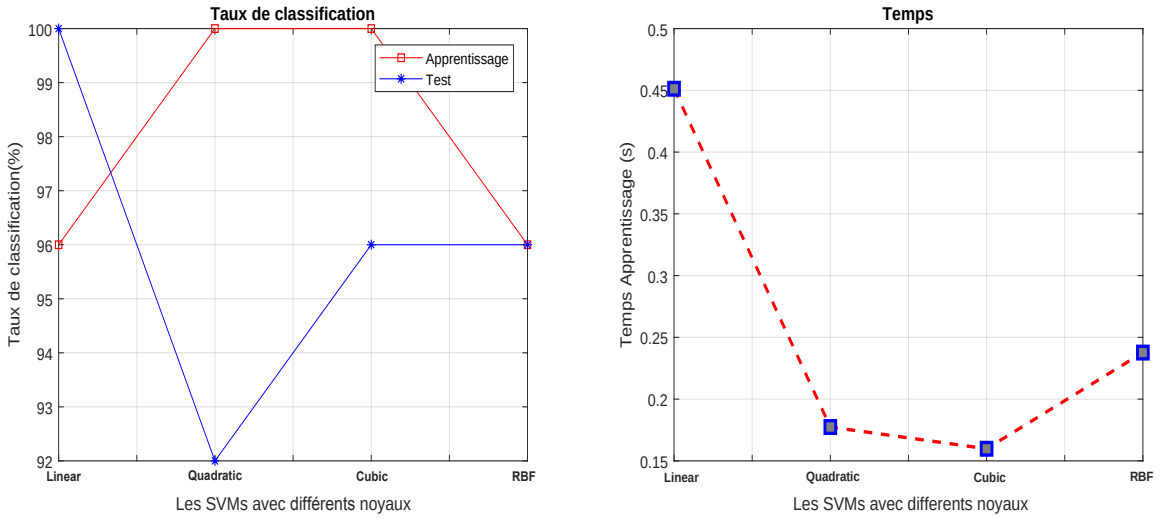
D'après les résultats, on remarque que la meilleure performance est montrée pour la

distance Euclidienne avec le nombre de K=10, et pour la méthode Cosine-NN et Weighted KNN par une valeur de 92%. Malgré la robustesse de la méthode, son taux est insuffisant.

3.4.2 La technique de classification « Les machines à vecteurs de support SVM »

Les machines à vecteurs de support ou séparateurs à vaste marge (en anglais Support Vector Machine) sont des algorithmes d'apprentissage initialement construits pour la classification. L'idée est de rechercher une règle de décision basée sur une séparation par hyperplan de marge optimale. Dans cette section, nous avons utilisé quatre types de noyaux (linéaire, quadratique, cubique et avec une fonction de base radiale) dont l'objectif est de classifier l'eau industrielle.

Les figures 3.3a et 3.3b présentent, respectivement, le taux et leurs temps de classification. Par ailleurs, les résultats montrent une rapidité de classification mais les meilleurs taux sont présentés avec les noyaux linéaire et cubique.

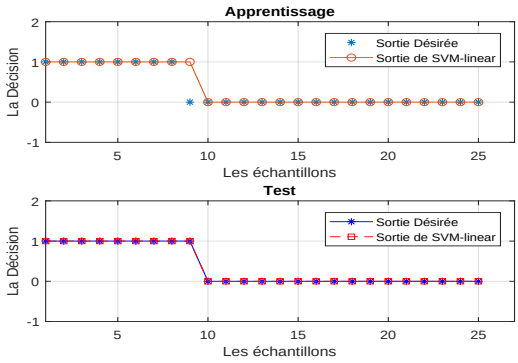


(a) Le Taux (b) Le Temps

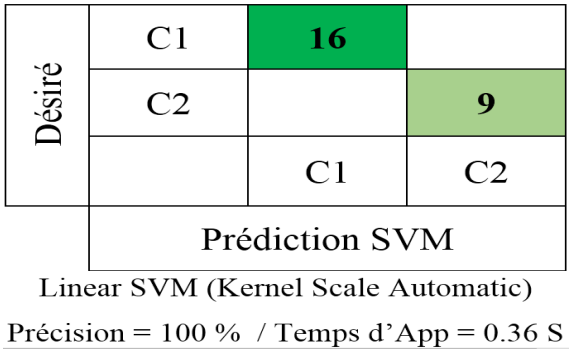
Figure 3.3 – Le Taux et le Temps de classification des données par SVM.

Ensuite, nous avons tracé les sorties désirées avec les sorties de classificateurs SVM dans les deux bases de données : l'apprentissage et le test avec leurs matrices de confusion comme suit :

3.4.2.1 SVM linéaire (Linear)



(a)



(b)

Figure 3.4 – (a) Sorties de l'apprentissage et du test pour SVM comparées avec les sorties désirées. (b) La Classification par la matrice de confusion.

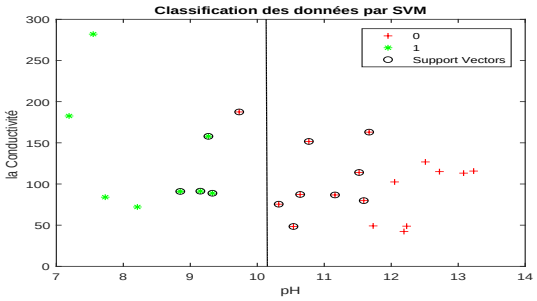
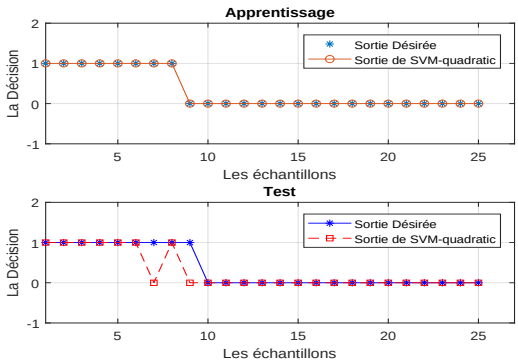


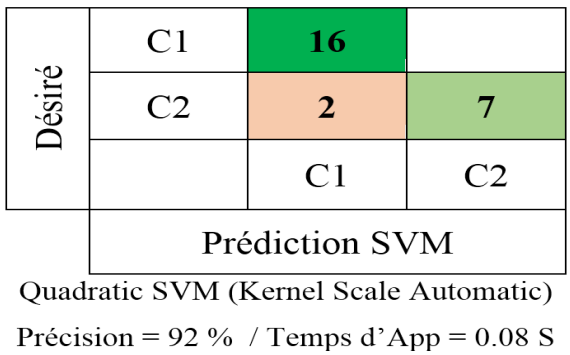
Figure 3.5 – Classification des données par SVM à noyau linéaire.

Les frontières de décision fournies par ce modèle de classificateur sont linéaires (figure 3.5)

3.4.2.2 SVM quadratique (Quadratic)



(a)



(b)

Figure 3.6 – (a) Sorties de l'apprentissage et du test pour SVM comparées avec les sorties désirées. (b) La matrice de confusion.

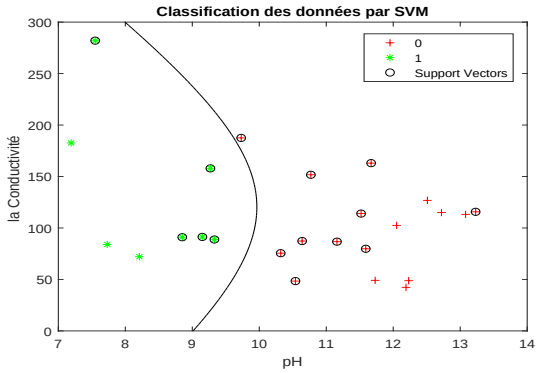
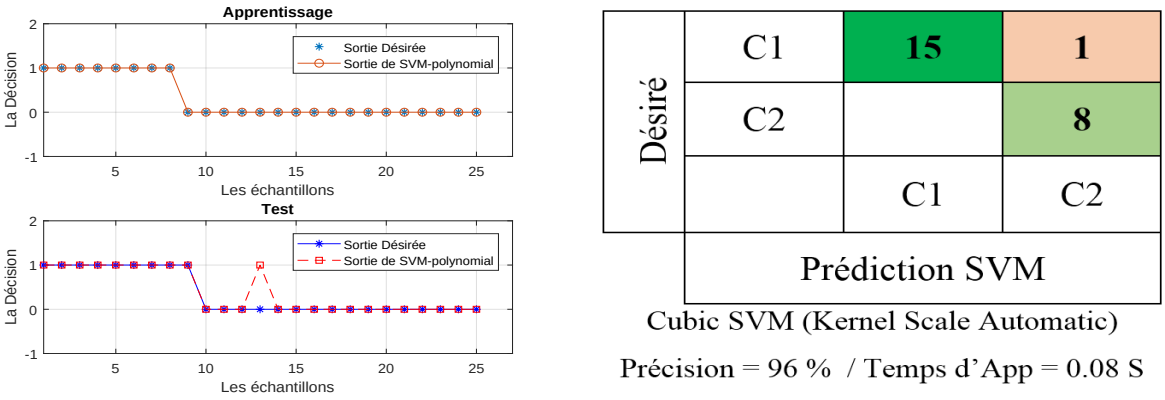


Figure 3.7 – Classification des données par SVM à noyau linéaire.

Les frontières de décision fournies par ce modèle de classificateur sont linéaires (figure 3.7)

3.4.2.3 SVM cubique (Cubic)



(a) (b)

Figure 3.8 – (a) Sorties de l'apprentissage et du test pour SVM comparées avec les sorties désirées. (b) La Classification par la matrice de confusion.

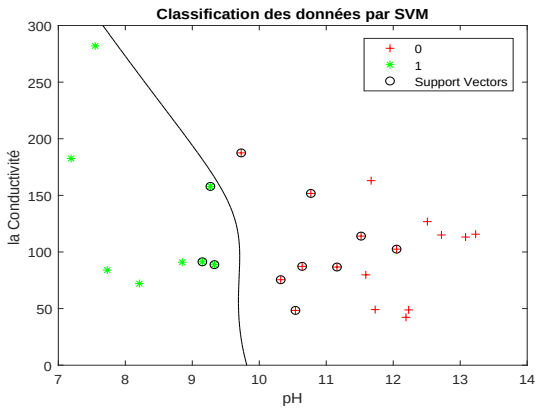
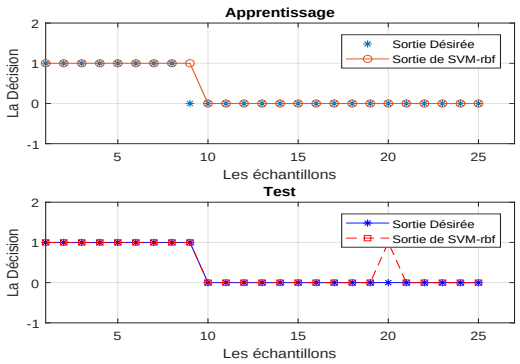


Figure 3.9 – Classification des données par SVM à noyau linéaire.

Les frontières de décision fournies par ce modèle de classificateur sont linéaires (figure 3.9)

3.4.2.4 SVM Fonction de base radiale (Radial Basis Function- RBF)



(a)

Désiré			
	C1	15	1
	C2		8
		C1	C2
	Prédiction SVM		

Fine Gaussien SVM (Kernel Scale = 0.5)

Précision = 96 % / Temps d'App = 0.08 S

(b)

Figure 3.10 – (a) Sorties de l'apprentissage et du test pour SVM comparées avec les sorties désirées. (b) La Classification par la matrice de confusion.

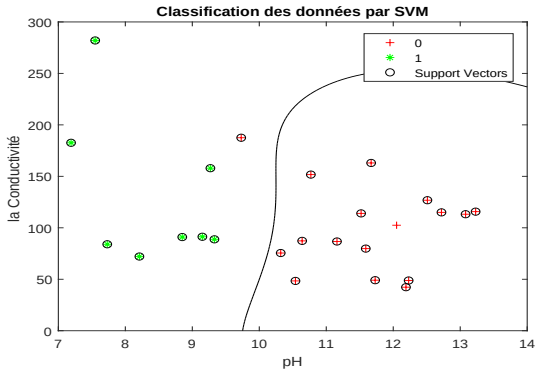


Figure 3.11 – Classification des données par SVM à noyau linéaire.

Les frontières de décision fournies par ce modèle de classificateur sont linéaires (figure 3.11)

Les Figures 3.4a, 3.6a, 3.8a et 3.10représentent la correspondance entre les sorties désirées (apprentissage et tests) et les sorties de classification en utilisant la méthode de SVM. En particulier, les noyaux utilisés dans les figures précédentes sont linéaires, quadratique, cubique et avec une fonction de base radiale. Plus précisément, le noyau linéaire présente une précision importante dans le test comme il est montré dans la matrice de confusion (figure 3.4a). Cependant, l'apprentissage présente un échantillon mal classé la figure 3.5 par un support linéaire. Par ailleurs, le noyau quadratique présente un taux d'apprentissage très suffisant comme il est montré dans la figure 3.5a. Le SVM basé sur la fonction quadratique présente une séparation des données suffisantes comme la figure 3.7, tandis que, la partie de test présente

deux échantillons mal classés. En effet, la matrice de confusion dans la figure 3.6b présente une précision de 92%. En outre, dans le cas d'un SVM avec un noyau cubique, on remarque un échantillon mal classé pour la base de test. Cependant,toute la base d'apprentissage est bien classée. Les résultats obtenus après avoir utilisé le noyau en fonction de base radiale sont présentés dans la figure 3.10a où la comparaison de sortie désirée avec la sortie de classificateur nous a permis de remarquer qu' il y a deux échantillons dans la base d'apprentissage et de test sont mal classés.

3.4.3 Les Réseaux de Neurones Artificiels RNA (MLP)

Dans ce cas, nous avons proposé l'architecture de RNA comme le montre la figure suivante :

- La couche d'entrée comporte les données expérimentale de station des eaux.
- La couche cachée comporte 5 à 25 neurones pour chaque cas avec une fonction d'activation de type 'tansig'.
- Pour la couche de sortie une seule neurone avec une fonction d'activation de type 'purelin'.

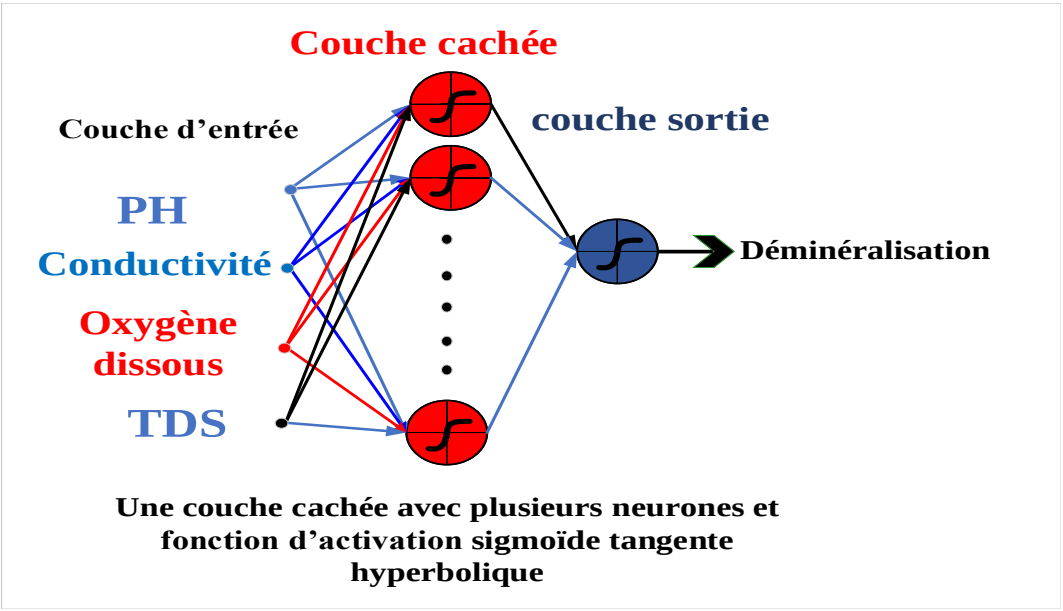


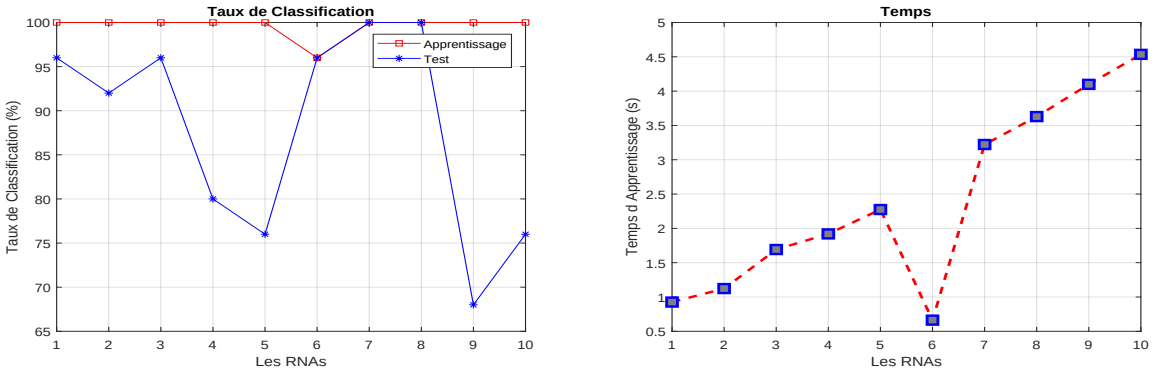
Figure 3.12 – Architecture de RNA dédié au diagnostic des eaux avec quatre entrées (PH, Conductivité, Oxygène dissous et TDS

On remarque que le réseau contient quatre paramètres dans l'entrée et une couche de sortie pour déterminer la déminéralisation des eaux, et une couche cachée qui contient un

nombre des neurones comme suit :

3.4.3.1 Résultat de RNA pour une couche cachée de cinq(05) neurones

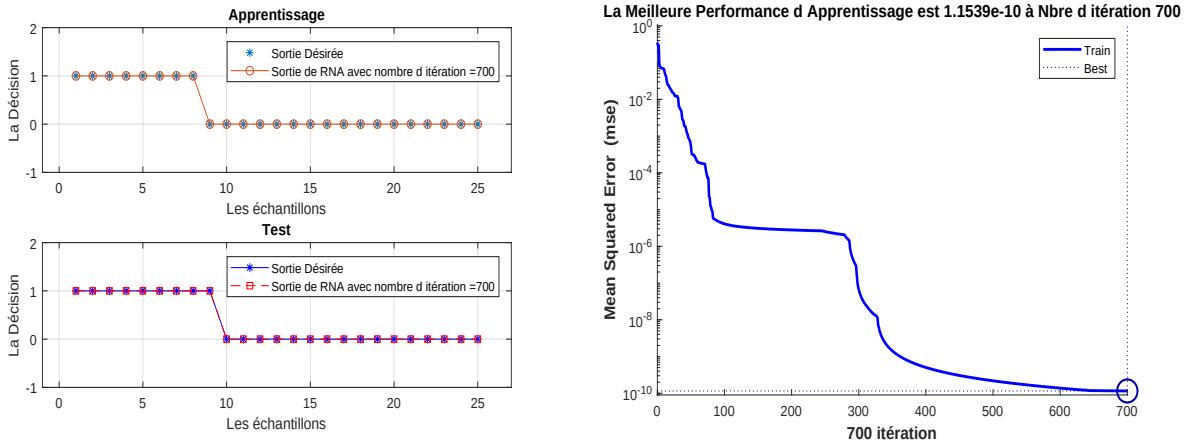
Les figures suivantes illustrent le taux de classification d'apprentissage et de test et aussi le temps de chaque réseau de neurone RNA.



(a) (b)

Figure 3.13 – (a) Le Taux de classification de l'Apprentissage et le Test pour le réseau de neurone RNA. (b) Le Temps d'Apprentissage pour le réseau de neurone RNA.

Résultat d'apprentissage, test et performance : Dans cette section, nous présentons les résultats d'apprentissage, le test et la performance de RNA avec 700 itérations qui ont été choisies d'une façon aléatoire. De plus, les résultats montrent une précision importante pour les deux étapes d'apprentissage et du test comme le montre les figures suivantes :



(a) (b)

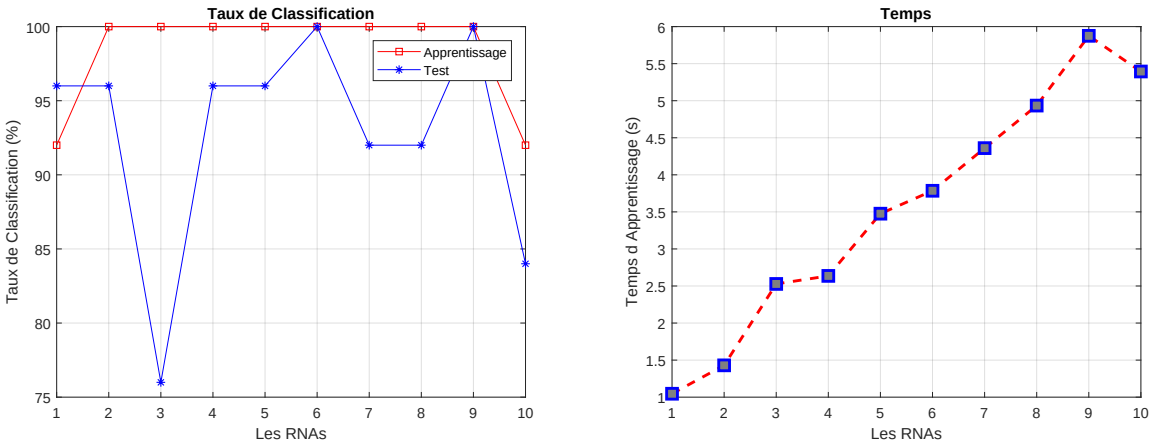
Figure 3.14 – (a) Sorties de l'apprentissage et de test pour RNA comparées avec les sorties désirées. (b) Performance d'apprentissage pour le réseau de neurone RNA.

La figure 3.14a représente une précision importante pour les sorties de l'apprentissage et

de test désirés avec la sortie de RNA. Dans ce cas, nous avons remarqué que la meilleure performance d'apprentissage est de 1.1539×10^{-10} à l'itération 700. Ce qui montre qu'il y a un apprentissage parfait, ce qui influence avantageusement sur le test. Dans la première étape, nous avons utilisé 5 neurones alors que dans les étapes suivantes, nous avons utilisé une série de plusieurs nombres de neurones : 10, 15, 20 et 25 dans l'objectif de savoir quel nombre de neurone pourrait nous donner une meilleure performance. Les résultats sont présentés ci-dessous.

3.4.3.2 Résultat de RNA pour une couche cachée de dix (10) neurones

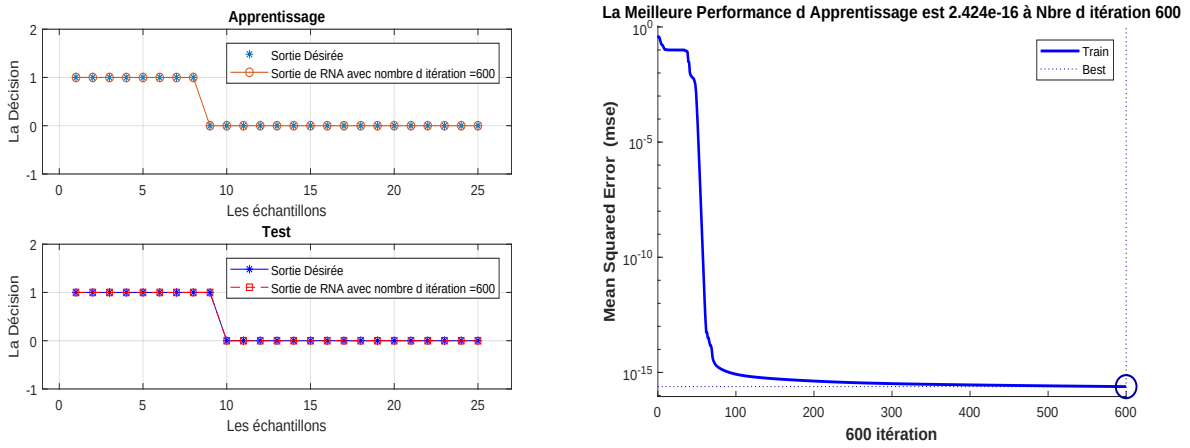
Le taux de classification d'apprentissage, test et le temps de chaque réseau de neurone RNA :



(a) (b)

Figure 3.15 – (a) Le Taux de classification de l'Apprentissage et le Test pour le réseau de neurone RNA. (b) Le Temps d'Apprentissage pour le réseau de neurone RNA.

Résultat d'apprentissage, test et performance :

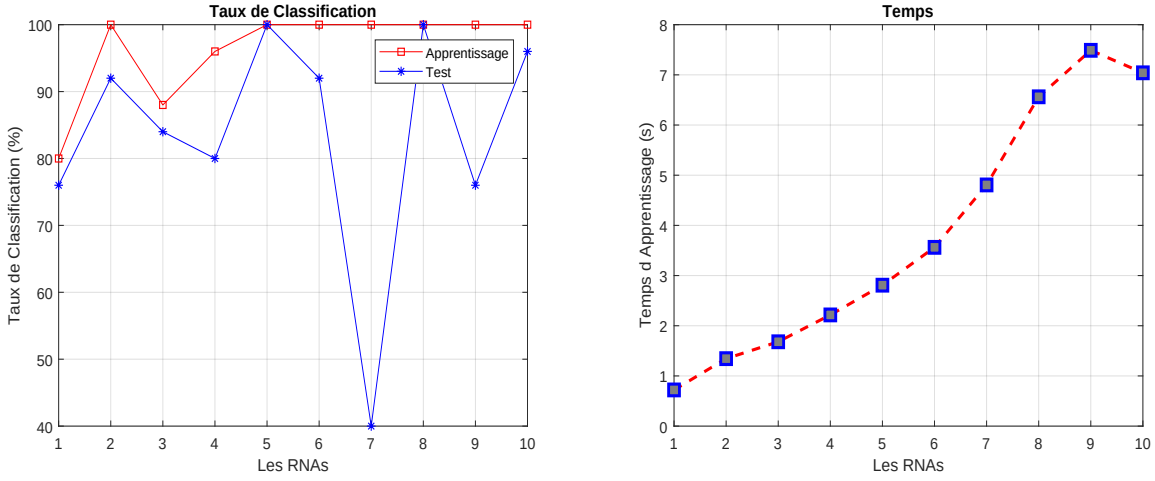


(a) (b)

Figure 3.16 – (a) Sorties de l'apprentissage et du test pour RNA comparées avec les sorties désirées. (b) Performance d'apprentissage pour le réseau de neurone RNA.

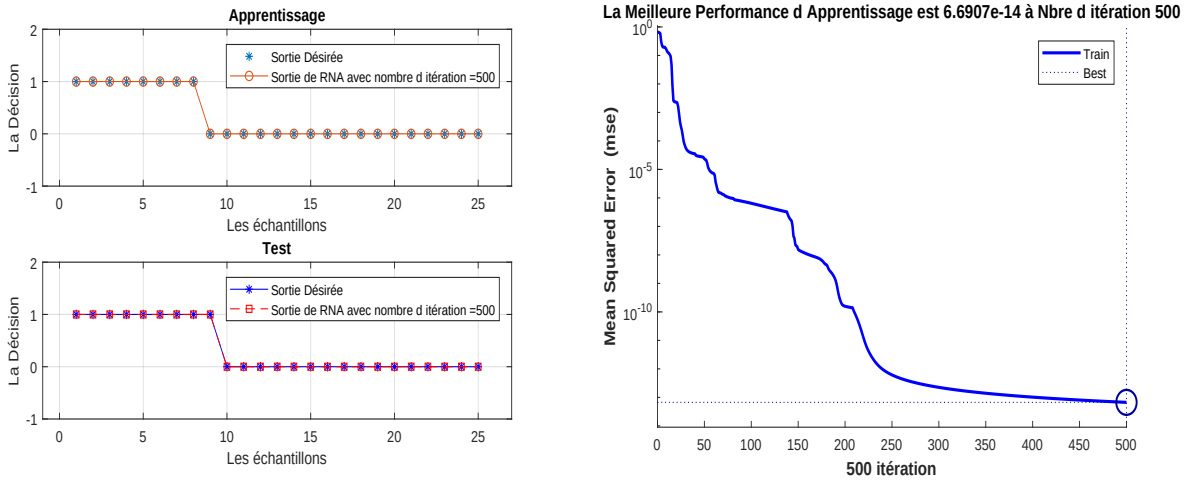
3.4.3.3 Résultat de RNA pour une couche cachée de quinze (15) neurones

Le taux de classification d'apprentissage, test et le temps de chaque réseau de neurone RNA :



(a) (b)

Figure 3.17 – (a) Le Taux de classification de l'Apprentissage et le Test pour le réseau de neurone RNA. (b) Le Temps d'Apprentissage pour le réseau de neurone RNA.

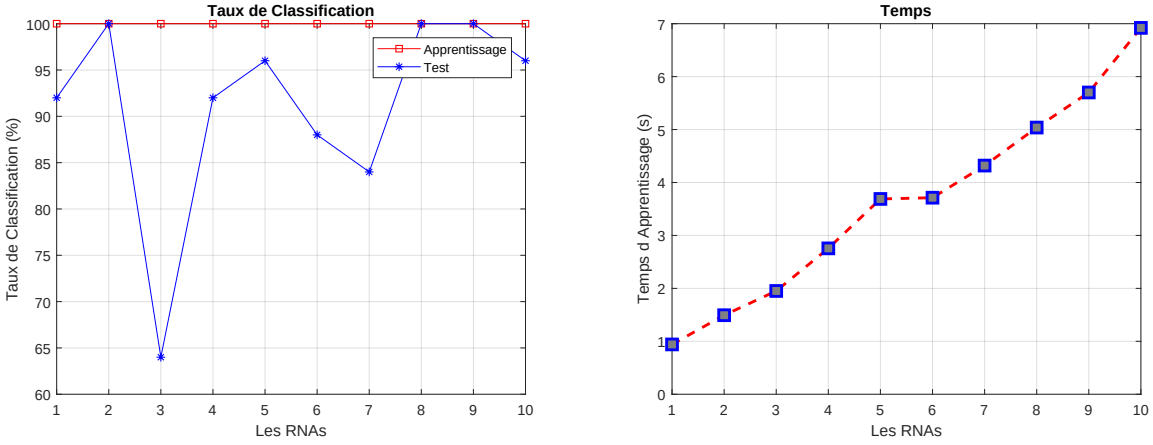


(a) (b)

Figure 3.18 – (a) Sorties de l'apprentissage et du test pour RNA comparées avec les sorties désirées. (b) Performance d'apprentissage pour le réseau de neurone RNA.

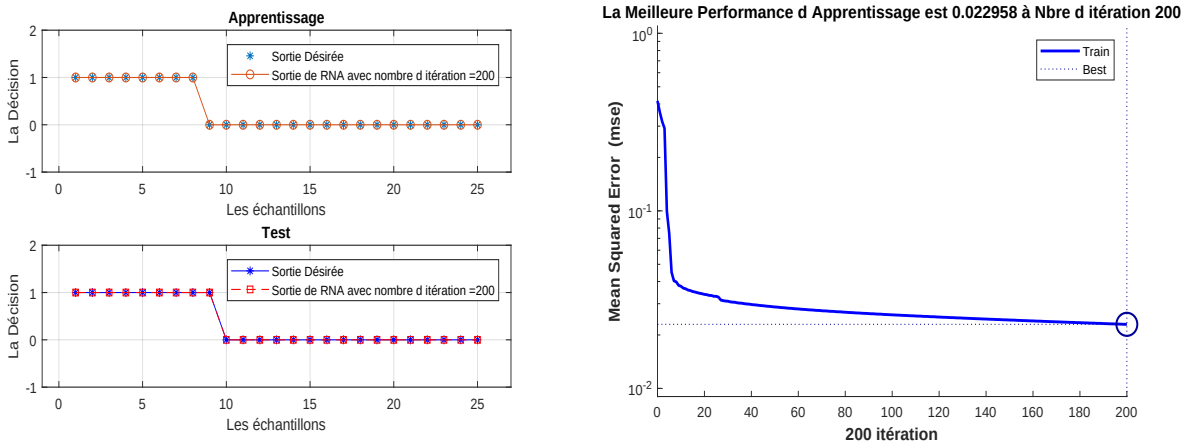
3.4.3.4 Résultat de RNA pour une couche cachée de vingt (20) neurones

Le taux de classification d'apprentissage, test et le temps de chaque réseau de neurone RNA :



(a) (b)

Figure 3.19 – (a) Le Taux de classification de l'Apprentissage et le Test pour le réseau de neurone RNA. (b) Le Temps d'Apprentissage pour le réseau de neurone RNA.

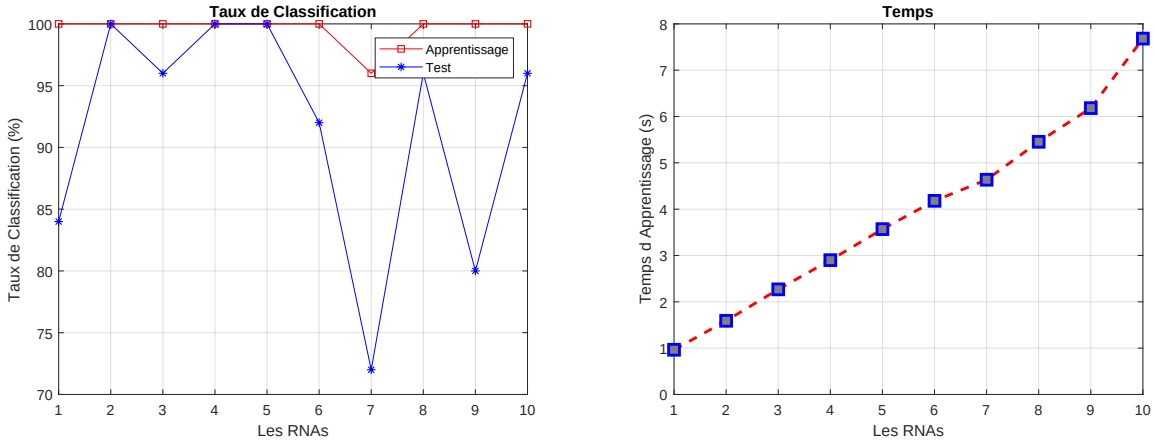


(a) (b)

Figure 3.20 – (a) Sorties de l'apprentissage et du test pour RNA comparées avec les sorties désirées. (b) Performance d'apprentissage pour le réseau de neurone RNA.

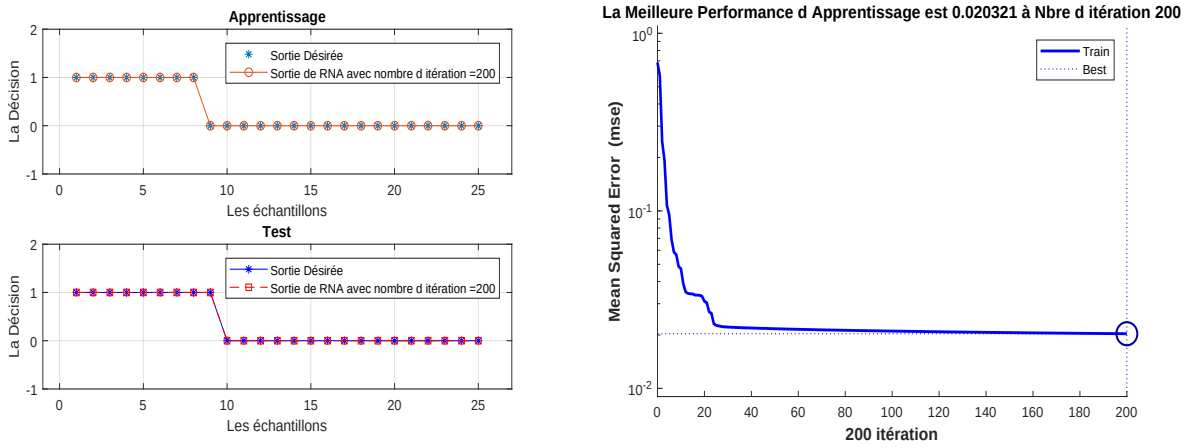
3.4.3.5 Résultat de RNA pour une couche cachée de vingt-cinq (25) neurones

Le taux de classification d'apprentissage, test et le temps de chaque réseau de neurone RNA :



(a) (b)

Figure 3.21 – (a) Le Taux de classification de l'Apprentissage et le Test pour le réseau de neurone RNA. (b) Le Temps d'Apprentissage pour le réseau de neurone RNA.



(a) (b)

Figure 3.22 – (a) Sorties de l'apprentissage et du test pour RNA comparées avec les sorties désirées. (b) Performance d'apprentissage pour le réseau de neurone RNA.

D'après les résultats obtenus, nous avons enregistré les erreurs 1.1539×10^{-10}, $2,42 \times 10^{-16}$, $6,69 \times 10^{-14}$, 0.0229 et 0.02032 avec les nombres des itérations 700, 600, 500, 200 et même 200 qui correspondent aux neurones 5, 10, 15, 20 et 25. Alors, nous remarquons que le nombre d'itération diminue avec l'augmentation de nombre de neurones. Pour l'erreur, nous ne pouvons pas juger leurs variations à cause de l'initialisation aléatoire dans l'environ Matlab. Mais, nous remarquons que le nombre de neurones de 20 et 25 donne moins de performance. Nous avons obtenu ces résultats après plusieurs essais sous Matlab, et la possibilité d'avoir une même précision reste toujours très faible.

Les figures 3.13a, 3.15a, 3.17a, 3.19a et 3.21a présentent les taux de classification des ensembles des tests effectués sur les différentes structures des RNA de type MLP avec différents nombres de neurones de 5 à 25. Alors, d'après les résultats obtenus, nous avons constaté que l'augmentation de nombre de neurones implique la diminution de nombre des itérations appliquées sur les RNAs, ce qui permet d'obtenir un meilleur taux de classification Les Figures 3.13b, 3.15b, 3.17b, 3.19b et 3.21b présentent les temps d'apprentissage pour les RNAs proposés. Ces figures montrent que l'augmentation de nombre de neurone cause un temps d'apprentissage important.

Ensuite, nous avons proposé l'apprentissage approfondi en utilisant l'autoencodeur pour améliorer la bonne classification ainsi que le temps d'apprentissage. En effet, le RNA de type MLP a besoin de temps et de nombre de neurones important pour avoir les meilleures

performances. L'exemple le plus significatif est le cas de RNA avec une seule couche cachée de 25 neurones. Néanmoins, avec l'autoencodeur on peut avoir des résultats significatifs par une RNA de type perceptron (sans couche cachée et un seul neurone dans la sortie).

3.4.4 Deep learning (Auto-encoder)

Comme nous l'avons défini dans les sections précédentes(voir la section 2), les autoencodeurs sont des réseaux de neurones à trois couches, une couche de sortie, une couche d'entrée, et une seule couche cachée. Autre, l'autoencodeur est basé sur l'apprentissage non supervisé où l'algorithme d'apprentissage utilise la fonction d'erreur quadratique. La figure suivante montre la structure proposée dont l'objectif est de classifier les eaux industrielles.

Réseau 1 (Autoencodeur 1) : comporte une couche d'entrée et une couche de sortie de 4 neurones avec une couche cachée de 25 neurones. Les fonctions d'activation de couche cachée (Encodeur) est *'logsig'* et pour la sortie *'purelin'*.

Réseau 2 (Autoencodeur 2) : comporte une couche d'entrée et une couche de sortie de 25 neurones avec une couche cachée de 4 neurones. Les fonctions d'activation de couche cachée (Encodeur) est *'logsig'* et pour la sortie *'purelin'*.

Réseau 3 (perceptron) : comporte une couche d'entrée de 4 neurones, et une couche de sortie de 1 neurone. La fonction d'activation de couche de sortie est *'radbas'*.

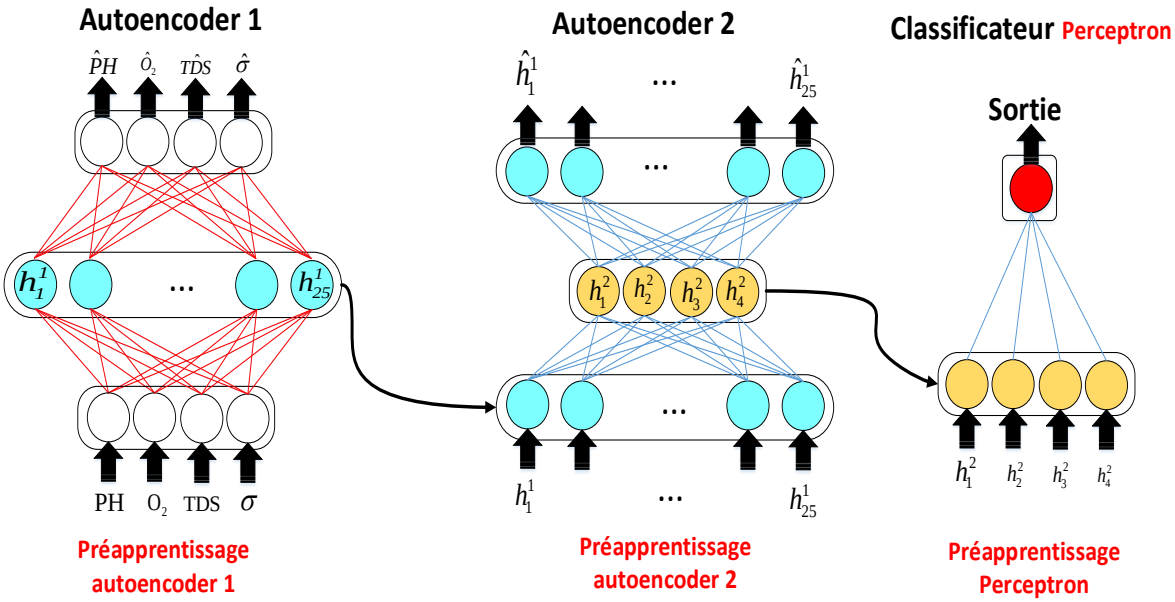
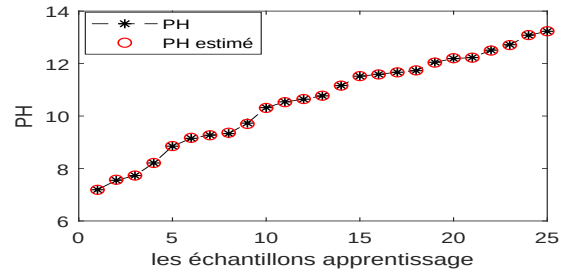
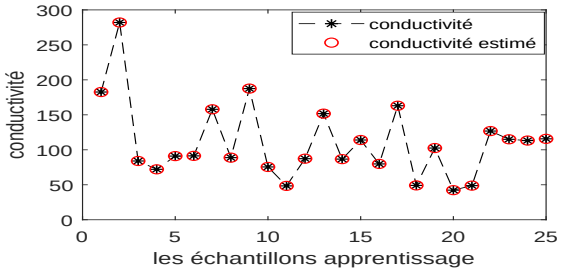


Figure 3.23 – La structure de classification basée sur l'autoencodeur empilé.

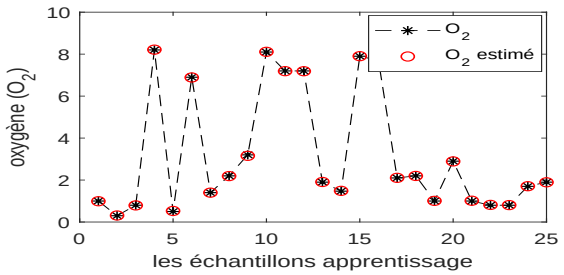
Dans cette application, nous avons utilisé quatre paramètres importants comme des entrées à la première autoencodeur dont l'objectif est d'obtenir une sortie estimée avec précision très suffisante comme le montrent les figures suivantes :



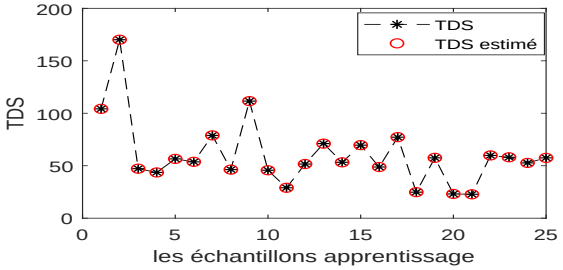
(a) PH



(b) conductivité

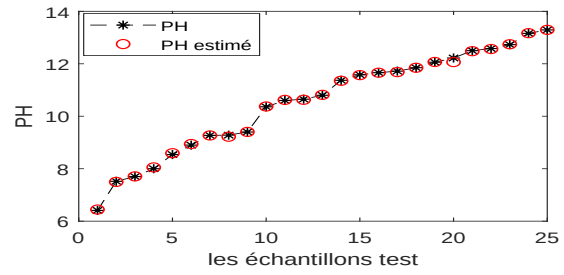


(c) O_2 dessous dans l'eau

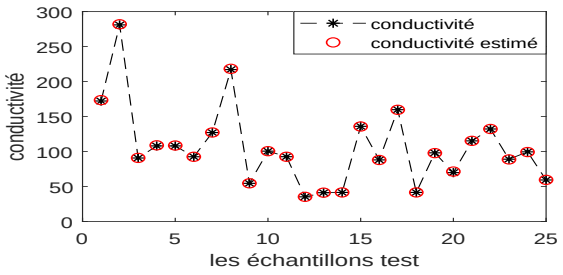


(d) TDS

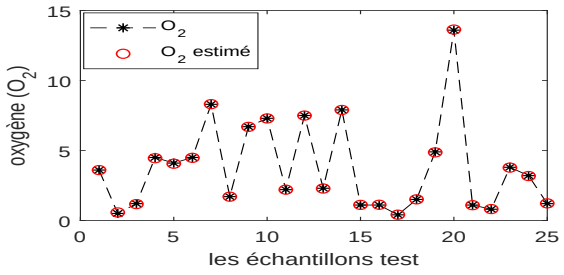
Figure 3.24 – Les données utilisées pour l'étape d'apprentissage des entrées et leurs reconstructions. Résultats obtenus après le préapprentissage auto-encodeur 1.



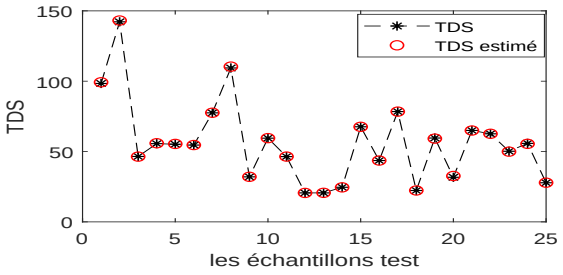
(a) PH



(b) conductivité



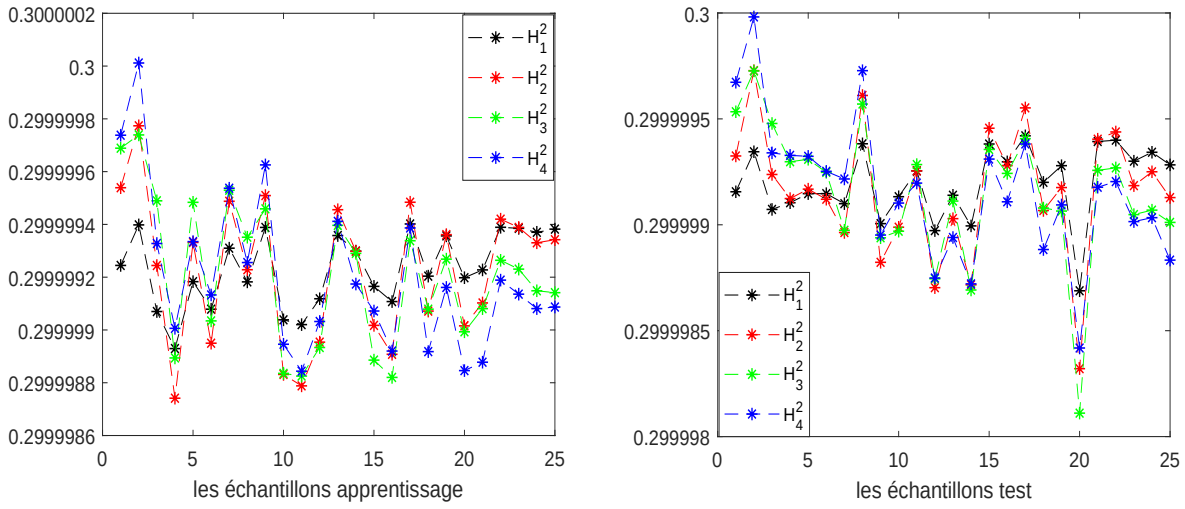
(c) O_2 dessous dans l'eau



(d) TDS

Figure 3.25 – Les données utilisées pour l'étape de Test des entrées et leurs reconstructions. Résultats obtenus après le préapprentissage auto-encodeur 1.

Dans la deuxième étape, on fait un appel au deuxième autoencodeur. Ce dernière est utilisé pour avoir de nouvelles caractéristiques. De plus, ces caractéristiques sont l'encode de le deuxième autoencodeur par les valeurs de vecteur du premier autoencodeur. Les résultats cités dans les figures ci-dessous montrent que les nouvelles caractéristiques sont bien séparées ; ce qui implique une très bonne précision si on utilise un seul neurone présenté par le réseau perceptron dans la partie suivante :



(a) (b)

Figure 3.26 – La base des données générée par le préapprentissage autoencodeur 2 (a) Apprentissage, (b) Test.

3.4.4.1 Réseau de neurones profond

Dans cette section, nous avons présenté la structure finale de réseau de neurone profond. Cette structure est utilisée pour la classification des eaux industrielles. Deux étapes de test sont utilisées pour atteindre les meilleures performances :

Test avant l'apprentissage affiné de l'ensemble de réseau profond

Une fois les couches cachées des autoencodeurs 1 et 2 sont ajustées avec la condition de la fonction d'objectif basé sur l'erreur quadratique moyen, nous avons testé le réseau profond par les deux bases d'apprentissage et de test. La figure ci-dessous présente la structure générale de réseau de neurone profond.

Les Figures suivantes présentent les résultats de comparaison entre la sortie désirée et la sortie de réseau profond, ainsi que les matrices de confusion de la base d'apprentissage et de

test. nous remarquons que les résultats d'apprentissage sont très suffisants à 100% comme le montre leur matrice de confusion. Par contre, la phase de test montre une confusion de 4%.

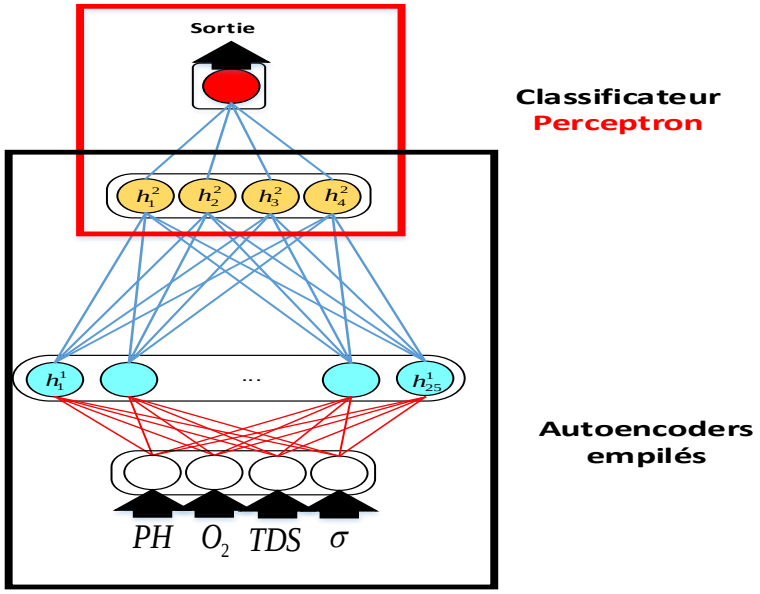
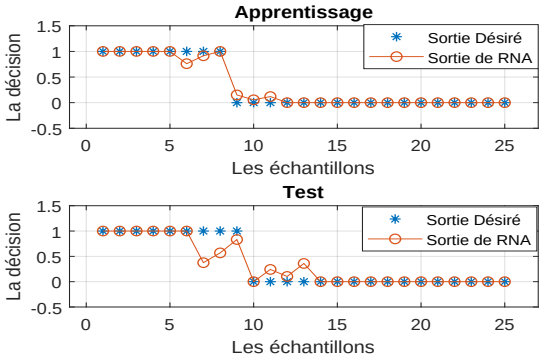
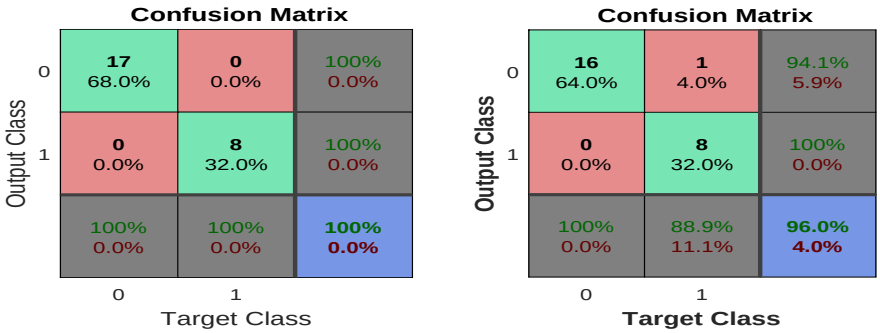


Figure 3.27 – Réseau de neurone profond proposé pour le contrôle de qualité des eaux industrielles.



(a) Résultats de comparaison entre la sortie désirée et la sortie de réseau profond



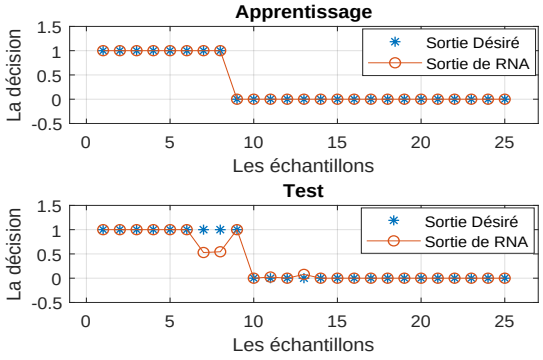
(b) Matrice de confusion pour la base d'apprentissage

(c) Matrice de confusion pour la base de test

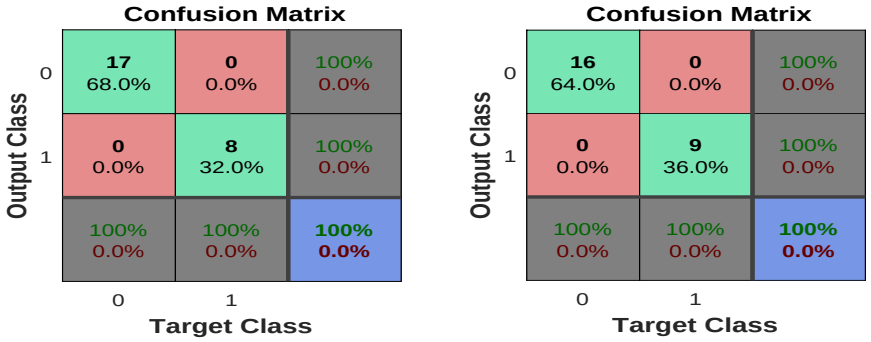
Figure 3.28 – Résultats de classification de l'eau industrielle avant l'apprentissage affiné de réseau profond.

Test après l'apprentissage affiné de l'ensemble de réseau profond :

Dans cette section, nous avons fait un apprentissage affiné pour l'ensemble de réseau profond. Cette approche montre la technique d'apprentissage approfondi et leur effet sur l'amélioration des performances de classification. Les figures suivantes montrent le résultat obtenu et le taux de classification élevé à 100% pour les deux bases d'apprentissage et de test.



(a) Résultats de comparaison entre la sortie désirée et la sortie de réseau profond



(b) Matrice de confusion pour la base d'apprentissage

(c) Matrice de confusion pour la base de test

Figure 3.29 – Résultats de classification de l'eau industrielle après l'apprentissage affiné de réseau profond.

3.5 Classification des chiffres

La reconnaissance des chiffres est souvent considérée comme l'un des problèmes d'apprentissage. Notre objectif est de développer un système de reconnaissance des chiffres synthétiques dans le cadre d'un apprentissage automatique. La base des données utilisée dans ce travail est constituée des images synthétiques des chiffres qui se trouve dans la bibliothèque Matlab. Par ailleurs, cette base est construite par une technique de changement des angles.

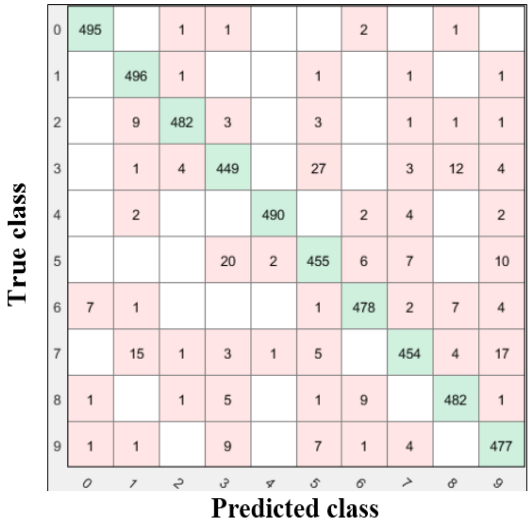
Description de la base de données utilisée

Cette base des données est entièrement synthétique, pour les apprentissages et les tests. Les images synthétiques ont été générées en appliquant des transformations affines aléatoires à des images numériques crées en utilisant différentes polices. On a 10000 images, chaque image contient des différents types de chiffres qui varient de 0 jusqu'à 9. Chaque image numérique est de 28 par 28 pixels, et il existe 5000 exemples d'apprentissage et même autres pour le test.

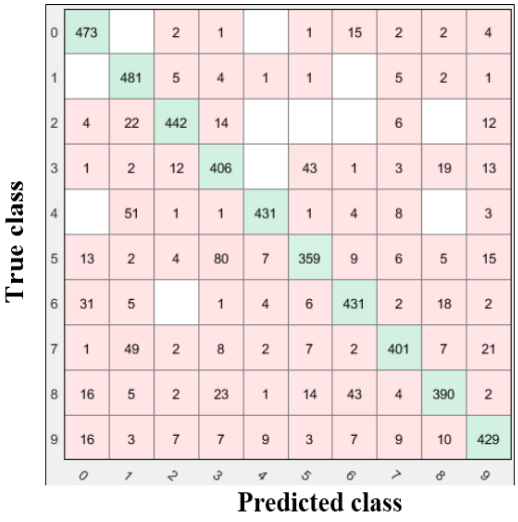
3.6 Application sur la base des chiffres

3.6.1 K plus porches voisins k-PPV (k-nearest neighbors KNN)

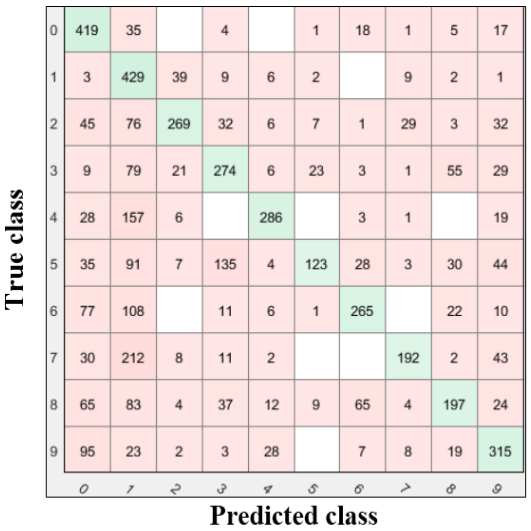
Les tableaux ci-dessous représentent les résultats de reconnaissance des chiffres synthétique avec la technique KNN. Ces résultats varient avec le changement de type de KNN. Pour le type Coarse, quelques chiffres sont mal reconnus (Précision=55.4 %). classifieur dans ce cas là n'arrive pas à bien classifier certains chiffres. Pour les types Cubic, Cosine et Meduim, il y a un faible changement dans le taux de reconnaissance par rapport au type coarse. Pour le Weighted-K-NN, nous avons relevé une faible amélioration mais le taux de bonne classification reste toujours la meilleure par rapport aux 03 K-NN précédents. Le meilleur résultat est obtenu avec Fine-K-NN (Précision=95.2 %).



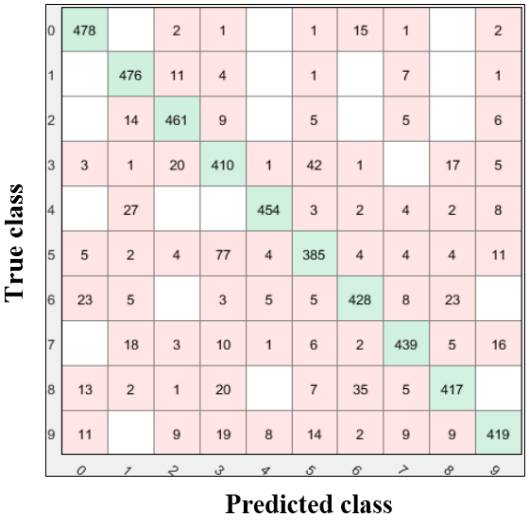
(a) Fine (K=1, Précision=95.2 %, temps =46.57S, distance Euclidienne)



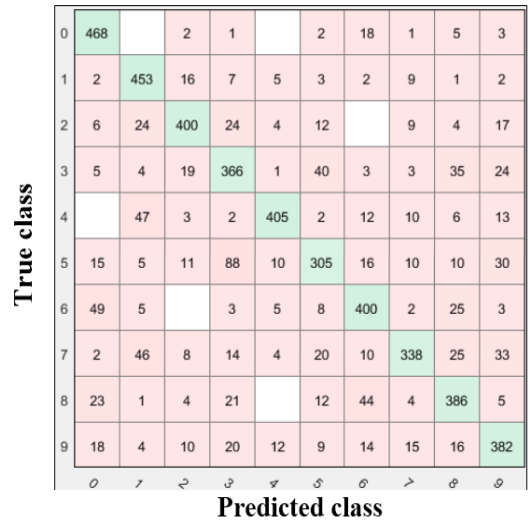
(b) Medium (K=10, Précision=84.9 %, temps =45.13S, distance Euclidienne)



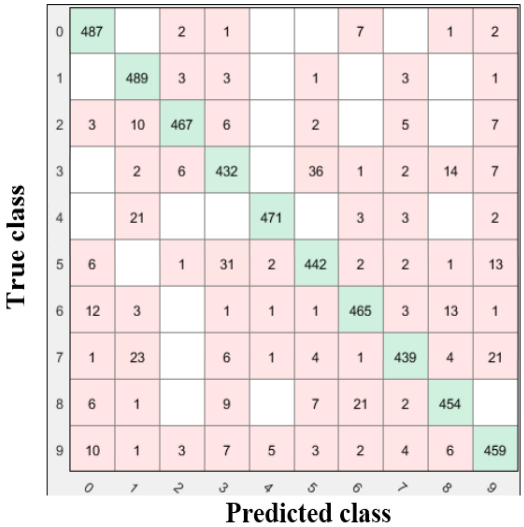
(c) Coarse (K=100, Précision=55.4 %, temps =46.27S, distance Euclidienne)



(d) Cosine (K=10, Précision=87.3%, temps =44.41S, distance Cosine)



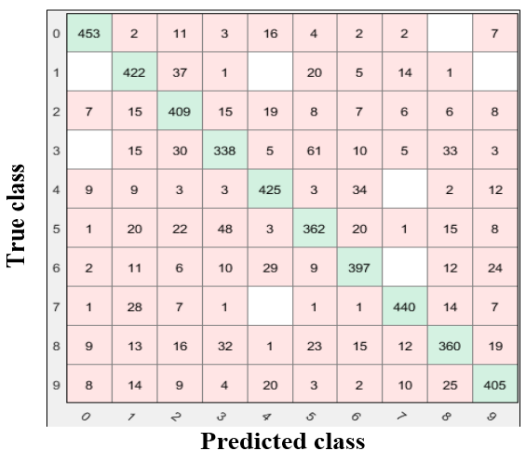
(e) Cubic (K=10, Précision=78.1 %, temps =798.23S, distance Minkowski)



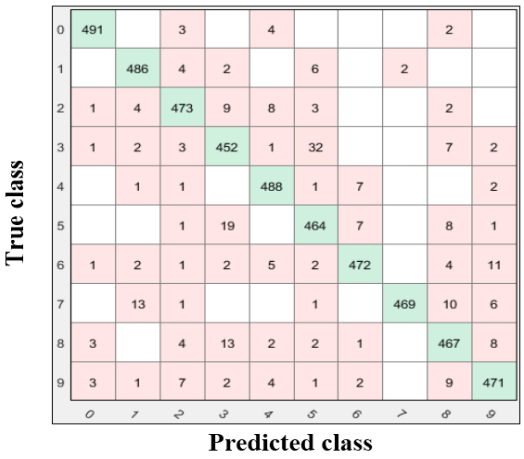
(f) Weighted (K=10, Précision=92.1 %, temps =40.25S, distance Euclidienne)

Figure 3.30 – Les matrices des confusions par la technique KNNs.

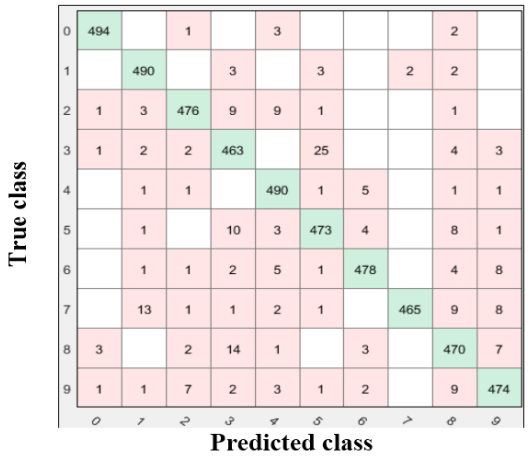
3.6.2 Les machines à vecteurs de support (Support Vector Machine SVM)



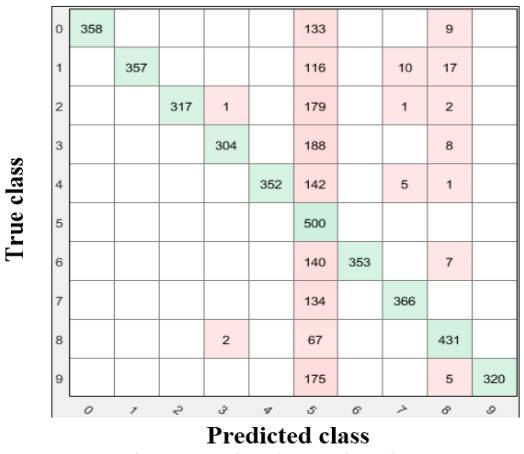
(a) Linear (Kernel scale Automatic, Précision=80.2 %, temps =89.17 s)



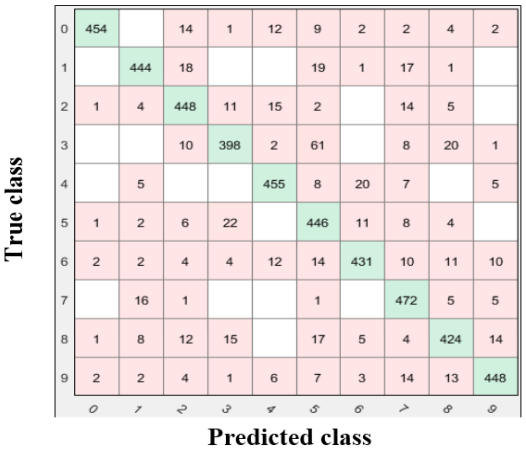
(b) Quadratic (Kernel scale Automatic, Précision=94.7 %, temps =97.59 s)



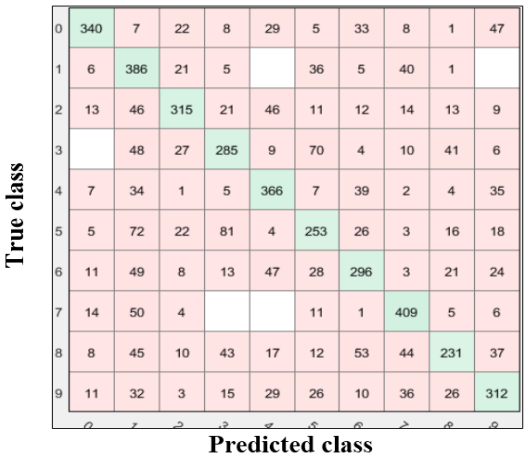
(c) Cubic (Kernel scale Automatic, Précision=95.5 %,temps=104.16 s))



(d) Fine Gaussien (Kernel scale=7, Précision=73.2 %, temps =356.06 s)



(e) Cubic Medium Gaussien (Kernel scale =28, Précision=88.4 %, temps =159.7 s)



(f) Coarse Gaussien (Kernel scale=110, Précision=63.9 %, temps =246.15 s)

Figure 3.31 – Les matrices des confusions par la technique SVMs.

Les tableaux ci-dessus représentent les résultats de reconnaissance des chiffres synthétique avec la technique SVM. Ces résultats évoluent avec le changement des noyaux (Linear, Quadratic, Cubic, Fine Gaussien, Medium Gaussien, Coarse Gaussien).

Les meilleurs résultats sont présentés avec les noyaux cubic (Précision=95.5 %), et nous avons aussi une autre précision importante par le noyau quadratique (Précision=94.7 %). D'autre part, les autres noyaux présentent des faibles précisions de 63.9% à 83.4%.

Alors, nous clairement remarquons que les deux techniques KNN et SVM présentent de faibles précisons. Dans la partie suivante, nous avons opté pour le RNA de type MLP.

3.6.3 Les Réseaux de Neurones Artificiels RNA (MLP)

Classiquement, les couches d'un réseau de neurones sont complètement connectées, c'est-à-dire que la valeur d'un neurone d'une couche n va dépendre des valeurs de tous les neurones de la couche $n-1$. Ainsi le nombre de connexions (et donc de poids, de paramètres) peut être très grand. Par exemple pour une image de taille 28×28, la dimension de l'entrée d'un MLP est de 784. Si la couche cachée comporte 100 neurones, alors le nombre de paramètres de cette couche est de $100 \times 784 = 78400$. Le nombre de paramètres va ainsi augmenter exponentiellement avec la dimension de l'entrée (des images). Cette grande complexité du réseau rend la phase d'apprentissage nécessite un nombre d'échantillons important,mais malheureusement ce n'est pas toujours disponible. Le réseau va donc avoir tendance à faire un sur-apprentissage, et proposera donc une mauvaise capacité de généralisation.

Un autre inconvénient de MLP pour une classification des images est qu'il n'est pas bien adapté à l'invariance en transformations, ce qui arrive très souvent avec des images (légères translations, rotations ou distorsions).

Finalement, les MLP ne prennent pas en compte la corrélation entre les informations des images, ce qui influence négativement sur les performance globales du système de reconnaissance des formes.

3.6.4 Apprentissage d'un réseau de neurones profond pour la classification de chiffres

dans cette section nous allons présenter deux types de RNA pour améliorer notre résultats.

3.6.4.1 Classification des chiffres en utilisant l'autoencodeur empilé

A cause des inconvénients des techniques KNN, SVM et RNA de type MLP qui présentent une mauvaise performance, nous avons proposé la structure suivante :

Réseau 1 (Autoencodeur 1) : comporte une entrée et une sortie de 748 avec une couche cachée de 200 neurones. Les fonctions d'activation de couche cachée (Encodeur) est 'logsig' et pour la sortie 'purelin'.

Réseau 2 (Autoencodeur 2) : comporte une entrée et une sortie de 200 avec une couche cachée de 20 neurones. Les fonctions d'activation de couche cachée (Encodeur) est 'logsig' et pour la sortie 'purelin'.

Réseau 3 (sofmax) : comporte une entrée de 20 et une sortie de 10 neurones. La fonction d'activation de couche de la sortie est softmax. On peut voir le tableau ci-dessous :

Tableau 3.2 – Représentation de la sortie de réseau profond

	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
	1	0	0	0	0	0	0	0	0	0
	0	**1**	0	0	0	0	0	0	0	0
	0	0	**1**	0	0	0	0	0	0	0
	0	0	0	**1**	0	0	0	0	0	0
Le chiffre	0	0	0	0	**1**	0	0	0	0	0
	0	0	0	0	0	**1**	0	0	0	0
	0	0	0	0	0	0	**1**	0	0	0
	0	0	0	0	0	0	0	**1**	0	0
	0	0	0	0	0	0	0	0	**1**	0
	0	0	0	0	0	0	0	0	0	**1**

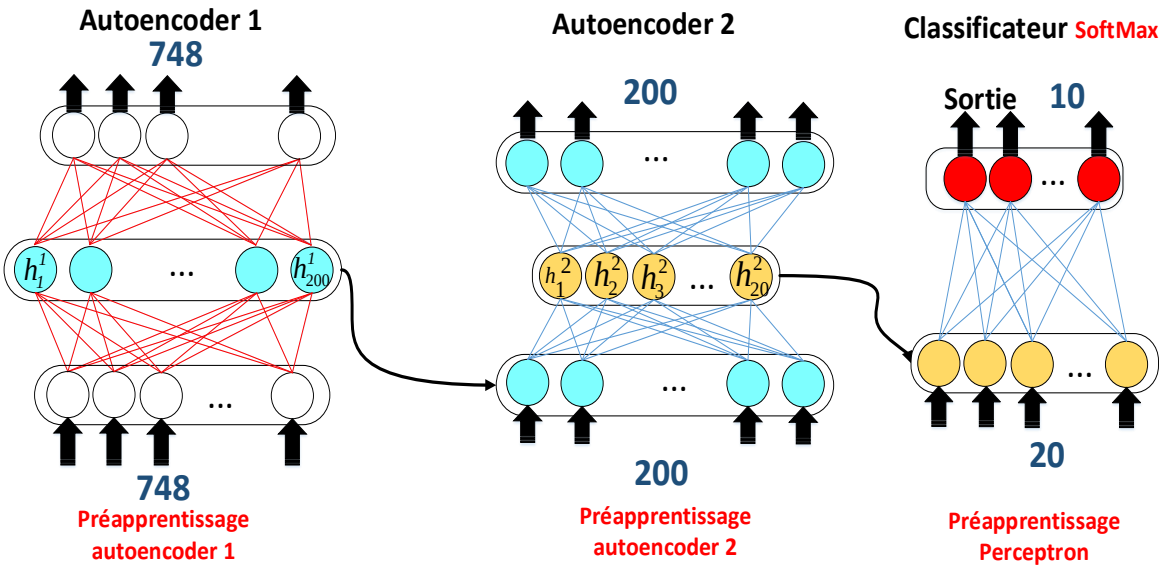


Figure 3.32 – La structure de classification basée sur l'autoencodeur empilé.

Nous remarquons que le premier autoencodeur contient 748 neurones dans les couches d'entrées et de sorties, et 200 neurones pour la couche cachée. Ensuite le deuxième autoencodeur contient 20 neurones dans la couche cachée. Le choix sera fait par une boucle de calcul afin de choisir les nombres de neurones pour chaque couche. La boucle est incrémentée par 10 neurones pour chaque étape et dans chaque cas, nous avons calculé et comparé taux de classification obtenu. Au premier temps, nous avons effectué un test dont l'objectif est de déterminer le nombre de neurones qu'il faut utiliser dans le premier et le deuxième autoencodeur.

D'après le tableau ci-dessous, le meilleur taux de classification est de 99.5%.

Tableau 3.3 – les taux de classification pour la base de test en fonction de nombre de neurone.

Nombre de neurone de l'autoencodeur 2 \ *Nombre de neurone de l'autoencodeur 1*	**10**	**20**	**30**	**40**	**50**	**60**	**70**	**80**	**90**	**100**	**110**	**120**	**130**	**140**	**150**	**160**	**170**	**180**	**190**	**200**
10	69,6	68,58	74,44	39,9	37,12	38	72,06	46,68	71,84	55,82	45,12	44,62	62,9	47,16	44,2	46,28	45,92	52,34	48,36	56,08
20	88,3	54,62	75,62	51,34	94,5	93,02	93,38	94,88	94,44	92,3	93,06	94,28	93,74	92,86	93,62	93,88	93,94	93,52	93,8	93
30	65,4	95,72	96,16	95,78	95,74	96,26	96,6	95,64	95,78	96,68	96,28	96,9	96,6	96,02	96,88	96,58	96,66	96,16	96,6	96,08
40	73,64	97,02	97,06	96,1	97,32	97,12	97,34	96,52	96,28	96,68	97,1	97,4	97,26	96,76	97,44	97,46	97,2	97,72	97,1	96,96
50	95,52	97,18	98,12	96,86	97,52	96,98	97,98	97,58	97,9	97,4	97,22	97,82	97,74	98,08	98,16	97,82	97,44	97,58	97,88	98,04
60	94,3	97,2	98,1	98,42	98	97,92	98,28	98,22	98,54	97,94	97,98	98,22	98,34	98,14	97,9	98,1	98,2	98,28	97,9	98,04
70	95,4	98,14	97,72	97,8	98,42	98,52	98,18	98,56	98,26	98,42	98,46	98,36	98,5	98,28	98,58	98,36	98,06	98,38	98,3	98,44
80	97,44	98,56	98,64	98,86	98,38	99,04	98,74	98,92	98,44	98,42	98,9	98,84	98,58	98,72	98,68	98,8	98,76	98,64	98,04	97,96
90	97,54	98,12	98,28	98,88	98,72	98,94	98,76	98,92	98,86	98,86	98,74	98,68	98,9	98,8	98,82	98,64	98,58	98,64	98,94	98,76
100	96,62	98,54	98,68	98,92	98,94	98,96	98,88	98,84	99,1	98,76	98,82	98,92	98,96	98,96	98,92	98,66	98,78	98,76	98,9	98,6
110	98,62	98,3	98,54	98,88	99,16	98,52	98,88	98,94	98,98	99,08	99,18	99,06	98,98	98,88	99,04	98,82	98,92	98,46	98,6	98,74
120	98,32	98,88	99,06	98,86	98,88	99,02	98,96	99,04	99,14	99,16	99,12	99,06	98,5	98,9	98,76	99,12	98,76	98,94	98,84	98,64
130	97,86	98,26	98,92	98,72	99,04	99,14	99,12	99,2	99,04	99,02	98,96	98,98	98,96	98,88	98,78	98,74	98,76	98,7	98,66	98,72
140	98,22	98,72	98,72	98,84	98,82	99,2	99,08	99,08	99,38	98,86	98,94	99,16	99,02	98,8	98,8	98,82	98,84	98,78	98,28	98,94
150	99,12	98,92	99,12	98,94	99,22	99,24	98,98	99,08	99,12	99,1	98,84	99,12	99,02	99,16	99,16	98,88	98,74	98,82	98,64	99,32
160	99,26	99,1	98,98	99,1	99,38	99,26	99,06	99,38	99,34	99,26	99,02	99,22	99,38	99,16	99,24	99,3	98,9	98,84	98,84	99
170	99,34	99,22	99,04	99,28	99,38	99,24	99,32	99,08	99,28	99,34	99,34	99,32	98,92	99,16	98,86	99	99,08	98,78	99,18	99,02
180	99,02	98,82	99,28	99,42	99,36	99,4	99,42	99,4	99,32	99,08	99,08	98,9	99	99,1	99,22	98,76	98,82	99,38	99,16	98,98
190	98,96	99,26	99,16	99,24	99,34	99,44	99,4	99,18	99,28	99,26	99,32	98,98	98,94	98,78	98,96	98,82	98,8	98,98	98,84	98,72
200	99,1	**99,5**	98,8	99,4	99,14	99,42	99,42	99,36	99,08	99,02	99,04	99	98,76	99,32	99	98,98	99,08	99,28	99,22	99,02

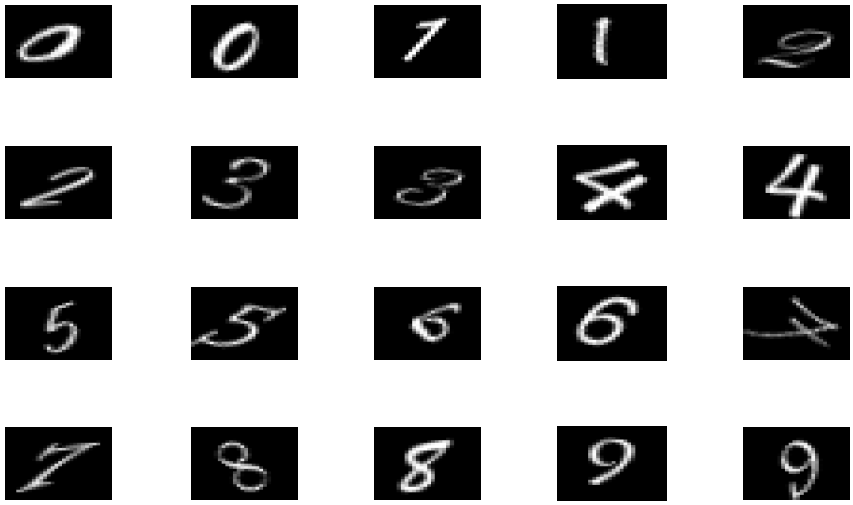
Comme nous l'avons montré, l'algorithme de l'apprentissage non-supervisé utilisé pour l'entrainement des autoencodeurs dans le chapitre précédant l'entrée de l'autoencodeur sera envoyé à la sortie. Alors, la figure 3.33 présente la base des chiffres utilisée pour l'apprentissage et leurs sorties estimées. D'après le résultat obtenu, nous remarquons que le réseau est bien entrainé.

(a) Exemple de la base des données utilisée pour l'apprentissage non supervisé (chiffre d'entre

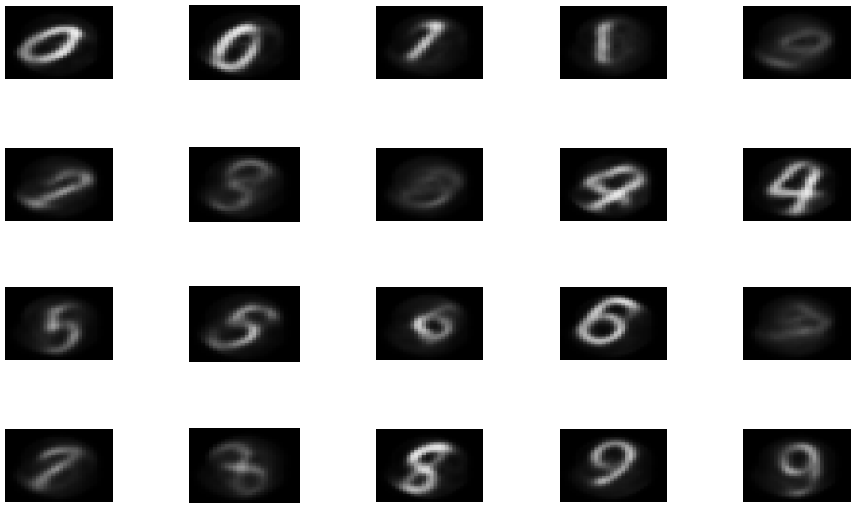
(b) Les résultats des chiffres estimés dans la sortie de premier autoencodeur 1

Figure 3.33 – Base des données d'apprentissage.

Pour valider la partie de l'apprentissage, nous avons utilisé la base de test pour voir la capacité de reconstruction des images des chiffres. La figure 3.34 présente un exemple des tests en prenant des chiffres de 0 à 9, nous remarquons que l'autoencodeur 1 bien identifie les chiffres de la base de test.



(a) Exemple de la base des données utilisée pour l'apprentissage non supervisé (chiffre d'entre)



(b) Les résultats des chiffres estimés dans la sortie de premier autoencodeur 1

Figure 3.34 – Base des données de test.

Le deuxième autoencodeur 2 est utilisé pour créer de nouvelles caractéristiques qui peuvent améliorer la performance du réseau profond présenté dans la figure 3.35.

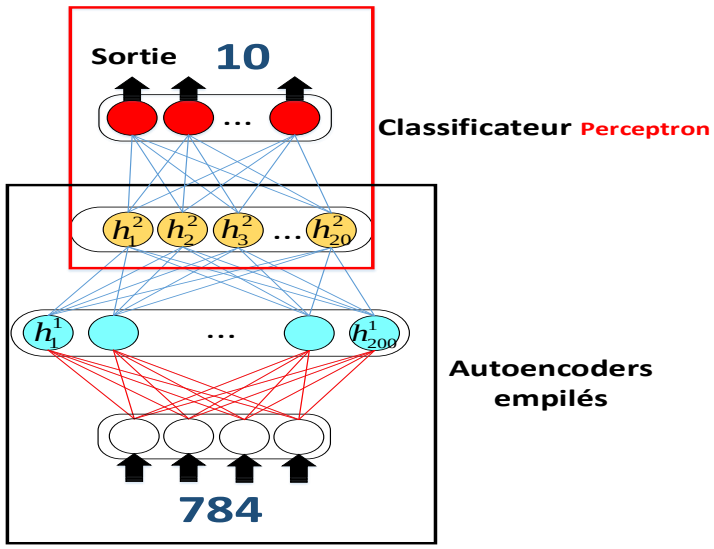
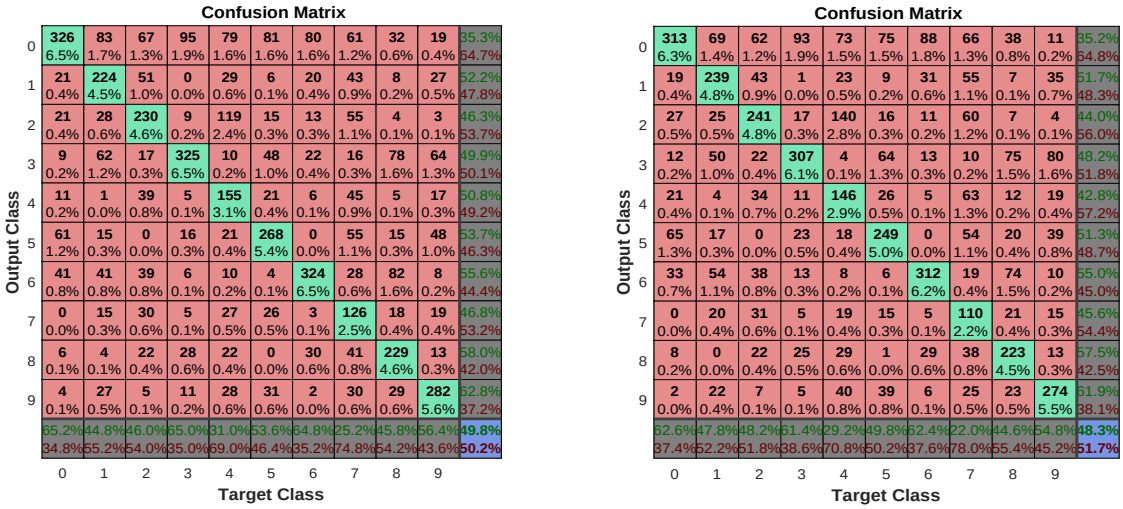


Figure 3.35 – Réseau de neurone profond proposé pour les chiffres synthétiques.

Le premier test de réseau profond avant l'apprentissage affiné de l'ensemble des réseaux est présenté ci-dessous pour la base d'apprentissage et de test.

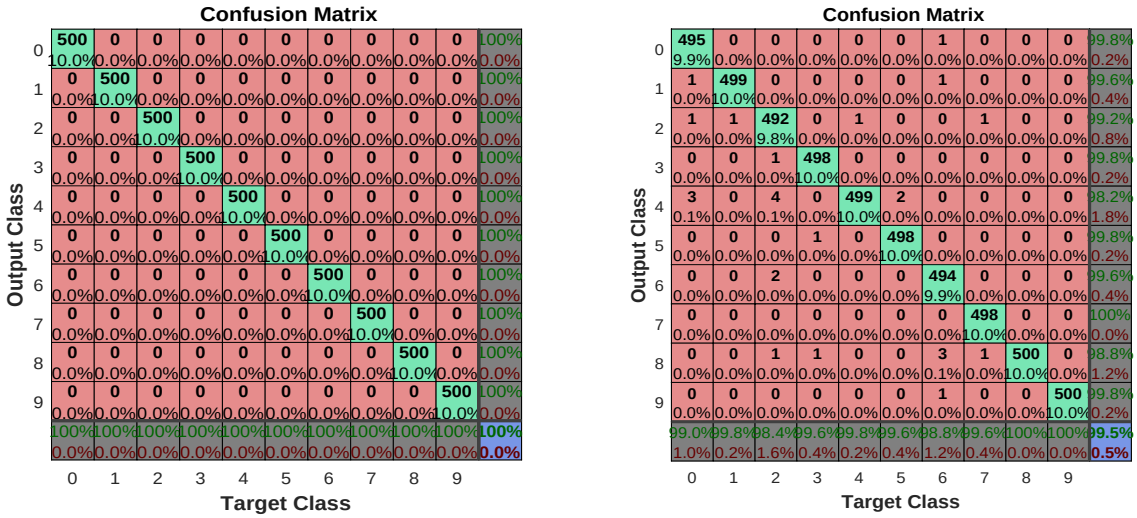


(a) Matrice de confusion pour la base d'apprentissage

(b) Matrice de confusion pour la base de test

Figure 3.36 – Les matrices de confusion avant l'apprentissage affiné.

Nous clairement remarquons qu' il ya des fausses classifications ; ce qui implique le passage à la partie de l'apprentissage affiné pour améliorer la classification. Les résultats pour le réseau neurone profond peuvent être améliorés en effectuant une rétropropagation sur l'ensemble du réseau multicouche. Ce processus est souvent appelé réglage fin ou l'apprentissage affiné. Nous allons affiner le réseau en le recyclant sur les données d'entraînement de manière supervisée.



(a) Matrice de confusion pour la base d'apprentissage

(b) Matrice de confusion pour la base de test

Figure 3.37 – Les matrices de confusion après l'apprentissage affiné.

La figure 3.37 présente la matrice de confusion pour la base d'apprentissage. Nous avons obtenu un taux de classification parfait (100%) ; ce qui montre l'efficacité de l'apprentissage affiné. D'autre part, la matrice de confusion de la base de test présente 99.5% de taux de classification. Dans le but de bien vérifier ces résultats, nous avons tracé les dix sorties qui représentent la couche de sortie.

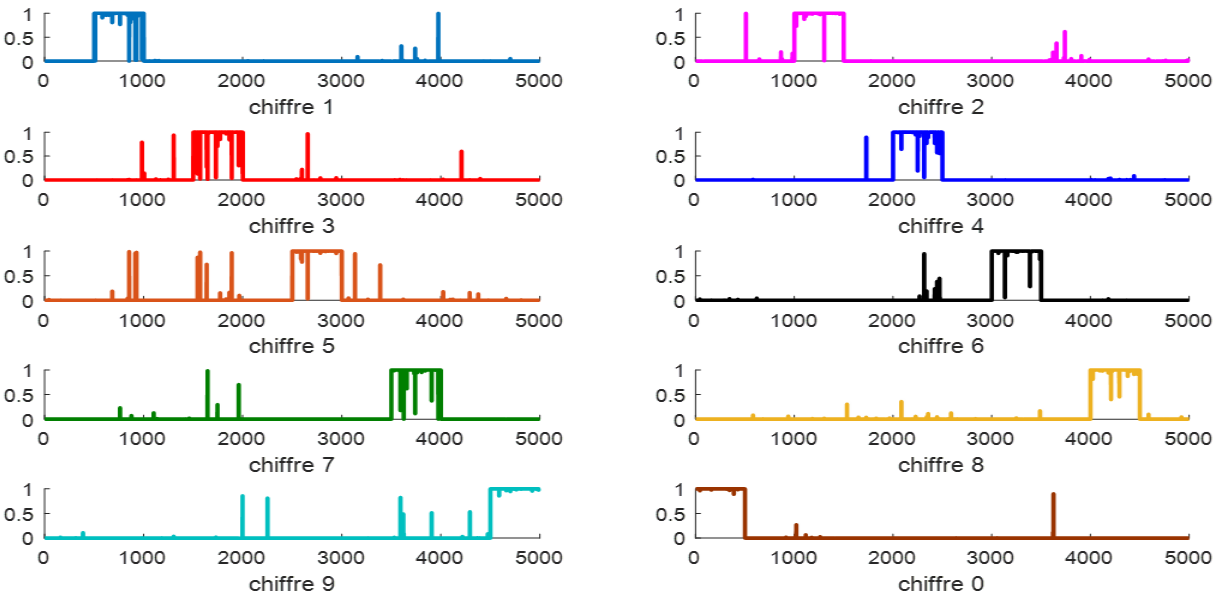


Figure 3.38 – La représentation de la sortie de la couche softmax après l'apprentissage affiné.

D'après la figure 3.38,l'efficacité de la technique d'apprentissage profond. Par ailleurs,

ce test a montré comment former un réseau de neurones profonds pour classer les chiffres à l'aide d'un autoencodeur empilé. Les étapes qui ont été décrites peuvent être appliquées à d'autres problèmes similaires, tels que la classification des images de lettres, ou même de petites images d'objets d'une catégorie spécifique.

Ce taux de classification peut être amélioré par l'application d'autres types de réseau de neurone comme le réseau de neurone convolutif en deux dimensions. Alors, la partie suivante traite cette technique.

3.6.5 Réseau de Neurone Convolutif

Le réseau convolutif (Convolutional Neural Networks CNN) a réalisé les meilleures performances dans la reconnaissance des objets pour des bases de données d'images multiples : MNIST, NORB, HWDB1.0, CIFAR10 et l'ensemble de données ImageNet.

Dans les réseaux de neurones convolutifs, chaque couche agit comme un filtre de détection pour la présence des caractéristiques spécifiques ou des motifs présents dans les données d'origine. Les premières couches d'un convolutif détectent des caractéristiques qui peuvent être reconnues et interprétées relativement et facilement. Les couches ultérieures détectent de plus en plus des caractéristiques plus abstraites. La dernière couche du réseau convolutif est capable de faire une classification ultra-spécifique en combinant toutes les caractéristiques spécifiques détectées par les couches précédentes dans les données d'entrée.

Notre architecture de classification est combinée de convolution et Max pooling. Cependant, pour obtenir une classification rapide permettant une classification et localisation en temps réel, la Figure 3.39 montre les sept couches de notre réseau convolutif. Dans ce travail, nous allons proposer la structure de réseau neuronal convolutionnel, comme le montre la figure ci-dessous.

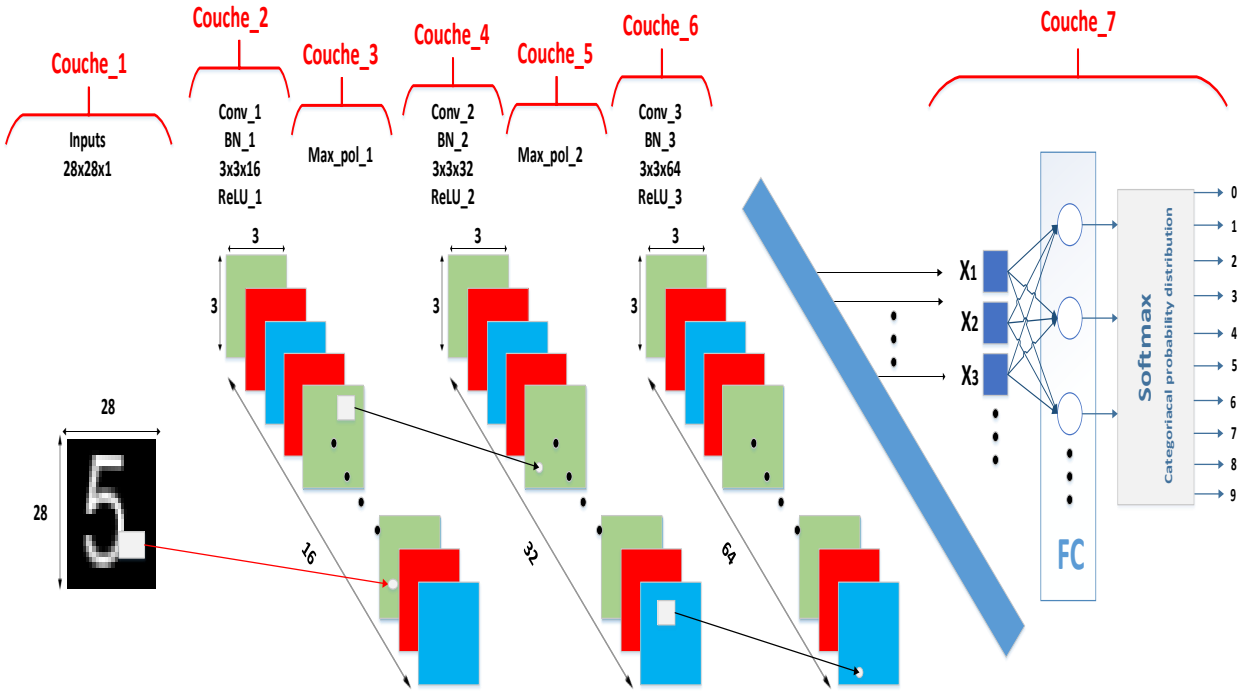


Figure 3.39 – La structure proposée de réseau de neurone convolutif.

Notre structure se compose de plusieurs couches. Dans la partie suivante, nous allons expliquer les différentes couches de réseau de neurone convolutionnel :

1. Couche des images d'entrée : Cette couche est spécifiée par sa taille. Dans notre cas, elle est 28,28 par 1 ([28,28,1]), les nombres correspondent respectivement à la hauteur, la largeur, et la taille de canal. Les données des images des chiffres sont de type de niveau de gris, alors, la taille de canal est égale à 1. Pour une image qui contient des couleurs, la taille de canal est 3 qui correspond aux couleurs RGB.

2. Couches convolutives : Les couches convolutives constituent le noyau du réseau convolutif. Ces couches se composent d'une grille rectangulaire de neurones qui ont un petit champ réceptif étendu à travers toute la profondeur du volume d'entrée. Ainsi, la couche convolutionnelle est juste une convolution d'images de la couche précédente, où les poids spécifient le filtre de convolution. Dans notre réseau, nous avons utilisé trois couches convolutives de (3,16), (3,32) et (3,64) comme le montre le tableau ci-dessous (voir le tableau 3.4).

   Normalisation : Après la convolution, il peut y avoir une normalisation dont l'objectif

est de faciliter l'apprentissage en réseau avec la fonction d'activation de type ReLU.

3. Couches de pooling : Après chaque couche convolutive, il peut y avoir une couche de pooling qui sous échantillonne leur entrée. Il y a plusieurs façons de faire cette mise en commun, comme prendre la moyenne ou le maximum, ou une combinaison linéaire apprise des neurones dans le bloc. Dans notre cas et comme le montre la figure 3.39 max pooling sur une fenêtre 2×2.

4. Couches totalement connectées : De plus, après plusieurs couches de convolution et pooling nous avons utilisés sept couches. Le raisonnement de haut niveau dans le réseau neuronal se fait via des couches totalement connectées. La couche six et sept sont entièrement connectées. La sortie de la dernière couche totalement connectée alimente un Softmax 10 voies produisant une distribution sur 10 classes.

5. Couche de SoftMax : Avant la dernière couche, la fonction d'activation softmax normalise la sortie de la couche entièrement connectée. La sortie de couche softmax est constituée des nombres positifs qui totalisent les uns, qui peuvent ensuite être utilisés comme probabilités par la couche de classification.

6. Couche de classification : La couche finale est la couche de classification qui utilise les probabilités renvoyées par la fonction d'activation softmax.

Tableau 3.4 – La structure de CNN.

Image Input Layer	[28 28 1]
Convolution 2d Layer batch Normalization Layer relu Layer	(3,16)
Max Pooling 2d Layer	'Stride' (2,2)
Convolution 2d Layer batch Normalization Layer relu Layer	(3,32)
Max Pooling 2d Layer	'Stride' (2,2)
Convolution 2d Layer batch Normalization Layer relu Layer	(3,64)
Fully Connected Layer Softmax Layer Classification Layer	10

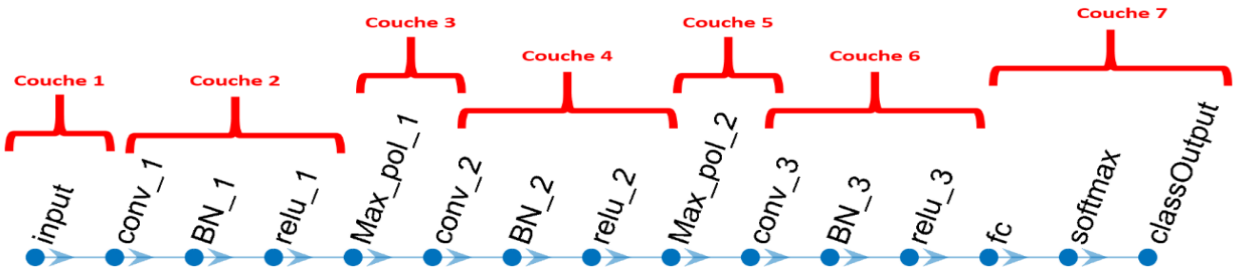


Figure 3.40 – Les étapes de classification de CNN.

La base de données se compose de 10000 images qui contient des chiffres de 0 à 9 et chaque classe a 1000 échantillons. Nous avons subdivisé la base des données sur deux parties pour l'apprentissage et le test. La précision de classification sur l'ensemble d'apprentissage est de 100% et sur l'ensemble de test est de 99.8%. Le temps d'apprentissage est écoulé d'environ de 11min et 11s sous 290 itérations. La figure ci-dessous présente la précision et la perte d'apprentissage en fonction de nombre d'itérations imposées.

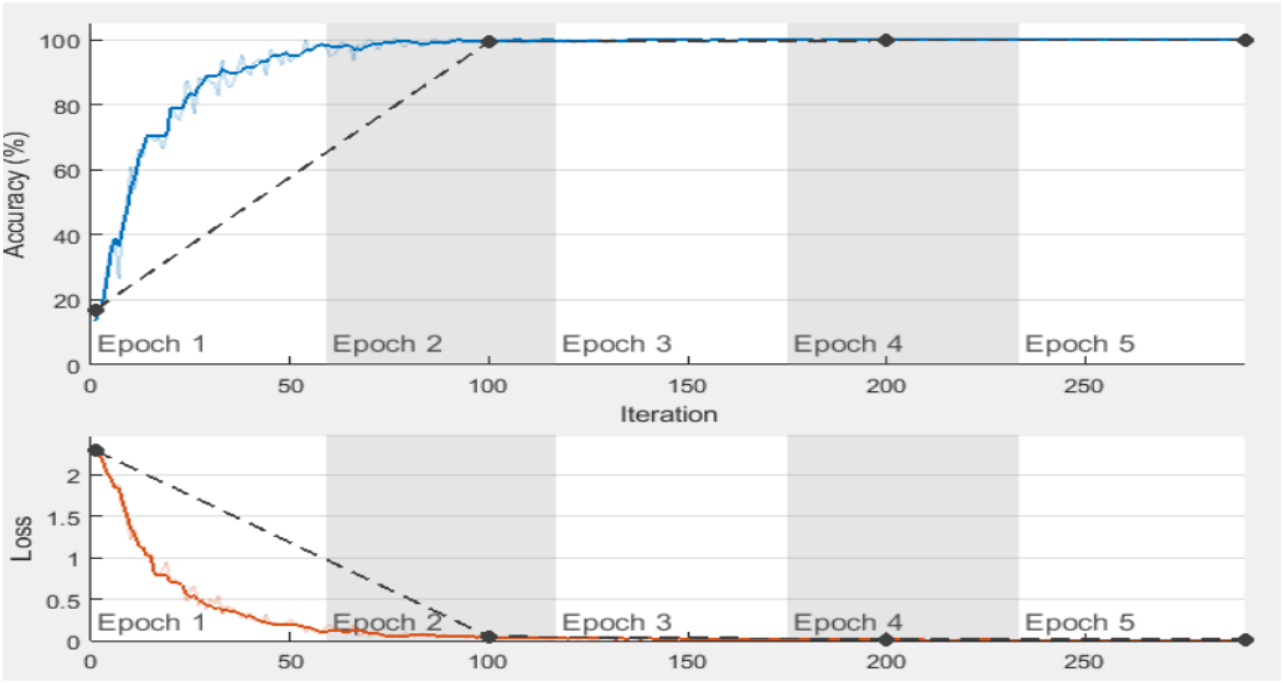


Figure 3.41 – La précision et la perte de la classification par CNN.

Afin de comparer les résultats de l'apprentissage et de test avec leurs sorties désirées, la correspondance entre la sortie désirée et la sortie de réseau CNN pour la base d'apprentissage est illustrée dans la figure 3.42 et pour la base de test dans la figure 3.43.

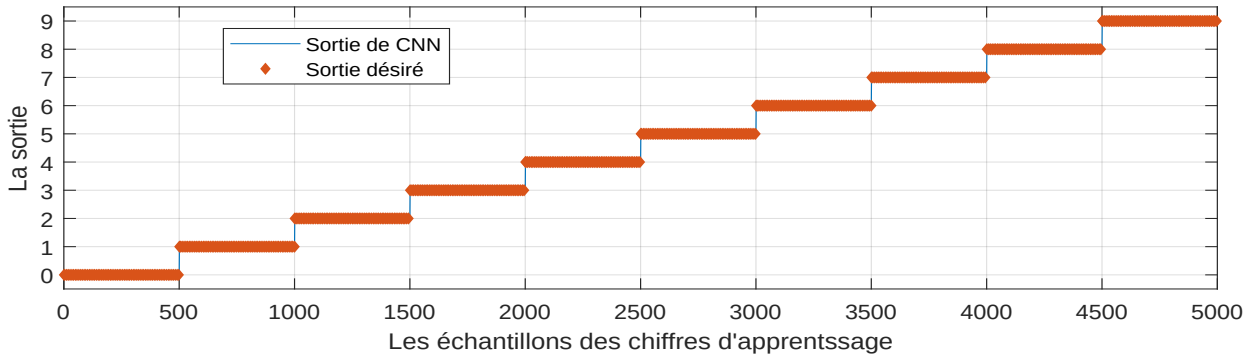


Figure 3.42 – La précision et la perte de la classification par CNN.

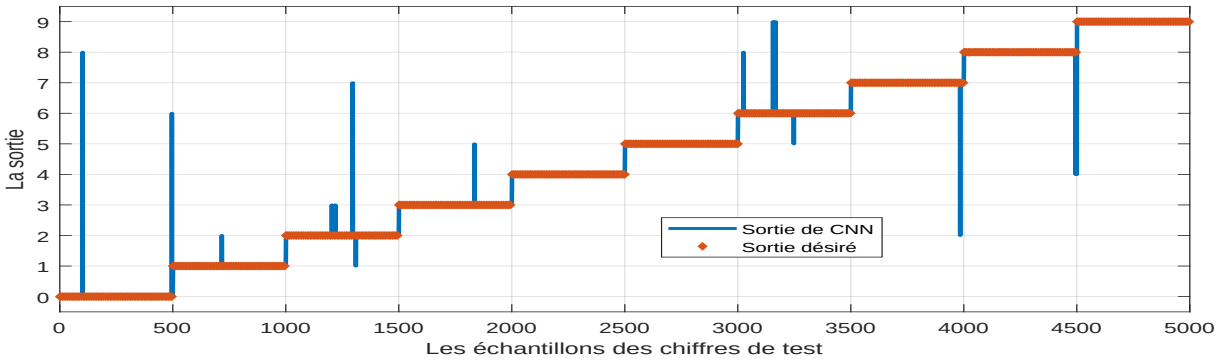


Figure 3.43 – La précision et la perte de la classification par CNN.

Nous remarquons que la technique d'apprentissage approfondi basé sur le réseau de neurone convolutif donne des meilleures précisions par rapport aux autoencodeurs empilés et les autres techniques classiques.

3.7 Étude comparative

Le tableau ci-dessous récapitule une étude comparative entre les techniques utilisées dans ce mémoire.

Nous commençons premièrement par la base des eaux industriels. Nous pouvons facilement constaté que le temps d'exécution est presque pareil pour toutes les quatre techniques (KNN, SVM, RNA (MLP) et L'apprentissage approfondi des réseaux de neurone (Deep Learning) à cause de la petite taille de la base de données. En effet, l'augmentation de nombre des neurones diminue le nombre d'itération, d'autre part, plus le nombre de neurone augmente plus le réseau de neurone converge rapidement. De surcroît, le réseau de neurone profond (autoencodeurs + perceptron) présente un temps d'apprentissage très petite et un nombre

d'itération inferieur par rapport les autres techniques. Partant de ce fait, cette technique présente une faible précision pour le test étant donné un taux de classification très important de 100%.

Tableau 3.5 – Tableau récapitulatif pour la base donnée des eaux.

Les techniques	Les paramètres	Le temps d'apprentissage (S)	Nombre d'itération	Taux de classification (%)		Erreur quadratique moyenne	
				Apprentissage	Test	Apprentissage	Test
KNN	Fine	0.50	/	100	88	/	/
	Medium	0.08	/	100	92	/	/
	Coarse	0.07	/	100	68	/	/
	Cosine	0.11	/	100	92	/	/
	Cubic	0.07	/	100	88	/	/
	Weighted	0.07	/	100	92	/	/
SVM	Linear	4,6	/	96	100	/	/
	quadratic	0,4	/	100	92	/	/
	Cubic	0,3	/	100	96	/	/
	Fine Gaussian	0,3	/	96	96	/	/
RNA (MLP)	5 n	3,2	700	100	100	1.15×10^{-10}	1.96×10^{-4}
	10 n	3,8	600	100	100	2.42×10^{-16}	0.01
	15 n	2,8	500	100	100	9.69×10^{-14}	0.0037
	20 n	1,5	200	100	100	0	0.0435
	25 n	1,6	200	100	100	0	0.0585
Deep learning	**Autoencodeur et perceptron**	**0.01**	**3**	**100**	**100**	2.16×10^{-21}	**0.0174**

Tableau 3.6 – Tableau récapitulatif pour la base des données des chiffres.

Les techniques	Les paramètres	Le temps d'apprentissage (S)	Nombre d'itération	Taux de classification (%)		Erreur quadratique moyenne	
				Apprentissage	Test	Apprentissage	Test
KNN	Fine	46.57	/	100	95.2	/	/
	Medium	45.13	/	100	84.9	/	/
	Coarse	46.27	/	100	55.4	/	/
	Cosine	44.41	/	100	87.3	/	/
	Cubic	798.23	/	100	78.1	/	/
	Weighted	40.25	/	100	92.1	/	/
SVM	Linear	89.17	/	100	80.2	/	/
	quadratic	97.59	/	100	94.7	/	/
	Cubic	104.16	/	100	95.5	/	/
	Fine Gaussian	356.06	/	100	73.2	/	/
	Medium Gaussian	159.7	/	100	88.4	/	/
	Coarse Gaussian	246.15	/	100	63.9	/	/
Deep learning	**Autoencodeur et softmax**	**83,94**	**120**	**100**	**99,5**	2.73×10^{-11}	8.87×10^{-04}
	CNN	**671**	**290**	**100**	**99,8**	2.92×10^{-23}	8.87×10^{-11}

La table ci-dessus présente les techniques utilisées pour la base des chiffres synthétiques, KNN et SVM avec six différents noyaux ainsi que le RNA (MPL) avec 20 et 30 neurones. D'ailleurs, Autoencodeurs empiles avec 10 neurones de type Softmax dans l'objectif de la classification des dix chiffres de 0 à 9, aussi, un réseau de neurone convolutif. La meilleur résultats présenter par la technique de SVM avec un noyau avec taux de classification de

95% de plus un temps important. D'après les résultats obtenus nous remarquons que le taux de classification est augmente à 99.5% en utilisant l'autoencodeur avec un temps moyen important. Le CNN est présente un taux de classification de 99.8% ce qui montre l'efficacité de la technique d'apprentissage approfondi, mais il présente un temps d'apprentissage très élevée.

3.8 Conclusion

Dans ce chapitre, nous avons présenté une évaluation expérimentale des quatre différentes machines d'apprentissage appliquées sur deux bases des données de différentes natures : une base destinée à la classification des eaux industrielles et l'autre pour la classification des chiffres. Nous avons testé des machines d'apprentissage conventionnelles (KNN, SVM et RNA de type MLP) afin de dégager les avantages et les inconvénients de chacune. Également, un grand intérêt a été focalisé sur l'apprentissage profond basé sur les méthodes des auto-encodeurs empilés et le réseau de neurone convolutif. Les résultats obtenus sont analysés et comparés pour sortir avec des conclusions importantes sur l'efficacité et les lacunes de chaque machine d'apprentissage. Nous avons remarqué qu'avec une base des données de petite taille, Il n'y a pas vraiment une différence importante entre ces techniques et c'est le cas des eaux industrielles. Par contre, lorsque nous utilisons une grande collection des données en deux dimensions, le deep learning présente une puissance considérable dans la décision.

Conclusion Générale

Le travail présenté dans ce mémoire a été consacré à la mise en œuvre des techniques d'apprentissage statistique appliquées à la reconnaissance des formes dans deux applications différentes. La première était le contrôle et la surveillance des eaux déminéralisées utilisées dans l'industrie. Cette étude s'inscrit dans le cadre des progrès importants qui ont été enregistrés ces dernières années dans le but et l'intérêt d'une surveillance moderne et plus efficace de la qualité des eaux déminéralisées. La deuxième s'est articulée autour de la reconnaissance automatique des chiffres.

Les divers dispositifs et outils de surveillance dans ce domaine existants actuellement, sont réalisés dans le but d'assurer une surveillance permanente et efficace des installations de production de l'eau déminéralisée. C'est dans l'esprit et l'intérêt que présente la surveillance de la qualité de l'eau dans les usines de production et de distribution, que nous avons étudié dans ce mémoire. Notre objectif était d'aboutir à une technique de contrôle bien adaptée à la surveillance de cette ressource précieuse de manière efficace et permanente.

Cette étude a été structurée autour de trois chapitres essentiels. Le premier consacré à une introduction au domaine de reconnaissance des formes, le principe général des méthodes de reconnaissance des formes et Les différentes phases pour construire un système par RdF.

Le deuxième chapitre a été particulièrement dédié aux mécanismes théoriques des méthodes de classification des données à apprentissage statistique supervisé. Dans ce chapitre, Nous avons mis en œuvre les techniques classiques (KNN, SVM et ARN) et les machines d'apprentissage puissantes (CNN et Auto-encodeur) fondées sur ce type d'apprentissage.

Le dernier chapitre a fait l'objet d'une étude en simulation concernant la mise en œuvre de ces modèles d'apprentissage statistique appliqués dans le domaine du contrôle et de surveillance des eaux déminéralisées et la reconnaissance automatique des chiffres synthétiques. Cette

étude a permis la validation et l'évaluation des performances de chacune des méthodes présentées. Une étude comparative permettant un choix décisif de la méthode la plus adaptée à l'application proposée a été effectuée. La discussion des résultats obtenus des paramètres liés au temps d'apprentissage, taux de classification et l'erreur, a permis d'opter pour les techniques basées sur l'apprentissage approfondi pour leurs efficacités au problème posé. Cette technique a fourni de très bons résultats qui permettent de trouver une solution satisfaisante pour notre problème de classification.

Deux bases des données pour évaluer les performances de chaque machine :

- Base des données réelle provenant de la station de déminéralisation de la wilaya de M'sila appliquée à la technique en question, a montré l'efficacité de l'approche.
- Base des données constituée des imagesde la bibliothèque Matlab des chiffres synthétiques.

Comme perspectives, nous suggérons aux futures promotions d'implanter des systèmes de reconnaissance des formes basés sur les machines d'apprentissage approfondi sur des cartes DSP afin de les mettre directement dans l'industrie. De plus, nous proposons des nouvelles techniques pour l'amélioration des performances comme la transformation de l'apprentissage approfondi (Deep Transfert Learning).

Références bibliographiques

[Aboulaiche and Meskine, 2010] Aboulaiche, N. and Meskine, N. (2010). *Implémentation d'un systéme de reconnaissance de mots arabes manuscrits basé sur la transformée en Ridgelets et les SVMs.* PhD thesis, Université des Sciences et de la Technologie Houari Boumediene.

[Baccouche, 2013] Baccouche, M. (2013). *Apprentissage neuronal de caractéristiques spatio-temporelles pour la classification automatique de séquences vidéo.* PhD thesis, INSA de Lyon.

[Bishop et al., 1995a] Bishop, C., Bishop, C. M., et al. (1995a). *Neural networks for pattern recognition.* Oxford university press.

[Bishop et al., 1995b] Bishop, C., Bishop, C. M., et al. (1995b). *Neural networks for pattern recognition.* Oxford university press.

[Blum and Langley, 1997] Blum, A. L. and Langley, P. (1997). Selection of relevant features and examples in machine learning. *Artificial intelligence*, 97(1-2), pp 245–271.

[Boser et al., 1992] Boser, B. E., Guyon, I. M., and Vapnik, V. N. (1992). A training algorithm for optimal margin classifiers. In *Proceedings of the fifth annual workshop on Computational learning theory*, pages 144–152. ACM.

[Boudaoud, 1997] Boudaoud, N. (1997). *Conception d'un Système de Diagnostic Adaptatif en Ligne pour la Surveillance des Systèmes Evolutifs.* PhD thesis.

[Boutleux, 1996] Boutleux, E. (1996). *Diagnostic et suivi d'évolution de l'état d'un système par reconnaissance des formes floues : application au modèle du réseau téléphonique français.* PhD thesis, Compiègne.

[Bouziane et al., 2008] Bouziane, H., Messabih, B., and Chouarfia, A. (2008). Prédiction de la structure 2d des protéines par les réseaux de neurones. *Communications of the IBIMA*, 6, pp 201–207.

[Burman, 1989] Burman, P. (1989). A comparative study of ordinary cross-validation, v-fold cross-validation and the repeated learning-testing methods. *Biometrika*, 76(3), pp 503–514.

[Cortes and Vapnik, 1995] Cortes, C. and Vapnik, V. (1995). Support-vector networks. *Machine learning*, 20(3), pp 273–297.

[Dash and Liu, 1997] Dash, M. and Liu, H. (1997). Feature selection for classification. *Intelligent data analysis*, 1(3), pp 131–156.

[Diday, 1971] Diday, E. (1971). Une nouvelle méthode en classification automatique et reconnaissance des formes : la méthode des nuées dynamiques. *Revue de statistique appliquée*, 19(2), pp 19–33.

[Dubuisson, 2001] Dubuisson, B. (2001). *Diagnostic, intelligence artificielle et reconnaissance des formes.* Hermès science publications.

[Duda et al., 2012] Duda, R. O., Hart, P. E., and Stork, D. G. (2012). *Pattern classification.* John Wiley & Sons.

[Duda et al., 2001] Duda, R. O., Hart, P. E., Stork, D. G., et al. (2001). Pattern classification. 2nd. *Edition. New York*, page 55.

[Fukunaga, 2013] Fukunaga, K. (2013). *Introduction to statistical pattern recognition*. Academic press.

[Fukushima, 1979] Fukushima, K. (1979). Neural network model for a mechanism of pattern recognition unaffected by shift in position-neocognitron. *IEICE Technical Report, A*, 62(10), pp 658–665.

[Gosselin, 1996] Gosselin, B. (1996). Application de réseaux de neurones artificiels à la reconnaissance automatique de caractères manuscrits. *Faculté Polytechnique de Mons*.

[Hartert, 2010] Hartert, L. (2010). *Reconnaissance des formes dans un environnement dynamique appliquée au diagnostic et au suivi des systèmes évolutifs*. PhD thesis, Université de Reims-Champagne Ardenne.

[Haykin, 1994] Haykin, S. (1994). Neural networks : A comprehensive foundation : Macmillan college publishing company. *New York*.

[Hechenbichler and Schliep, 2004] Hechenbichler, K. and Schliep, K. (2004). Weighted k-nearest-neighbor techniques and ordinal classification.

[Hubel and Wiesel, 1962] Hubel, D. H. and Wiesel, T. N. (1962). Receptive fields, binocular interaction and functional architecture in the cat's visual cortex. *The Journal of physiology*, 160(1), pp 106–154.

[Jain and Dubes, 1988] Jain, A. K. and Dubes, R. C. (1988). Algorithms for clustering data.

[Jain et al., 2000] Jain, A. K., Duin, R. P. W., and Mao, J. (2000). Statistical pattern recognition : A review. *IEEE Transactions on pattern analysis and machine intelligence*, 22(1), pp 4–37.

[Jaumard-Hakoun, 2016] Jaumard-Hakoun, A. (2016). *Modélisation et synthèse de voix chantée à partir de descripteurs visuels extraits d'images échographiques et optiques des articulateurs*. PhD thesis, Université Pierre et Marie Curie-Paris VI.

[Ladjal, 2013] Ladjal, M. (2013). *Contribution au développement de systèmes de surveillance innovants dédiés au contrôle de la qualité des eaux potables*. PhD thesis, Laboratoire LASS, Univ de M'sila.

[Lebrun, 2006] Lebrun, G. (2006). *Sélection de modèles pour la classification supervisée avec des SVM (Séparateurs à Vaste Marge). Application en traitement et analyse d'images.* PhD thesis, Université de Caen Basse-Normandie.

[LeCun et al., 1990] LeCun, Y., Boser, B. E., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W. E., and Jackel, L. D. (1990). Handwritten digit recognition with a back-propagation network. In *Advances in neural information processing systems*, pages 396–404.

[Mamar, 2008] Mamar, Z. H. (2008). *Analyse temps-échelle et reconnaissance des formes pour le diagnostic du système de guidage d'un tramway sur pneumatiques*. PhD thesis, Thèse de doctorat, Université Blaise Pascal, Clermont-Ferrand II.

[Merhy et al., 2017] Merhy, M. a. et al. (2017). *Reconnaissance de formes basée géodésiques et déformations locales de formes*. PhD thesis, Brest.

[Mohamed et al., 2015] Mohamed, B., Hafaifa, A., Mouloud, G., et al. (2015). Vibration modeling improves pipeline performance, costs. *Oil & Gas Journal*, 113(3), pp 98–100.

[Perny, 1998] Perny, P. (1998). Multicriteria filtering methods based onconcordance and non-discordance principles. *Annals of operations Research*, 80, pp 137–165.

[Saadat, 2016] Saadat, B. (2016). Vibration analysis and measurement based on defect signal evaluation : Gas turbine investigation. *Journal of Advanced Research in Science and Technology*, 3(1), pp 271–280.

[Sarzeaud, 1995] Sarzeaud, O. (1995). *Les réseaux de neurones : contribution à une théorie*. Ouest éd.

[Tian et al., 2010] Tian, X., Herault, R., Gasso, G., Canu, S., and EA4108, L. (2010). Pré-apprentissage supervisé pour les réseaux profonds. In *Proceedings of Rfia*, volume 2010, page 36.

[Vitola et al., 2017] Vitola, J., Pozo, F., Tibaduiza, D. A., and Anaya, M. (2017). A sensor data fusion system based on k-nearest neighbor pattern classification for structural health monitoring applications. *Sensors*, 17(2), pp 417.

Résumé

La reconnaissance des formes a été largement étudiée au cours des dernières années. Dans ce travail, nous avons évalué plusieurs techniques des machines d'apprentissage en utilisant deux bases de données : base des chiffres et base des eaux industrielles. Nous avons introduit la technique d'apprentissage approfondi dont l'objectif est de classifier les eaux déminéraliser en utilisant l'autoencodeur. Nous avons aussi traité dans ce mémoire le problème de la reconnaissance des chiffres. Les solutions proposées permettent de s'adapter aux changements des styles d'écriture, aux différents angles d'orientation et différents fonts d'écriture ; pour augmenter les performances nous avons utilisé le réseau de neurone de type Auto-encodeur et CNN. Finalement, nous avons présenté une étude comparative pour savoir les points faibles et les points forts de chaque technique, nous avons eu les meilleurs résultats en appliquant la technique d'apprentissage approfondi (Auto-encodeur et CNN).

Mots clés : reconnaissance de forme, machine d'apprentissage, apprentissage approfondi, SVM, K-PPV, RNA (MLP), Autoencodeur, CNN.

Abstract

Recently, pattern recognition has been widely used in several applications. This work evaluates four techniques of machine learning including conventional and deep approaches applied on two different databases; digit and industrial water databases. First, the deep learning has been involved to classify demineralized water based on the stacked autoencoder. Whereas, the stacked autoencoder and the convolution neural network have been used to classify the ten classes of digit data. These techniques respond well to the changing in writing styles, different angles of orientation as well as writing fonts. Unlike traditional machine learning, the deep approaches provide a best results by increasing the classification accuracy. Furthermore, a comparative study have been presented to show the drawbacks and benefits for each techniques.

Keywords: pattern recognition, machine learning, deep learning, SVM, K-NN, ANN (MLP), Autoencoder, CNN.

ملخص

دراسة التعرف على الأنماط (Reconnaissance de formes) أصبحت واسعة في السنوات الأخيرة. في هذا العمل، قمنا بتقييم العديد من تقنيات آلات التعلم الآلي باستخدام قاعدتين من البيانات: قاعدة بيانات الأرقام وقاعدة بيانات المياه الصناعية. قدمنا تقنية التعلم العميق (Apprentissage Approfondi) التي تهدف إلى تصنيف المياه المنزوعة المعادن (Eaux déminéraliser) باستخدام الأوتو أنكودر (Auto-encodeur) . كما عالجنا في هذا العمل مشكلة التعرف على الأرقام (Reconnaissance des chiffres) . الحلول المقترحة تسمح بالتكيف مع التغيرات في أساليب الكتابة، في اتجاهات الزوايا المختلفة ومختلف خطوط الكتابة؛ من أجل زيادة الأداء استخدمنا شبكة العصبونات من نوع الأوتو أنكودر Autoencodeur و الشبكات العصبونية الالتفافية CNN . و في الأخير، قدمنا دراسة مقارنة من أجل معرفة نقاط الضعف ونقاط القوة لكل تقنية، وكانت لدينا أفضل النتائج من خلال تطبيق تقنية التعلم العميق (Auto-encodeur و CNN).

الكلمات المفتاحية: التعرف على الأنماط، آلة التعلم، التعلم العميق، آلات دعم التمييز SVM، KNN ، الشبكات العصبونية الاصطناعية RNA (MLP)، Auto-encodeur، الشبكات العصبونية الالتفافية CNN.