

الجمهورية الجزائرية الديمقراطية الشعبية
République Algérienne démocratique et populaire

وزارة التعليم العالي و البحث العلمي
Ministère de l'enseignement supérieur et de la recherche scientifique

جامعة سعد دحلب البليدة
Université SAAD DAHLAB de BLIDA

كلية التكنولوجيا
Faculté de Technologie

قسم الإلكترونيك
Département d'Électronique



Mémoire de Master

Filière : Électronique

Spécialité : Electronique des Systèmes Embarqués

présenté par

RAIS Chahinez Imene

&

Allaoui Ilhem

Conception d'un classificateur des arythmies cardiaques à base des réseaux de neurones et son implémentation sur FPGA

Proposée par:

Mme. IZABOUDJEN Nouma

Encadrée par :

Mme. NACEUR Djamila

Année Universitaire 2019-2020

Remerciements

Toute notre gratitude et remerciements avant tout vont à Allah qui nous a donné la santé, la force, la patience le courage et la Volonté pour élaborer ce travail.

Un grand remerciement à notre encadreur **Mme. IZEBOUDJEN Nouma** de nous avoir proposé ce sujet qui nous a mené vers la découverte du monde de la recherche et de l'expérimental. Nous la remercions pour ses conseils précieux et nous comprenons son indisponibilité.

De même, nous adressons notre remerciement à notre promotrice **Mme. NACEUR Djamila** pour sa confiance, son aide et soutien et tous ses conseils précieux qu'elle nous a apporté pour réaliser ce travail.

Les travaux présentés dans ce manuscrit ont été réalisés au sein de la division microélectronique et nanotechnologies (DMN) du Centre de Développement des Technologies Avancées (CDTA) à Alger. Nos vifs remerciements au personnel.

.

Nos remerciements vont également à tous nos enseignements qui nous ont suivis tout au long de nos études et qui ont contribué à notre formation spécialement ceux de l'université SAAD DAHLEB Blida 1.

Nos remerciements vont aux membres du jury d'avoir honoré notre soutenance et pour l'effort fourni afin de juger ce travail.

Enfin, nous remercions tous nos collègues et nos amis qui nous ont apporté leur soutien moral ainsi que tous ceux qui ont contribué de près ou de loin au bon déroulement de ce travail.

Je dédie ce modeste travail à :

Mon père, qui peut être fier de trouver ici le résultat de longues années de sacrifices et de privations pour m'aider à avancer dans la vie. Puisse Dieu faire en sorte que ce travail porte son fruit ; Merci pour les valeurs nobles, l'éducation et le soutien permanent venu de toi.

Ma mère, qui a œuvré pour ma réussite, de par son amour, son soutien, tous les sacrifices consentis et ses précieux conseils, pour toute son assistance et sa présence dans ma vie, reçois à travers ce travail aussi modeste soit-il, l'expression de mes sentiments et de mon éternelle gratitude.

Et

A mes chères sœurs fahima et bouchra, mes chers frères bilal et otman.

Et

A mon mari Mhamed qui m'a énormément aidé

A Toute ma famille.

Et

J'exprime mes sentiments les plus profonds

A Melle chentir Amina qui m'a aidé dans mes études.

A tous mes enseignants du master et licence.

A toute personne qui m'a encouragé ou aidé.

THEM

Je dédie ce travail qui n'aura jamais pu voir le jour sans les soutiens indéfectibles et sans limites de mes chers parents qui ne cessent de me donner avec amour le nécessaire pour que je puisse arriver à ce que je suis aujourd'hui.

Que dieux vous protègent et que la réussite soit toujours à ma portée pour que je puisse vous combler de bonheur.

A ma chère sœur Meriem pour ses encouragements permanents, et son soutien moral.

A mes chers frères, Nazim, Abd el Karim, Nabil, Abd el Raouf et Mohamed pour leur encouragement.

A toute ma famille

A tous ceux qui me sont chers.

A tous mes enseignants.

Que ce travail soit l'accomplissement de vos vœux tant allégués, et le fruit de votre soutien infailible.

Merci d'être toujours là pour moi.

CHAHMEZ MENE

العمل المقدم في هذه الأطروحة يتعلق بشكل أساسي بالتصميم والمحاكاة تحت Matlab ومن ثم التنفيذ على FPGA (مصفوفة البوابة القابلة للبرمجة الميدانية)؛ دائرة متكاملة مخصصة لتصنيف عدم انتظام ضربات القلب باستخدام نهج الشبكة العصبية. من أجل أن نكون قادرين على تصنيف عدد من عدم انتظام ضربات القلب ، نقترح مصنف مورفولوجيا. لإجراء التصنيف، نستخدم تعلم خوارزمية التدرج العكسي (RPG). تتمثل المهمة الرئيسية لجزء البرنامج في تعلم المصنف ضمن مجموعة أدوات الشبكة العصبية في Matlab. (NNTool) نقدم نتائج التصنيف التي تم الحصول عليها في قاعدة بيانات "QT-Database". يقدم جزء الأجهزة مهندساً موازياً لتنفيذ الشبكة العصبية. من أجل التحقق من صحة هذه البنية، اخترنا استخدام لغة وصف VHDL في بيئة VIVADO الخاصة بـ XILINX. يتم تنفيذ وصف VHDL للمصنف على دائرة FPGA لعائلة Artix 7.

كلمات المفاتيح: التصنيف ، عدم انتظام ضربات القلب ، QT-Database ، الشبكة العصبية ، تطبيق FPGA .

Résumé: Les travaux présentés dans ce mémoire portent, essentiellement, sur la conception et la simulation sous Matlab et par la suite l'implémentation sur FPGA (Field Programmable Gate Array) ; d'un circuit intégré dédié à la classification des arythmies cardiaques par l'approche des réseaux de neurones. Afin de pouvoir classer un nombre d'arythmies, nous proposons un classificateur de morphologie. Pour réaliser la classification, nous utilisons l'apprentissage par l'algorithme de la rétropropagation du gradient (RPG). La partie software a pour tâche essentielle de réaliser l'apprentissage du classificateur sous neural network toolbox (NNTool) de Matlab. Nous présentons les résultats de classification obtenus sur la base de données « QT-Database ». La partie hardware propose une architecture parallèle pour l'implémentation du réseau de neurones. Afin de valider cette architecture nous avons opté pour l'utilisation du langage de description VHDL sous l'environnement VIVADO de XILINX. La description VHDL du classificateur est implémentée sur un circuit FPGA de la famille Artix 7.

Mots clés: classification, arythmies cardiaques, QT-Database, réseau de neurones, implementation FPGA.

Abstract: The work presented in this thesis mainly concerns the design and simulation under Matlab and subsequently the implementation on FPGA (Field Programmable Gate Array); of an integrated circuit dedicated to the classification of cardiac arrhythmias using the neural network approach. In order to be able to classify a number of arrhythmias, we propose a morphology classifier. To perform the classification, we use gradient backpropagation (RPG) algorithm learning. The main task of the software part is to learn the classifier under Matlab's neural network toolbox (NNTool). We present the classification results obtained on the basis of "QT-Database" data. The hardware part offers a parallel architect for the implementation of the neural network. In order to validate this architecture we opted for the use of the VHDL description language under the VIVADO environment of XILINX. The VHDL description of the classifier is implemented on an FPGA circuit of the Artix 7 family.

Keywords: classification, cardiac arrhythmias, QT-Database, neural network, FPGA implementation.

Listes des acronymes et abréviations

Modèle ART : Théorie de la Résonance Artificielle.

RNA : réseau de neurone artificiel.

ECG : électro-cardio-gramme .

ANNs : Artificial Neural Networks.

MLP : Multi Layer Perceptron.

ICD : implantable cardio-verter de fibrillation.

DSP :Digital Signal Processor.

ASIC : Application Spécifique Intégration par Ordinateur.

l'ASIP :application spécifique instruction set processor.

RNN :reseaux neural networks.

FANN : fast artificial neural network .

VLSI : neuro-calculateurs à applications spécifique

FPGA :Field Programmable Gate Arrays.

(qtdb) : QT Database.

(NSRDB) : Signal Normal Sinus Rhythm.

(sv) : signal supraventriculaire.

(CUDB) : creighton university Signal ventricular tachyaryhmia.

EMG : électromyogramme.

(RIF) : les filtres à réponse impulsionnelle finie.

(RII) : les filtres à réponse impulsionnelle infinie.

(nntool) : Network Tool.

TRAINGDM :Train gradient descent with momentum backpropagation.

LEARNGDM :learning gradient descent with momentum backpropagation.

MSE :mean square error .

LOGSIG : tangente sigmoïde.

Table des matières

Introduction générale	1
Chapitre I Réseau de neurone	3
I.1 introduction	3
I.2 Historique.....	3
I.3 Les réseaux de neurones	4
I.3.1 Définition.....	5
I.3.1.1 Le Neurone Biologique.....	5
I.3.1.2 Neurone formel	6
I.3.2 Analogie entre le neurone biologique et le neurone formel..	8
I.3.3 Fonction des réseaux de neurones	9
I.3.4 Architecture des réseaux de neurones.....	9
I.3.4.1 Réseaux de neurones non bouclés	9
I.3.4.2 Réseaux de neurones bouclés	11
I.3.5 Apprentissage des réseaux de neurones	12
I.3.5.1 définition	12
I.3.5.2 Principe.....	13
I.3.5.3 Les types d'apprentissage	13
1 L'apprentissage supervise	13
2 L'apprentissage renforcé	14
3 L'apprentissage non supervisé.....	14
I.3.5.4 règles d'apprentissage.....	15
I.4 Le Perceptron multicouche (PMC)	17
I.4.1 Apprentissage de PMC.....	17
I.4.2 Algorithme de la rétro-propagation du gradient d'erreur...	18
I.5 Propriété des réseaux de neurones	21
I.6 Domaines d'application des réseaux de neurones artificiels	22
I.7 Conclusion	22
Chapitre II Classification automatique des signaux cardiaque	23
II.1.introduction	23
II.2.Classification	23
II.2.1 définition.....	23
II.2.2 Principe	24
II.2.2.1 Le classificateur traditionnel.....	25
II.2.2.2 Le classificateur neuronal	26
II.3 La classification neuronale des données médicales	26
II.3.1 Classification des signaux cardiaque.....	26
II.3.1.1 Anatomie de cœur.....	26

II.3.1.2 Le cycle cardiaque	27
II.3.1.3 L'électrocardiographie de la surface.....	28
II.3.1.4 Classification et interprétation de l'ECG.....	28
II.3.1.5 Arythmies cardiaques	29
II.3.2 Les méthodes de classification automatique des signaux ECG	32
II.3.2.1 Synthèse des différentes approches de classification ECG ...	33
II. 3.3 les différents types d'implémentation.....	38
II. 3.3.1 Implémentation software.....	38
II. 3.3.2 Implémentation hardware.....	39
1 les neuro-calculateurs à application générale	39
2 Implémentation VLSI.....	39
II. 4 .Conclusion	40
Chapitre III Implémentation software d'un classificateur neuronal des arythmies cardiaques (CNAC)	43
III.1 Introduction.....	41
III.2 classification des signaux ECG avec de réseaux de neurones.....	41
III 3. Structure du CNAC	42
III.4 Présentation de la base de données	42
III.5 Les anomalies cardiaques traitées	43
III.5.1 MIT-BIH Normal Sinus Rhythm database.....	44
III.5.2 Cu ventriculaire tachyarrythmia Database	44
III.5.3 MIT-BIH Supraventricular Arrhythmia Database	44
III.6 Choix du vecteur d'entrée.....	44
III.7 Prétraitement, traitement et l'implémentation software	44
III.7.1 Prétraitement du signal ECG	44
III.7. 2 Types de bruits présents dans le signal ECG	45
III .7.2.1 bruit d'origine technique	45
III.7.2.2 Bruits d'origine physiques	45
III.7.3 Traitement du signal ECG	46
III.7.3.3. Filtrage numérique	47
III.7.4 Echantillonnage	48
III.7.5 Implémentation software	48
III.7.5.1 Création des données	49

III.7.5.2	Création du réseau	49
III.7.5.3	le bloqué diagramme de réseau de neurone	51
III.7.5.4	apprentissage, simulation et test	51
III.7.5.5	Mesure des performances d'un classificateur.....	56
1	définition des performances d'un classificateur.....	56
III.8	Discussion et conclusion	58
Chapitre IV: Implémentation Hardware du classificateur neuronal des arythmies cardiaques.....		59
IV.1	Introduction	59
IV.2	Les circuits FPGA.....	59
IV.2.1	Définitions.....	59
IV.2.2	Architecture des FPGA.....	60
IV.2.3	Structure de base.....	60
IV.2.4	composition de base de carte FPGA.....	61
IV.2.5	Structure des blocs logiques programmables de notre carte.....	61
IV.2.6	Technologie de programmation pour FPGA (ressource d'interconnexions).....	62
IV.2.7	Programmation des circuits FPGA	63
IV.2.7.1	Langage VHDL	63
IV.2.7.2	Structure générale.....	64
IV.2.7.3	Simulation en VHDL	64
IV.2.7.4	Synthèse	64
IV.3	Implémentation sur FPGA du classificateur neuronale.....	65
IV.3.1	Architecture de neurone	65
IV.3.2	Architecture de réseau de neurons.....	66
IV.3.3	L'outil de conception de Xilinx : Vivado.....	67
IV.4	Description en VHDL.....	68
IV.4.1	Description du Neurone	68
IV.4.1.1	RTL analyse d'un neurone.....	70
IV.4.1.2	Synthèse d'un neurone	71
IV.4.1.3	implémentation d'un neurone.....	72
IV.4.2	description d'une couche cache.....	73
IV.4.2.1	RTL analyse d'une couche cachée.....	73

IV.4.2.2 Synthèse d'une couche cache.....	74
IV .4.2.3 Implémentation d'une couche cache	75
IV.4.3 description d'une Couche de sortie	75
IV.4.4 description d'une couche cachée et couche de sortie	75
IV.4.4.1 RTL analyse.....	75
IV.4.4.2 Synthèse	76
IV.4.4.3 Implémentation.....	77
IV.5 Conclusion.....	77
Conclusion générale	78
Annexe A	80
Annexe B	81
Bibliographie	86

Liste des figures

Figure I.1: neurone biologique.....	6
Figure I.2 : Modèle mathématique du neurone	7
Figure I.3: fonction des réseaux de neurones	9
Figure I4: Schéma de réseau monocouche	10
Figure I.5 : Schéma de réseau perception multicouche	10
Figure I.6 : Schéma d'un réseau de neurones à connexions locales	11
Figure I.7 : Schéma de réseau de neurones bouclé	12
Figure I.8 : Différentes architectures des réseaux de neurones	12
Figure I.9: Principe de l'apprentissage supervisé	13
Figure I.10: Principe de l'apprentissage par renforcement	14
Figure I.11: Principe de l'apprentissage non supervisé	15
Figure I.12 : Perceptron multicouche.....	18
Figure I.13 : Représentation fonctionnelle de l'algorithme de la rétro propagation du gradient	19
Figure II.1: classificateur	24
Figure II.2 : Schéma de principe de la classification supervisée. L'apprentissage (a) permet de fixer les paramètres du modèle grâce aux observations d'entraînement. La phase de prédiction (b) utilise le modèle pour déterminer la classe ou le label	25
Figure II.3: anatomie du cœur	27
Figure II.4: Cycle cardiaque	28
Figure II.5: les phases d'un battement de cœur, tracées dans l'ECG	28
Figure II.6: bradycardie sinusal	30
Figure II.7: extrasystole auriculaire	30
Figure II.8: flutter auriculaire	30

Figure II.9: fibrillation auriculaire	31
Figure II.10: extrasystole ventriculaire	31
Figure II.11: tachycardie ventriculaire	31
Figure II.12: flutter ventriculaire	32
Figure II.13: Fibration ventriculaire	32
Figure II.14 : les différentes méthodes utilisées pour classification d'ECG	33
Figure II.15: méthode arbre	34
Figure II.16 : classificateur neuronal supervisé	37
Figure II.17: classificateur neuronal non-supervisé	38
Figure III.1: Architecture du classifieur des arithmies.....	42
Figure III.2: Base de données www.physioNet.org	43
Figure III.3: de traitement d'ECG	47
Figure III.4: Interface graphique « nntool ».....	48
Figure III.5 : extraction des données.....	49
Figure III.6: Paramètres du RNA dans le cas d'utilisation seul couche cachées et 8 neurones pour la couche.....	50
Figure III.7: Interface graphique « nntool », après la création du réseau appelée network2.....	51
Figure III.8 : bloC diagramme de réseau de neurone.....	51
Figure III.9 : Fenêtre de l'apprentissage de l'outil" nntool".....	52
Figure III.10 : Courbes de régression.....	54
Figure III .11 : fenêtre de simulation.....	55
Figure IV.1: architecture interne d'un FPGA.....	60
Figure IV.2 : FPGA de Xilinx série 7 : tranche de type L (SLICEL).....	62
Figure IV.3: architecture de neurone.....	66
Figure IV.4: le réseau de classificateur.....	67

Figure IV.5: XILINX VIVADO.....	68
Figure IV.6 : programme VHDL d'un neurone.....	70
Figure IV.7: analyse d'un neurone.....	70
Figure IV.8: synthèse d'un neurone.....	71
Figure IV.9: implémentation d'un neurone.....	72
Figure IV.10: RTL analyse de la couche cachée	73
Figure IV.11: résultat de synthèse de la couche cachée	74
Figure IV.12: résultat de RTL analyse de la couche cachée et couche de sortie	75
Figure IV.13: résultat de synthèse de la couche cachée et la couche de sortie	76
Figure IV.14: résultat de l'implémentation de la couche cachée et couche de sortie	77

Liste des tableaux

Tab .I.1: Fonctions de transfert.....	08
Tableau I.2 : Analogie entre le neurone biologique et le neurone formel.....	08
Tableau III.1: la base de données.....	43
Tableau III.2 : représente les entrées et les sorties de système.....	52
Tableau III. 3 : types des signaux de test.....	55
Tableau III.4: résultats de classification.....	56
Tab IV.1 : table résultat de RTL analyse d'un neurone	71
Tab IV.2 : table résultat de la synthèse	71
Tab IV.3 : table RTL analyse de la couche cachée	73
Tab IV.4 : table résultat de la synthèse de la couche cachée	74
Tab IV.5 : table résultat de RTL analyse de la couche cachée et couche de sortie	76
Tab IV.6 : table résultat de la synthèse de la couche cachée et couche de sortie	76

Le cerveau humain est considéré comme le siège de l'intelligence, de la créativité, de l'émotivité, de la conscience et de la mémoire.

Très tôt l'homme s'est intéressé à cet objet complexe. Chacun a une idée assez vague de ce qu'est l'intelligence : la capacité à observer, à comprendre, à se souvenir, à résoudre des problèmes, à apprendre, à créer,...etc.

Dans le monde de la vie quotidienne, on retrouve des applications complexes non linéaires et dynamiques, où le besoin de reconnaissance des objets pour faciliter le travail, à base de l'intelligence humaine se ressent et peut intégrer de nouveaux concepts dans l'objectif de la reconnaissance et de l'intelligence artificielle, tout en s'adaptant à l'environnement et à la capacité d'apprentissage.

Dans le monde, la première cause de mortalité provient des maladies cardiovasculaires. Les causes en sont multiples. Le cœur possède sa propre activité rythmique, et n'a donc pas besoin de stimulus pour se contracter. Les cellules qui le composent, possèdent leur propre excitation électrique, qui est indépendante du système nerveux, ce sont ces cellules qui engendrent le rythme cardiaque.

Nous avons proposé un système qui fait la classification des données cardiaques de certaines maladies pour l'aide au diagnostic des médecins dans leur travail et ainsi développer une application de l'intelligence artificielle basée sur les réseaux de neurones dans le domaine de la santé.

Les réseaux de neurones peuvent être implémentés en software et en hardware (sur des circuits programmables) pour des applications demandant beaucoup de flexibilité et pour assurer plus de performances. De ce fait, on est amené à nous demander : Quels sont les différentes étapes, de la conception à la fabrication, dans la réalisation d'un réseau des neurones artificielle ?

Dans ce contexte, l'objectif premier de ce projet porte sur la conception d'un circuit permettant l'implémentation sur un circuit FPGA (Field Programmable gate Arrays) d'un algorithme intelligent pour la classification des données à partir d'un algorithme d'intelligence artificielle basé sur la rétro propagation du gradient qui, sera intégré sur une carte FPGA composée d'une unité de conversion CAN, d'une unité de traitement (microcontrôleur ou microprocesseur), d'une unité de communication, de mémoires et d'une source d'alimentation (pile ou batterie).

Parmi les applications implémentées sur une carte FPGA, les applications en médecine, les réseaux de capteurs peuvent être utilisés pour assurer une surveillance permanente des organes vitaux de l'être humain grâce à des micro-capteurs qui pourront être avalés ou implantés sous la peau. Ils peuvent aussi

faciliter le diagnostic de quelques maladies en effectuant des mesures physiologiques telles que : la tension artérielle, et les battements du cœur.

La classification de ces mesures physiologiques est effectuée par un algorithme d'apprentissage appelé algorithme de la rétro-propagation du gradient d'erreur. L'idée de base de cet algorithme est de minimiser l'erreur par rapport aux poids de connexion. L'erreur est définie par la différence entre les sorties désirées et les sorties obtenues.

Pour cela, le projet est divisé en 4 grandes étapes détaillées dans les 4 chapitres suivants:

Le premier chapitre présente dans un premier temps un bref historique sur les réseaux de neurones RNA et des généralités sur leurs concepts de base. Nous abordons ensuite le processus d'apprentissage.

Le second chapitre contient les notions sur les signaux cardiaques à savoir l'anatomie cardiaque et les signaux cardiaques qui définissent l'état du cœur.

Le troisième chapitre définit les différentes approches et les techniques utilisées pour l'implémentation des réseaux de neurones ; nous présentons notre implémentation sous Matlab, où nous avons simulé un classificateur des arythmies et calculés les paramètres de classification.

Le quatrième chapitre est dédié à l'intégration du réseau de neurones dans l'architecture FPGA, qui consiste à la création d'une architecture de réseau de neurones, la taille du réseau est déterminée dans la phase de classification, l'implémentation digitale de ce réseau a été faite sous le logiciel xilinx vivado, Et enfin, nous terminerons par une conclusion générale.

I.1 Introduction

Les cellules participant à l'élaboration de toutes les fonctions cérébrales d'un être vivant sont appelées neurones. Cette découverte a été déclarée pour la première fois, en 1906, par Camillo Golgi et Santiago Ramón y Cajal sur la structure et l'organisation du neurone biologique et qui leur a valu le prix Nobel de médecine. Et pour la deuxième fois, en 1943, par McCulloch et Walter Pitts sur le neurone formel qui symbolise le modèle mathématique simplifié du neurone biologique.

Le domaine des réseaux de neurones, qu'il soit naturel ou artificiel, est le siège d'une intense activité scientifique. Il se trouve maintenant à la convergence de recherches et d'applications relevant de la biologie, des mathématiques, des procédures de l'informatique, des techniques de traitement de signal et de l'information.

Dans ce chapitre, nous présentons un aperçu général sur les réseaux de neurones, nous commençons par présenter l'historique de la modélisation des réseaux de neurones artificiels ensuite nous définissons les fondements biologiques sur lesquels reposent les concepts de base des réseaux de neurone tels que, le principe de fonctionnement des réseaux de neurones et leur construction de manière générale.

Nous introduisons par la suite le concept de l'apprentissage, ainsi que les différents types et règles de ce dernier et des différentes architectures des réseaux de neurones en prêtant une attention particulière au perceptron multicouche que nous utilisons dans notre application où l'algorithme nous semble le plus approprié vu les contraintes liées à notre application. Nous présentons également les propriétés générales ainsi que les différents domaines d'application des réseaux de neurones et enfin nous terminons par une conclusion.

I.2 Historique

Depuis l'ère de la cybernétique, l'objectif des chercheurs était de construire une machine capable de reproduire le plus fidèlement possible certains aspects de l'intelligence humaine. Les premières tentatives de modélisation du cerveau sont anciennes. [1]

C'est en 1943 que McCulloch (neuro-physiologiste) et Pitts (logicien) ont proposé les premières notions de neurone formel mimant les neurones biologiques et capables de mémoriser des fonctions booléennes simples.

En 1949, D. Hebb, physiologiste américain explique le conditionnement chez l'animal par les propriétés des neurones eux-mêmes. Ainsi, un conditionnement de type pavlovien tel que, nourrir tous les jours à la même heure un chien, entraîne chez cet animal la sécrétion de salive à cette heure précise

même en l'absence de nourriture. La loi de modification des propriétés des connexions entre neurones qu'il propose explique en partie ce type de résultats expérimentaux.

En 1957, F. Rosenblatt développe le modèle du Perceptron. Il construit le premier neuroordinateur basé sur ce modèle et l'applique au domaine de la reconnaissance de formes. Notons qu'à cet époque les moyens à sa disposition sont limités et c'est une prouesse technologique que de réussir à faire fonctionner correctement cette machine plus de quelques minutes.

Les limites du perceptron monocouche du point de vue possibilité de classification ont été montrées en 1969 par les mathématiciens Minsky et Papert et il a fallu attendre de nouveaux travaux, en particulier ceux de Hopfield en 1982, pour réaliser des réseaux de neurones capables de résoudre des problèmes d'optimisation et ceux de Kohonen pour résoudre les problèmes de reconnaissance et de classification [2].

En 1985, La rétropropagation de gradient apparaît. C'est un algorithme d'apprentissage adapté aux réseaux de neurones multicouches (aussi appelés Perceptrons multicouches). Sa découverte réalisée par trois groupes de chercheurs indépendants indique que dès cette découverte, nous avons la possibilité de réaliser une fonction non linéaire d'entrée/sortie sur un réseau en décomposant cette fonction en une suite d'étapes linéairement séparables [3][4].

La modélisation neuronale biologique est aussi un défi actuel, dont l'objectif est le décryptage du fonctionnement du cerveau en souhaitant progresser jusqu'à la connaissance de la manifestation des sentiments et de la conscience. Malgré les progrès atteints jusqu'aujourd'hui ce domaine reste un champ de recherche ouvert.

I.3 Les réseaux de neurones

Le cerveau capable d'apprendre et de réaliser des raisonnements complexes est constitué d'un très grand nombre de neurones (environ 10^{15}) reliés entre eux (entre 10^3 et 10^4 connexions par neurones).

Les réseaux de neurones est un ensemble d'intersections des éléments primaires (un neurone) modélisé à un neurone biologique à base de concept de neurones. Chaque neurone réalise un traitement et fonctionne indépendamment des autres qui eux fonctionnent en parallèle, donc le réseau réalise le parallélisme de fonctionnement. L'information est stockée dans le réseau sous forme de coefficients synaptiques ou de fonction d'activation, en résumé le réseau ne contient pas de mémoire et pas de zone de calcul, pas de programmation entraîné grâce à l'apprentissage.

Dans la structure des neurones organiques présents au sein d'un cerveau humain, les messages sont véhiculés à travers les différents nœuds interconnectés et ce, jusqu'à la sortie. Le réseau peut être agencé de multiples manières, voire avec des branches reliant la sortie de certains nœuds à d'autres situés en amont (opérant ainsi une boucle de retour des informations) et modéliser de ce fait des fonctions très complexes.[3]

I.3.1 Définition

I.3.1.1 Le Neurone Biologique

Un neurone est une cellule nerveuse constituant la base du système nerveux spécialisée dans le traitement des signaux électriques. En biologie, le cerveau humain contient un grand nombre de neurones fortement interconnectés constituant des réseaux de neurones. Ces neurones vous permettent entre autres, de lire ce texte tout en maintenant une respiration régulière permettant d'oxygéner votre sang, en actionnant votre cœur qui assure une circulation efficace de ce sang pour nourrir vos cellules.

Chaque neurone est une entité autonome au sein du cerveau. Un neurone comprend un corps cellulaire ou cellule somatique ou soma, centre de contrôle de celui-ci, qui fait la somme des informations qui lui parviennent. Il traite ensuite l'information et renvoie le résultat sous forme de signaux électriques, du corps cellulaire à l'entrée des autres neurones au travers de son axone.

Les axones reliant les neurones entre eux jouent donc un rôle important dans le comportement logique de l'ensemble. Le neurone est également constitué de plusieurs branches nommées dendrites, qui sont les récepteurs principaux du neurone, par lesquelles transite l'information venue de l'extérieur vers le corps cellulaire. Les synapses du neurone quant à elles reçoivent les informations des autres neurones via l'axone et permettent donc aux neurones de communiquer entre eux [3][4]

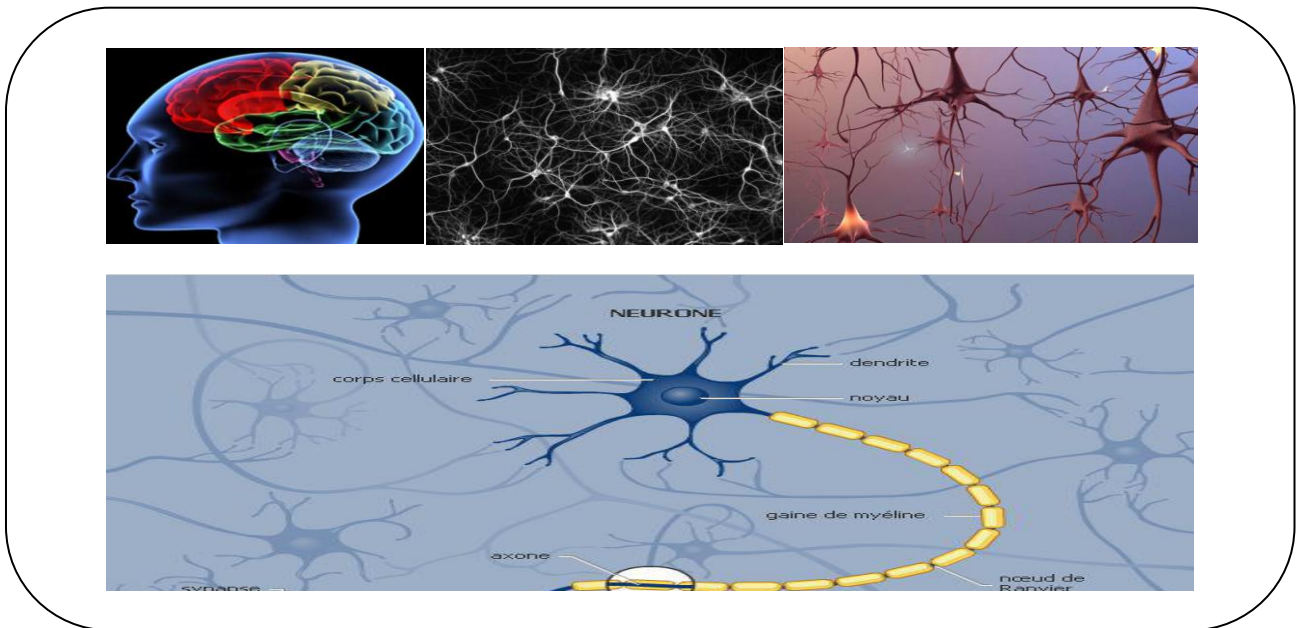


Figure I.1: neurone biologique

I.3.1.2 Neurone formel

Le neurone formel est un modèle mathématique simplifié du neurone biologique, il présente un certain nombre d'entrées (les dendrites), un corps traitant les entrées suivant la méthode du tout ou rien, et un axone véhiculant la réponse du neurone.

A. Structure

Les réseaux de neurones artificiels (formels) sont des modèles à base de neurones biologiques, chaque neurone artificiel est un processeur élémentaire de calcul simple. Il reçoit un nombre variable d'entrées en provenance de neurones amont. À chacune de ces entrées est associé un poids (W) abréviation de poids synaptique représentatif de la force de la connexion. Chaque processeur élémentaire est doté d'une sortie unique, qui se ramifie ensuite pour alimenter un nombre variable de neurones avals [3].

B. Comportement

Nous distinguons deux phases :

1/ Le calcul de la somme pondérée des entrées (Y) par les poids respectifs des connexions, de $n+1$ signaux x_0, x_1, \dots, x_n qui lui parviennent (Figure I.2), selon l'expression suivante :

$$Y = \sum (w_n \cdot x_n) \quad (\text{I-1})$$

2/ A partir de cette valeur, une fonction de transfert (d'activation) calcule la valeur de l'état du neurone.

$$Z = f(Y)$$

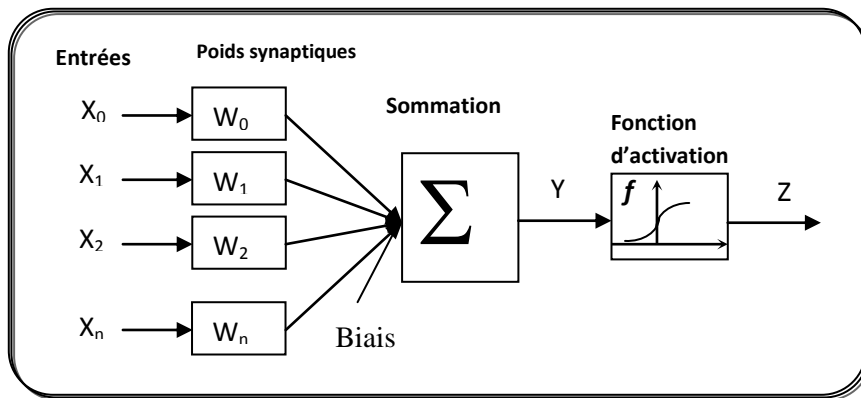











Figure I.2 : Modèle mathématique du neurone

Les coefficients de pondération W_i , $i=0,1, \dots, n$ s'appellent les poids synaptiques, ils représentent la force de la connexion entre neurones.

Le neurone créé par McCulloch et Pitts est un neurone utilisant une fonction d'activation f binaire à seuil de type tout ou rien prenant les valeurs 0 ou 1 [1]. La valeur de la sortie du neurone est donc calculée par la fonction de transfert et c'est cette valeur qui sera transmise aux neurones avals.

Jusqu'à présent, nous n'avons pas spécifié la nature de la fonction d'activation de notre modèle. Il se trouve que plusieurs possibilités existent.

Différentes fonctions de transfert pouvant être utilisées comme fonction d'activation du neurone sont énumérées au tableau I.1. Les trois les plus utilisées sont les fonctions «seuil», «linéaire » et «sigmoïde».

Nom de la fonction	Relation d'entrée/sortie	Icône	Nom Matlab
seuil	$a = 0$ si $n < 0$ $a = 1$ si $n \geq 0$		hardlim
seuil symétrique	$a = -1$ si $n < 0$ $a = 1$ si $n \geq 0$		hardlims
linéaire	$a = n$		purelin
linéaire saturée	$a = 0$ si $n < 0$ $a = n$ si $0 \leq n \leq 1$ $a = 1$ si $n > 1$		satlin
linéaire saturée symétrique	$a = -1$ si $n < -1$ $a = n$ si $-1 \leq n \leq 1$ $a = 1$ si $n > 1$		satlins
linéaire positive	$a = 0$ si $n < 0$ $a = n$ si $n \geq 0$		poslin
sigmoïde	$a = \frac{1}{1+\exp^{-n}}$		logsig
tangente hyperbolique	$a = \frac{e^n - e^{-n}}{e^n + e^{-n}}$		tansig
compétitive	$a = 1$ si n maximum $a = 0$ autrement		compet

Tab .I.1: Fonctions de transfert

I .3.2 Analogie entre le neurone biologique et le neurone formel

Nous pourrions résumer cette modélisation par le tableau I.2, qui nous permettra de voir clairement la transition entre le neurone biologique et le neurone formel.

Neurone biologique	Neurone formel
Synapses	Poids de la connexion
Axones	Sorties
Dendrites	Entrées
Noyau ou Somme	Fonction d'activation

Tab I.2: Analogie entre le neurone biologique et le neurone formel.

I.3.3. Fonction des réseaux de neurones

Un neurone possède un ou plusieurs connexions entrantes, à chacune d'elle est associée un poids, si la somme pondérée des entrées est inférieure à un seuil, le neurone reste inactif, si cette somme dépasse le seuil, le neurone devient actif et émet un signal avant de retourner au repos.

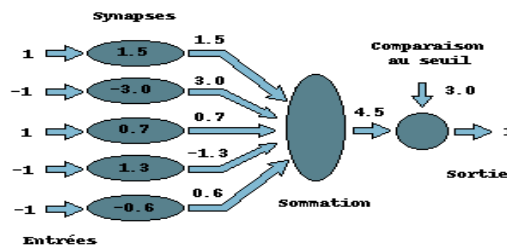


Figure I.3: fonction des réseaux de neurones

Les réseaux de neurones se distinguent par leur topologie (architecture) et par leur mode d'apprentissage [1].

I.3.4 Architecture des réseaux de neurones

Un réseau de neurones est en général composé de plusieurs couches de neurones, des entrées jusqu'aux sorties.

Les réseaux de neurones peuvent être regroupés en deux catégories : réseaux de neurones bouclés et réseaux de neurones non bouclés.

I.3.4.1 Réseaux de neurones non bouclés

Les «Feed-forward networks» sont représentés graphiquement par un ensemble de neurones « connectés » entre eux. L'information circulant des entrées vers les sorties sans retour en arrière, si nous présentons le réseau comme un graphe dont les nœuds sont des neurones et les arêtes sont les connexions entre ceux-ci, le graphe d'un réseau non bouclé est acyclique. Ainsi, si nous nous déplaçons dans le réseau, à partir d'un neurone quelconque, en suivant les connexions, nous ne pouvons pas revenir au neurone de départ [7].

Dans cette catégorie, nous trouvons les réseaux monocouche (perceptron) et les réseaux multicouches (perceptron multicouche).

Les réseaux de neurones non bouclés sont des objets statiques ; Ils sont utilisés principalement pour effectuer des tâches d'approximation de fonction non linéaire, de classification ou de modélisation de processus statiques non linéaires.

a. Réseau de neurones monocouche

Une couche de suite de neurones connectée à des entrées obtient des sorties par couche modifiable de poids.

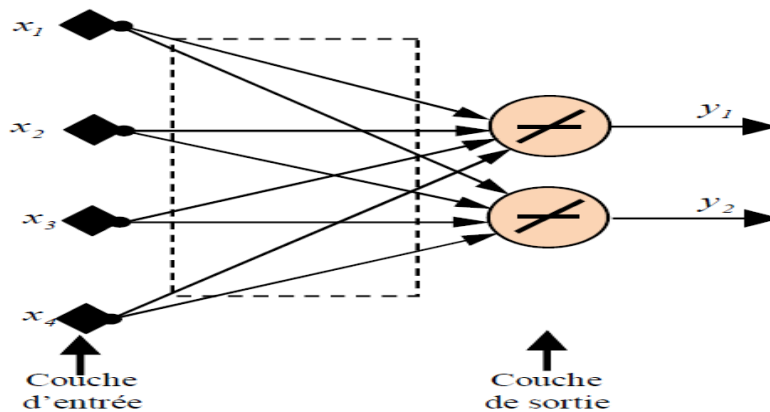


Figure I.4 : Schéma de réseau monocouche

b. Réseau de neurones multicouche

Il est constitué par deux couches ou plus, chaque couche rassemble plusieurs couches non connectés entre eux, par contre chaque neurone de couche est connecté à tous les neurones des autres couches. Ce réseau est constitué de couche d'entrée, de couche de sortie et des couches intermédiaires n'ayant aucun contact avec l'extérieur et qui sont appelées les couches cachées.

Le réseau des neurones non bouclé a une structure particulière, très fréquemment utilisé : une couche d'entrée, deux couches intermédiaires et une couche de sortie, il est appelé « perception multicouche » et est très puissant, sa fonction d'activation est une sigmoïde [7][1][3].

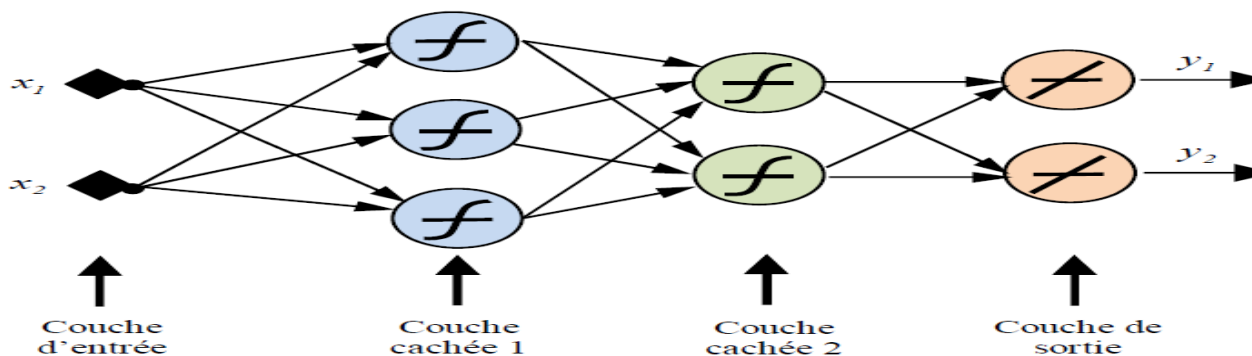


Figure 1.5: Schéma de réseau perception multicouche

c. Réseaux de neurones à connexions locales

Il s'agit d'une structure multicouche, mais qui à l'image de la rétine conserve une certaine topologie. Chaque neurone entretient des relations avec un nombre réduit et localisé des neurones de la couche avale. Les connexions sont donc moins nombreuses que dans le cas d'un réseau multicouche classique [1][2].

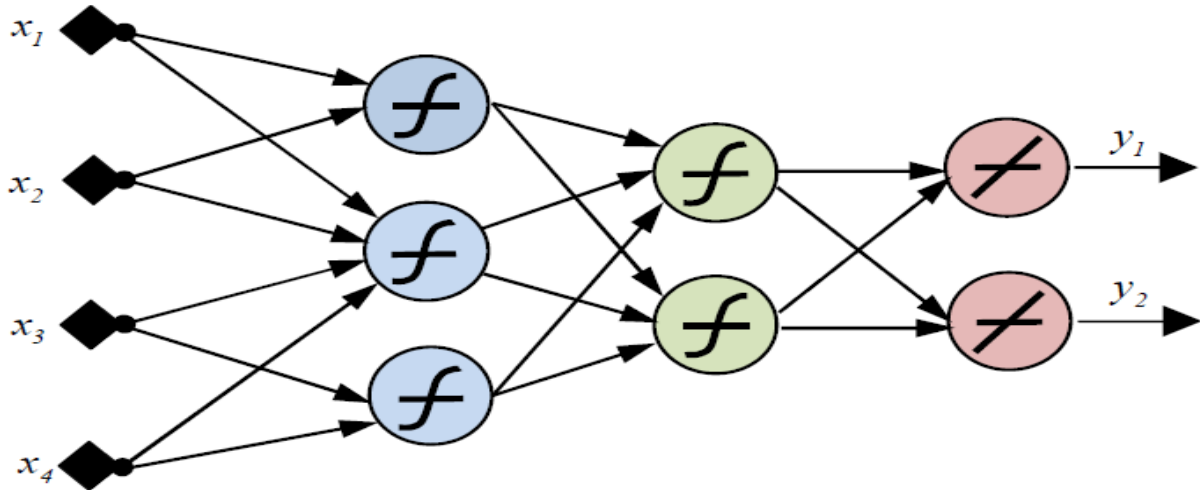


Figure 1.6: Schéma d'un réseau de neurones à connexions locales

I.3.4.2 Réseaux de neurones bouclés

- Les réseaux récurrents bouclés ou «Feed-back networks» présentés dans la figure I.7, sont des réseaux dont le graphe des connexions est cyclique : lorsque nous nous déplaçons dans le réseau en suivant le sens des connexions, il est possible de trouver au moins un chemin qui revient à son point de départ (un tel chemin est désigné sous le terme de « cycle ») La sortie d'un neurone peut donc être fonction d'elle-même [7]. Dans cette catégorie, nous trouvons les réseaux compétitifs, le réseau de Kohonen, le réseau de Hopfield et le modèle ART. La figure I.8 montre les différentes topologies des réseaux de neurones.

Pour qu'un tel système soit causal, il faut évidemment qu'à toute boucle soit associé un retard : un réseau de neurones bouclé est donc un système dynamique, régi par des équations différentielles. Les réseaux de neurones bouclés sont utilisés pour effectuer des tâches de modélisation de systèmes dynamiques, de commande de processus, ou de filtrage.

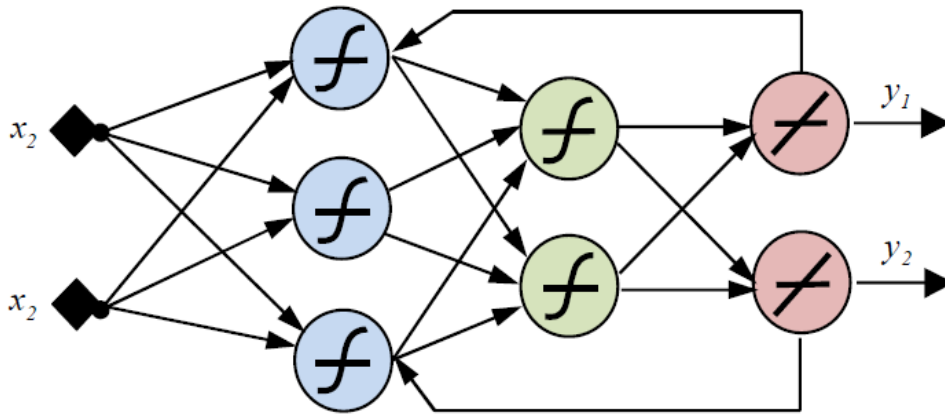


Figure I.7: Schéma de réseau de neurones bouclé

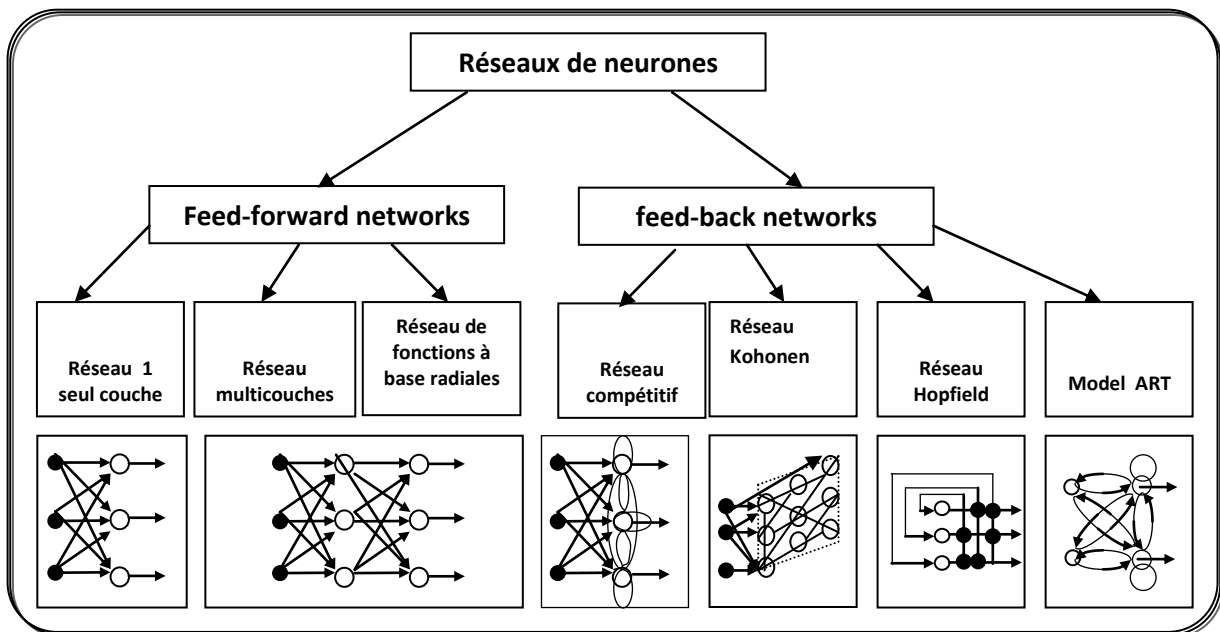


Figure I.8 : Différentes architectures des réseaux de neurones

I.3.5 Apprentissage des réseaux de neurones

L'apprentissage consiste à adapter la connaissance de RNA au problème posé. La particularité du réseau de neurones comme outil mathématique repose sur sa capacité d'apprentissage ; grâce à un processus d'entraînement [1].

I.3.5.1 définition

L'apprentissage est une phase de développement d'un réseau de neurone durant laquelle le réseau de neurones modifie sa structure interne jusqu'à l'obtention du comportement désiré [7].

I.3.5.2 Principe

Nous pouvons modifier les caractéristiques du réseau (les poids de ses liaisons par exemple) en réaction aux stimuli extérieurs que nous lui soumettons (les valeurs d'entrées), de manière à ce qu'il réagisse différemment si un même stimulus lui est appliqué ultérieurement. Le réseau s'améliore ainsi car à chaque erreur qu'il fait, il subit une correction qui le fait réagir différemment s'il est confronté à la même situation. Le but étant qu'une fois l'apprentissage terminé, le réseau effectue la tâche pour laquelle il a été conçu sans se tromper, c'est-à-dire qu'il fournit pour chaque stimulus d'entrée la « bonne » sortie, la sortie désirée par l'opérateur [1].

I.3.5.3 Les types d'apprentissage

1. Apprentissage supervisé

L'apprentissage supervisé est caractérisé par la présence de « professeur » qui correspond à la sortie désirée ou correcte de chaque entrée, afin de comparer le résultat obtenu avec le résultat désiré, les poids des connexions sont alors réajustés pour diminuer l'erreur [7].

En pratique, cette connaissance de professeur prend la forme d'un ensemble de Q couples de vecteurs d'entrée et de sortie que nous noterons $\{(x, d), (x, d), \dots, (x, d)\}$, x étant l'entrée et d la sortie. Un couple correspond à un espace de données sur le réseau, cet apprentissage permet de qualifier le réseau d'apprentissage par correction d'erreur [2].

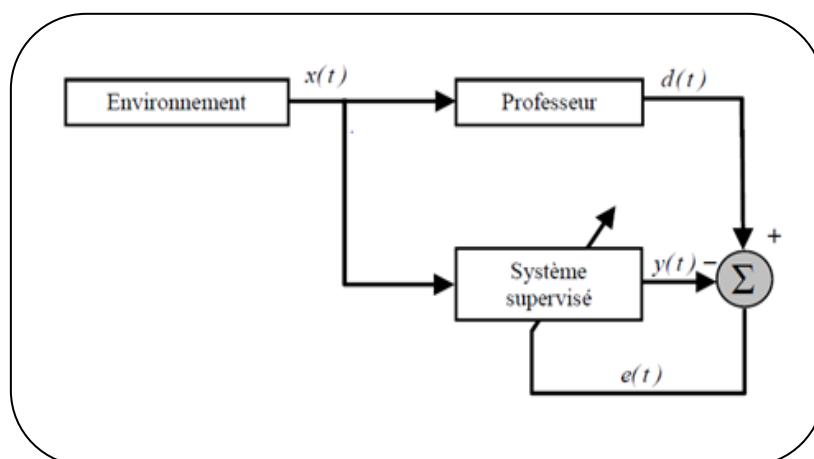


Figure I.9: Schéma bloc de l'apprentissage supervisé.

Dans la figure I.9, le professeur et le système supervisé du réseau de neurones reçoit une entrée $x(t)$ c'est une information de l'environnement, le professeur a une sortie désirée $d(t)$ et le réseau de neurone a une sortie de traitement pratique $y(t)$ soustraite à $d(t)$ pour obtenir l'erreur, cette dernière va être injectée au système où le réseau de neurones va modifier son comportement via une procédure itérative qui, éventuellement, lui permet de simuler la réponse du professeur. Par la suite, on peut éliminer le professeur et laisser le réseau fonctionner de façon autonome.

2. Apprentissage renforcé

Ce type d'apprentissage est une partie de l'apprentissage supervisé, figure I.10 mais avec un indice qui est un scalaire au lieu d'un signal d'erreur vectoriel. Cet indice est imprécis sur le comportement final «échec ou succès» du réseau. L'ajustement des poids du réseau se fait uniquement à partir d'une simple évaluation de la qualité de sa réponse [2].

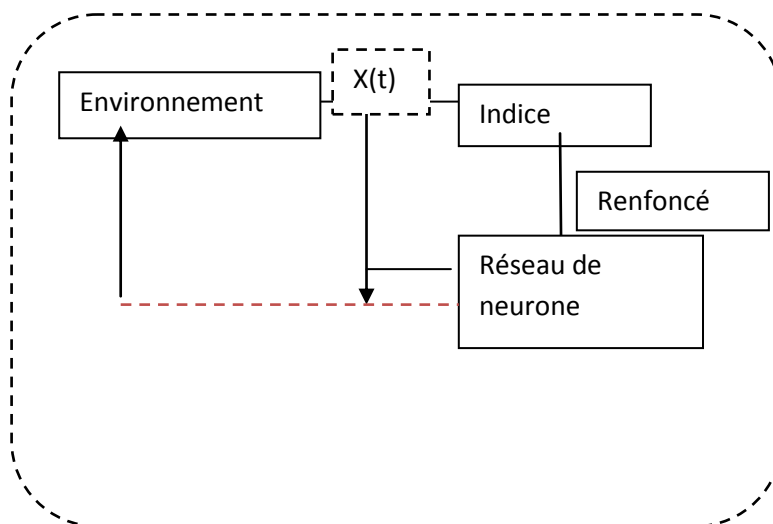


Figure I.10 : Principe de l'apprentissage par renforcement.

3. Apprentissage non supervisé

Cet apprentissage est caractérisé par l'absence de « professeur », pas de sortie désirée à chaque entrée, l'entrée ou l'information de l'environnement, est connectée directement à l'entrée du système de réseau de neurones. il n'y a pas d'estimation d'erreur pour corriger le système. Il va servir par la suite à l'optimisation des paramètres libres interne du réseau, c'est à-dire des poids synaptiques. À la fin de l'apprentissage, le réseau a développé une habilité à former des représentations internes des stimuli de l'environnement permettant d'encoder les caractéristiques de ceux-ci et, par conséquent, de créer automatiquement des classes de stimuli similaires.

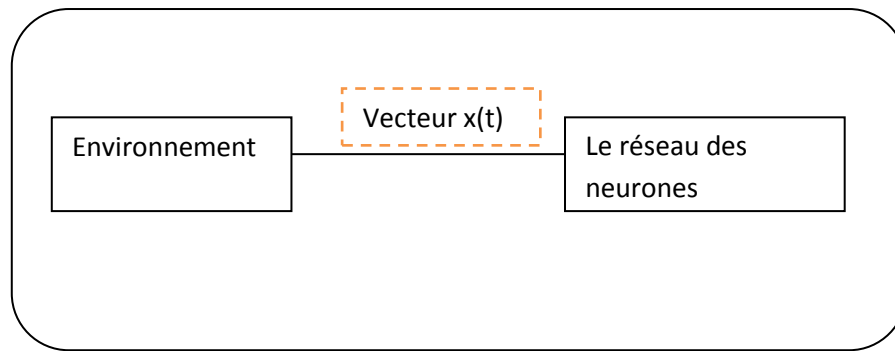


Figure I .11 : Principe de l'apprentissage non supervisé

4. règles d'apprentissage

L'apprentissage se traduit par un changement dans la valeur des poids qui relient les neurones d'une couche à l'autre. La méthode d'ajustement des poids synaptiques pendant l'apprentissage du réseau peut être choisi parmi un ensemble de règles citées ci-dessous : [3]

A. apprentissage par la règle de Hebb

Cette règle a été proposée par le neurophysiologiste Donald Hebb en 1949. Dans un contexte neurobiologique, Hebb cherchait à établir une forme d'apprentissage associative au niveau cellulaire. Il s'applique aux connexions entre neurones, La règle de Hebb s'exprime de la façon suivante : "*Si deux cellules sont activées en même temps alors la force de la connexion augmente*".

La loi de Hebb peut être modélisée par les équations suivantes :

$$W_{ij}(t+1) = W_{ij}(t) + \eta S_i S_j \quad (I-6)$$

Où,

$W_{ij}(t)$ est le poids de la connexion reliant les neurones S_i et S_j à l'étape t .

$W_{ij}(t+1)$ est le nouveau poids à l'étape $t+1$.

η : est un nombre compris entre 0 et 1, représentant le taux d'apprentissage.

t : représente l'étape d'apprentissage.

L'algorithme d'apprentissage modifie de façon itérative les poids pour adapter la réponse obtenue à la réponse désirée. Il s'agit en fait de modifier les poids lorsqu'il y a erreur seulement.

B. Apprentissage par la règle du Perceptron [2]

La fonction d'activation est une fonction discrète. Les sorties prennent des valeurs binaires (0 ou 1). La règle est la suivante :

$$\left\{ \begin{array}{ll} W_{ij}(t+1) = W_{ij}(t) + \eta x_i & \text{Si la sortie actuelle est égale à 0 et doit être égale à 1.} \\ W_{ij}(t+1) = W_{ij}(t) - \eta x_i & \text{Si la sortie actuelle est égale à 1 et doit être égale à 0.} \\ W_{ij}(t+1) = W_{ij}(t) & \text{Si la sortie est correcte.} \end{array} \right.$$

Où

$\eta > 0$: représente le coefficient d'apprentissage.

t : l'étape d'apprentissage.

x_i : l'entrée du neurone i .

W_{ij} : le poids synaptique ou connexion entre neurone i et le neurone j .

L'algorithme d'apprentissage du Perceptron est semblable à celui utilisé pour la loi de Hebb. Les différences se situent au niveau de la modification des poids.

L'inconvénient majeur du Perceptron est qu'il ne s'applique que si les classes sont linéairement séparables.

C. Apprentissage par la règle de Widrow-Hoff (règle delta)

Widrow et Hoff ont étudié l'algorithme d'apprentissage du Perceptron en considérant une fonction d'activation continue et différentielle. Le principe est le suivant :

- Calculer l'erreur quadratique selon la formule :

$$E = \sum_{j=1}^n (d_j - y_j)^2$$

Avec :

$$y_i = \sum_{i=1}^m x_i W_{ji}$$

- Minimiser cette erreur en modifiant les poids de chaque neurone suivant la règle :

Où,

n : nombre de neurones à la sortie

d_i : est la sortie désirée

y_j : est la sortie calculée

x_i : l'entrée i du neurone j

m : le nombre de neurones à l'entrée

D. apprentissage par la règle delta généralisée

La règle delta généralisée appelée règle de Rétro-propagation du gradient est une généralisation de la règle de Widrow-Hoff et s'applique à un réseau multicouche utilisant des fonctions d'activation différentiables et un apprentissage supervisé. Nous verrons en détail cette règle dans la prochaine section [3].

I. 4 Le Perceptron multicouche (PMC)

Le Perceptron multicouche est un réseau orienté de neurones artificiels organisés en couches et l'information ne circule que dans un seul sens.

Le perceptron multicouche a été développé pour résoudre le fameux problème du non séparabilité linéaire qui limitait les applications possibles des réseaux de neurones depuis de nombreuses années. Par conséquent, c'est le modèle qui a suscité le plus d'intérêt de la part des chercheurs et qui suscite encore beaucoup d'études et d'applications. En résumé, ce réseau est constitué de plusieurs couches de Perceptron, une couche d'entrée, une ou plusieurs couche(s) cachée(s) et une couche de sortie. L'information circule de l'entrée vers la sortie à travers la (les) couche(s) cachée(s). Chaque neurone du réseau représente un automate linéaire généralisé dont la fonction d'activation est une sigmoïde. Les neurones sont reliés entre eux par des connexions pondérées. Ce sont les poids de ces connexions qui gouvernent le fonctionnement du réseau et programment une application de l'espace des entrées vers l'espace des sorties à l'aide d'une transformation non linéaire ; comme représenté dans la figure (figure I.5). [2][6].

I.4.1 Apprentissage du réseau PMC

Dans ce cas, l'apprentissage est supervisé, c'est-à-dire que nous présentons au réseau, en même temps, une forme et son modèle. La fonction de transfert utilisée est une fonction sigmoïde, dont la dérivabilité joue un rôle important. L'apprentissage dans ce type de structure consiste à appliquer à l'entrée du réseau une base de données constituées de couples de valeurs (entrée, sortie désirée), et puis vérifier s'il s'agit de la bonne sortie qui est obtenue.

I.4.2 Algorithme de la rétro-propagation du gradient d'erreur

L'algorithme de rétro-propagation a été développé en particulier par Rumelhart et Parkenet le Cun en 1985. La technique de rétropropagation est une méthode qui permet de calculer le gradient de l'erreur pour chacun des neurones du réseau, de la dernière couche vers la première. On appelle souvent technique de rétropropagation du gradient, l'algorithme classique de correction des erreurs basé sur le calcul du gradient grâce à la rétro propagation, cette méthode permet de la correction d'erreur de manière significative les coefficients synaptiques à engendrer une erreur importante tout en pondérant également les neurones générant une erreur moins conséquente.

Il s'agit donc de calculer la sortie réelle de chaque neurone de la jème couche. Cette procédure est effectuée de proche en proche de la couche d'entrée vers celle de sortie. La valeur des sorties dépend des paramètres liés à la structure du réseau : topologie, fonctions d'agrégation et d'activation ainsi que les coefficients synaptiques. Ce sont généralement ces derniers qui sont modifiés, cela n'est pas toujours le cas. La modification peut également intervenir au niveau des fonctions d'agrégation ou d'activation.

Cela est appelée « propagation d'avant ». Par la suite l'erreur sera calculée puis propagée dans le réseau, ce qui donne lieu à une modification des poids. Le processus est répété sur tous les exemples jusqu'à obtenir une erreur considérée négligeable [6][3].

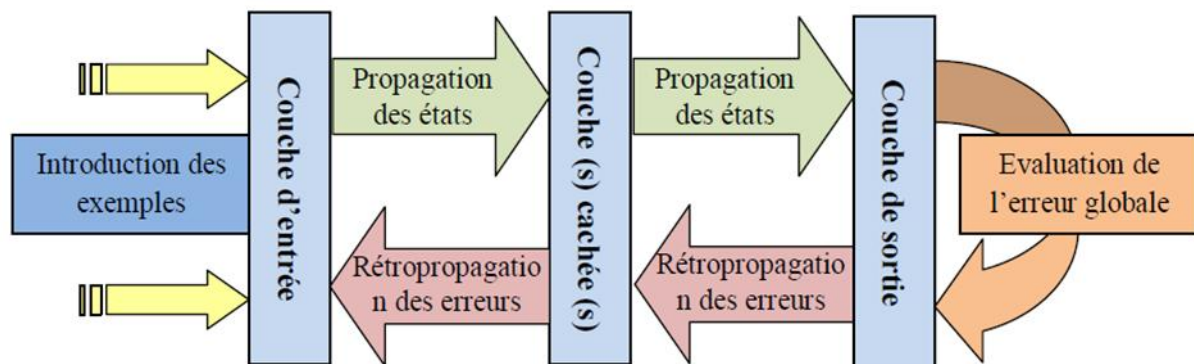


Figure I.12 : Apprentissage des réseaux de neurone par l'algorithme de rétropropagation

• Calcul détail

L’algorithme de la Rétro-propagation standard utilise la méthode de plus forte descente pour la minimisation de la moyenne carrée de l’erreur locale, qui est donnée par :

Soit $x(n)$ et $s(n)$ correspondant à l’entrée et la sortie respectivement du réseau avec n désignant la $n^{ième}$ donnée d’apprentissage. L’algorithme de rétro propagation consiste à mesurer l’erreur entre les sorties ciblées $S(n)$ et les sorties observées $Y(n)$ résultat de la propagation vers l’avant des informations d’entrées $X(n)$ et à rétro propager cette erreur des sorties vers les entrées. L’algorithme de rétro propagation procède donc à l’adaptation des poids neurone par neurone en commençant par la couche de sortie [6][3] :

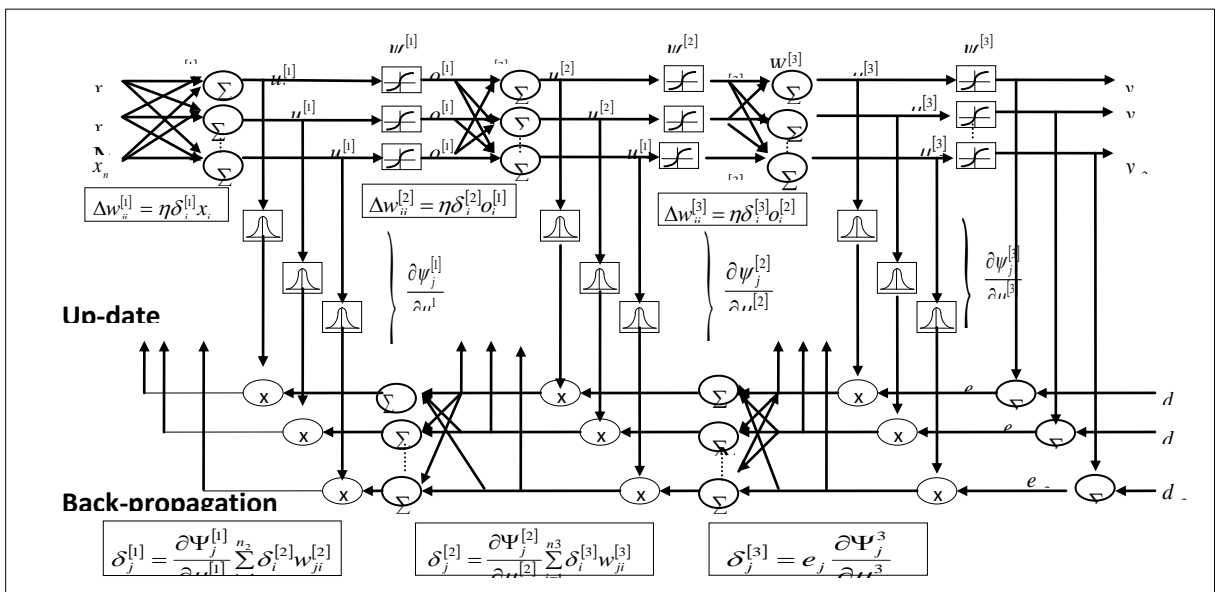


Figure I.13 : Représentation fonctionnelle de l’algorithme de la rétro propagation du gradient

Soit $e_i(n)$ l’erreur observée pour le neurone de sortie i et le motif d’apprentissage n :

$$e_i(n) = S_i(n) - Y_i(n) \tag{I.5}$$

Avec $d_i(n)$ les sorties ciblées et $Y_i(n)$ les sorties observées pour le neurone i

$$E_p = 1/2 \sum_{j=1}^n 1/2 (d_{jp}(n) - Y_{jp}(n))^2 = \sum_{j=1}^n e_{jp}^2 \tag{I.6}$$

Et de l’erreur globale donnée par :

$$ET = \sum EP \tag{I.7}$$

Où, d_{ij} et Y_{ij} représentent la sortie désirée et la sortie actuelle du j -ième neurone de sortie pour le i -ième couche.

L'objectif de l'algorithme est d'adapter les poids des connexions du réseau de manière à minimiser la somme des erreurs sur tous les neurones de sortie, la sortie $Y_j(n)$ du neurone i est définie par :

$$Y_j(n) = \Psi[V_j(n)] = \Psi\left[\sum_{j=0}^r (W_{ji}(n)Y_j(n))\right] \quad (I.8)$$

Avec $\Psi[\]$ la fonction d'activation du neurone, $V_j(n)$ la somme pondérée des entrées du neurone j , W_{ij} le coefficient synaptique entre le neurone i de la couche précédente et le neurone j de la couche courante et $Y_j(n)$ la sortie de ce même neurone j .

On considère que la couche précédente contient r neurones, que W_{j0} correspond au biais du neurone j et que $Y_0(n) = -1$

Pour corriger l'erreur observée, il faut modifier le coefficient $W_{ij}(n)$ dans le sens opposé au gradient de l'erreur calculé par la dérivée partielle suivante :

$E(n)$: l'erreur total

$$\frac{(\partial E_p(n))}{(\partial W_{ji}(n))} \quad (I.9)$$

L'objectif étant d'atteindre le minimum local en modifiant le coefficient synaptique dans sens ou dans l'autre. On exprime la variation de poids par la formule suivante :

$$\Delta W_{ij}(n) = -\eta (\partial E_p(n)) / (\partial W_{ij}(n)) \quad (I.10)$$

Avec $0 < \eta < 1$ représentant le taux d'apprentissage de l'algorithme.

En utilisant la règle de chaînage des dérivées partielles [Règle qui dit que $\frac{\partial f(y)}{\partial x} = \frac{\partial f(y)}{\partial y} = \frac{\partial y}{\partial x}$] on obtient :

$$\frac{(\partial E_p(n))}{(\partial w_{ij}(n))} = -E_{pj}(n) Y_j(n) [1 - Y_j(n)] Y_j(n) \quad (I.11)$$

Ce qui, associé à la règle du « Delta » exprimant la variation de poids, donne :

$$\Delta W_{ji}(n) = -\eta \partial E(n) / \partial W_{ji}(n) = -\eta \delta_j(n) Y_j(n) \quad (I.12)$$

$$\text{Avec } \delta_j(n) = E_{pj}(n) Y_j(n) [1 - Y_j(n)] \quad (I.13)$$

Ce qui correspond au gradient local de la couche de sortie. Hors, le réseau dispose encore de deux couches cachées sur lesquels la rétropropagation doit également s'effectuer, considérons donc à présent la seconde couche cachée du réseau (toutes les couches se traitant ensuite de la même façon).

L'objectif restant le même, c'est-à-dire, d'adapter les coefficients des connexions entre la couche précédente et la couche courante. Les indices i et j désignent

toujours respectivement un neurone de la couche précédente et un neurone de la couche courante. Enfin, l'indice k désigne un neurone de la couche suivante.

Si nous réutilisons la règle des dérivées partielles de l'erreur totale $E(n)$ nous devons réévaluer le terme de la dérivée partielle selon la sortie observée de la couche précédente : $k \in C$

$$(\partial E(n)) / (\partial Y_i(n)) = \sum (\delta_k(n) W_{kj}(n)) \quad (I.14)$$

Ce qui nous amène alors à formuler :

$$\frac{\partial E(n)}{\partial W_{ji}(n)} = -Y_j(n)[1 - Y_j(n)] \left[\sum_{k \in c} \delta_k(n) W_{kj}(n) \right] Y_j(n) \tag{I.15}$$

$$\text{Et } \Delta w_{ji}(n) = -\eta \frac{\partial E(n)}{\partial W_{ji}(n)} = -\eta \delta_j(n) Y_i(n) \tag{I.16}$$

$$\text{Avec } \delta_j(n) = Y_j(n)[1 - Y_j(n)] \sum_{k \in c} \delta_k(n) W_{kj}(n) \tag{I.17}$$

Ces deux équations sont utilisables pour la totalité des couches cachées, quel que soit leur nombre, toutefois, pour la première couche cachée, il faut remplacer $Y_i(n)$ par $X_i(n)$.

La procédure de rétro propagation peut se résumer à la série d'étapes suivantes :

1. Initialiser tous les poids à de petites valeurs aléatoires dans l'intervalle $[-0.5, 0.5]$.
2. Normaliser les données d'entraînement.
3. Permuter aléatoirement les données d'entraînement.
4. Pour chaque donnée d'entraînement n :
 - (a) Calculer les sorties observées en propageant les entrées vers l'avant ;
 - (b) Ajuster les poids en rétro propageant l'erreur observée :

$$\begin{aligned} \delta_j^{[2]} &= -\frac{\partial E_p}{\partial \mu_j^{[2]}} \\ &= -\frac{\partial E_p}{\partial O_j^{[2]}} \cdot \frac{\partial O_j^{[2]}}{\partial \mu_j^{[2]}} \dots \\ \text{Avec } \mu_j^{[2]} &= \sum_{i=1}^{n_1} W_{ji}^{[2]} x_i^{[2]} = \sum_{i=1}^{n_1} W_{ji}^{[2]} O_i^{[1]} \quad (j=1, \dots, n_2) \end{aligned} \tag{1}$$

$$\begin{aligned} \delta_j^{[1]} &= \frac{\partial \Psi^{[1]}}{\partial \mu_j^{[1]}} \sum_{i=1}^{n_2} \delta_i^{[2]} W_{ij}^{[2]} \dots \\ \mu_j^{[1]} &= \sum_{i=1}^{n_0} W_{ji}^{[1]} x_i \quad (j=1, \dots, n_1) \end{aligned} \tag{2}$$

Où le gradient local est défini par (1) s'il s'agit d'une couche de sortie et par (2) si c'est une couche cachée: D'où $Q_j(n) = Y_j(n) = \Psi[V_j(n)]$

5. Répéter les étapes 3 et 4 jusqu'à un nombre maximum d'itérations ou jusqu'à ce que l'erreur soit en dessous d'un certain seuil[6].

Tous les exemples d'apprentissage sont présentés périodiquement jusqu'à ce que les poids synaptiques soient stabilisés, i.e. jusqu'à ce que l'erreur pour tout l'ensemble complet soit acceptable et le réseau converge.

I.5 Propriétés des réseaux de neurones

Le réseau de neurones a des propriétés suivantes :

- La capacité d'apprentissage

La capacité d'apprentissage se traduit par la capacité du réseau de neurones à apprendre comme l'être humain

- La capacité de généralisation

La capacité de généralisation se traduit par la capacité d'un système à apprendre et à retrouver à partir d'un ensemble d'exemples des règles qui permettent de résoudre un problème donné non appris.

- Le parallélisme

Dépend de l'architecture des réseaux de neurones considérés comme un ensemble d'entités élémentaires qui travaillent simultanément, c'est à dire que chaque neurone traite une donnée seule et en parallèle à d'autres neurones. Le parallélisme permet une rapidité de calcul supérieure [7].

I.6 Domaines d'application des réseaux de neurones artificiels

Aujourd'hui, les réseaux de neurones artificiels ont de nombreuses applications dans des secteurs très variés :

- **Traitement d'images:** reconnaissance de caractères et de signatures, compression d'images, reconnaissance de forme, cryptage, classification, etc.
- **Traitement du signal:** filtrage, classification, identification de source, traitement de la parole... etc.
- **Contrôle:** commande de processus, diagnostic, contrôle qualité, asservissement des Robots, systèmes de guidage automatique des automobiles et des avions...etc.
- **Défense:** guidage des missiles, suivi de cible, reconnaissance du visage, radar, sonar, Lidar, compression de données, suppression du bruit...etc.
- **Optimisation:** planification, allocation de ressources, gestion et finances, etc.
- **Simulation:** simulation du vol, simulation de boîte noire, prévision météorologique, recopie de modèle... etc.
- **Médicale :** analyse des signaux EEG, ECG, prothèses, analyse du cancer,etc. [2][3][4].

I.7 Conclusion

Cette recherche nous a mené au constat suivant : plusieurs applications ont besoin de contrôle et de gestion, demandant plus de temps, plus de calculs et plus de manipulation de variables dont le nombre ne cesse d'augmenter.

Dans ce chapitre, nous avons présenté le contexte de notre travail où nous avons vu un aperçu général sur les notions des réseaux de neurones : neurones formels, fonction d'activation, différentes topologies de réseau et les types d'apprentissage.

Par la suite, nous nous sommes focalisées sur le perceptron multicouche et sur l'algorithme de retro propagation pour la classification des données.

II.1 introduction

Dans le monde occidental, la première cause de mortalité provient des maladies cardiovasculaires.

Même si les connaissances acquises en cardiologie sont grandes, le cœur n'a pas encore dévoilé tous ses secrets. Pourtant les médecins disposent de nombreux moyens pour l'étudier et vérifier son bon fonctionnement. Notamment, ils utilisent l'électrocardiogramme (ECG), qui est une représentation graphique temporelle des différences de potentiels des forces électriques qui conduisent à la contraction musculaire cardiaque.

Ce chapitre contient le principe et les méthodes de classification des signaux cardiaques, la classification des arythmies cardiaques sous un aspect clinique ensuite nous ferons une synthèse des méthodes de classification automatique de l'ECG.

Cette classification des signaux électrocardiographiques réalisée à base des algorithmes neuronaux donne des résultats supérieurs à tous les classificateurs classiques. L'ECG est un signal non linéaire et les réseaux de neurones ont les propriétés d'apprentissage et de généralisation, offrant la possibilité d'utilisation de modèle non linéaire.

On peut citer les taches principales des systèmes électrocardiographiques automatisés :

- Acquisition des données, traitement préalable et conditionnement du signal.
- Stockage des données pour une utilisation ultérieure.
- Calcul des paramètres électriques du signal cardiaque.
- Recherche de l'interprétation.
- Reconnaissance de la forme.
- Classification et diagnostic en termes médicaux [8].

Et à la fin nous terminons par les différents types d'implémentation des réseaux de neurone.

II.2 classification

II.2.1 définition

La classification consiste à regrouper des observations ayant des caractéristiques communes.

Les groupes formés sont appelés classes ou labels. L'objectif d'un processus de classification est d'assigner une classe à une nouvelle observation inconnue. Le nombre de classes est dépendant de l'application.

Lorsqu'il n'y a que deux classes, la classification est dite binaire et peut s'apparenter à de la détection.

II.2.2 Principe

De manière générale, la classification représentée par un vecteur sélectionne une classe.

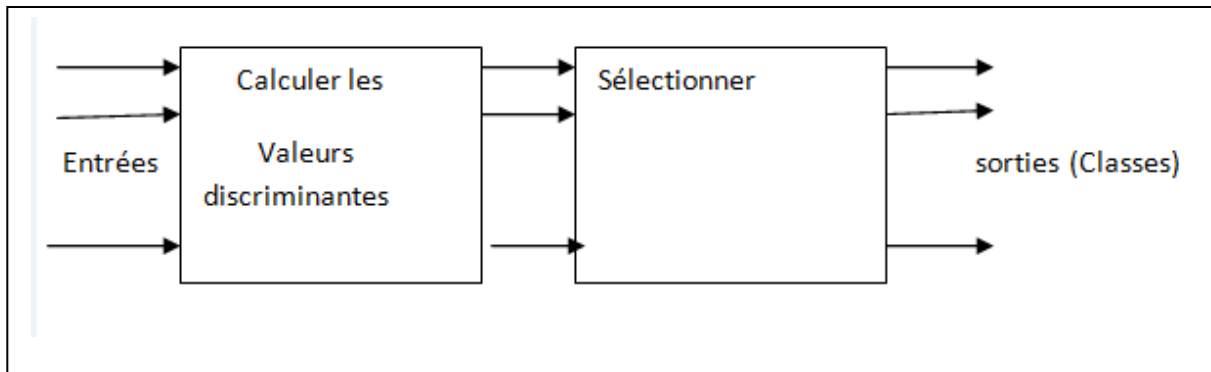


Figure II.1: classificateur

Cette formule peut exprimer le classifieur recherché, Soit

$$x \longrightarrow c(x) = 1, 2, 3, \dots, k, \quad k \text{ classe possible [8].}$$

C: est une classe à rechercher

X : est un signal d'entrée

C(x) la classe associée à un signal d'entrée

Le principe détaillé de la classification se fait en 2 phases :

- **apprentissage** : s'effectue à base d'entraînement du système à partir de données initiales.

Le système va extraire des paramètres lors de l'observation, ces signaux temporels sont issus de l'acquisition par le capteur. Cette étape dépend de l'application, on peut avoir une influence sur les performances dans le cas des signaux biologiques, donc pas possible de déterminer la consommation énergétique dans l'extraction des paramètres, pour cela il est nécessaire de fixer le cadre spécifique sur le type du signal considéré ainsi que l'application de classification,

Après extraction des paramètres, le modèle est configuré de classificateur.

L'étude peut se faire selon 2 catégories :

Non supervisé : l'algorithme ne connaît pas la classe, ni leur nombre, le but de cet algorithme est de trouver des similitudes entre différentes observations pour les regrouper en classes.

Supervisé : chaque observation d'entraînement est marqué avec un label, l'algorithme ajuste le modèle afin que la décision de classe de chaque observation correspond à celle donnée en entrée, cela permet de modifier les paramètres pour obtenir une bonne sortie.

- **prédiction** : permet d'extraire un vecteur de paramètres de nouvelles observations et détermine la classe d'appartenance.

On peut appeler cette étape généralisation car le modèle a une capacité de classer des observations [9].

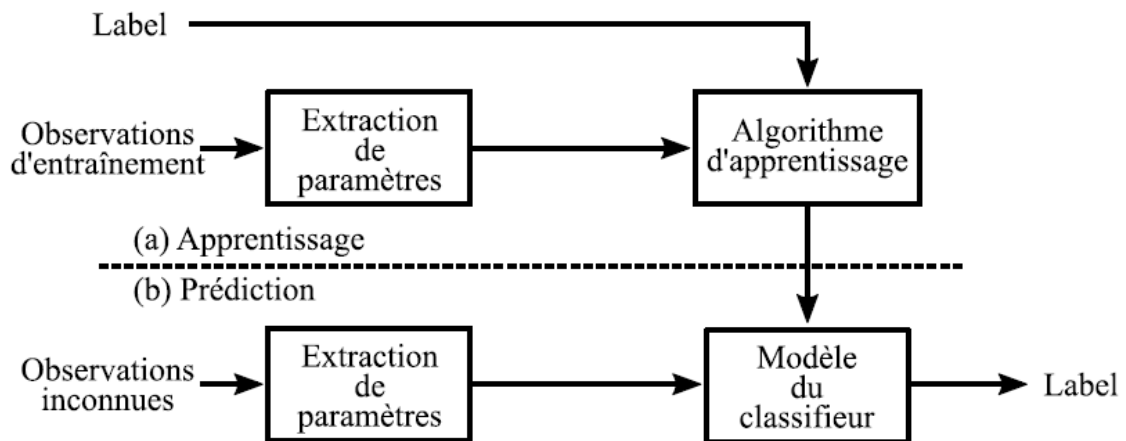


Figure II.2 : Schéma de principe de la classification supervisée. L'apprentissage (a) permet de fixer les paramètres du modèle grâce aux observations d'entraînement. La phase de prédiction (b) utilise le modèle pour déterminer la classe ou le label.

II.2.2.1 Le classificateur traditionnel

Composé en deux blocs, le premier calcule la fonction discriminante (probabiliste) pour chaque classe et est donné par :

$$f_i(x) = \text{prob}(\text{classe } i/x)$$

X: vecteur d'entrée

L'indice i: la classe correspondante

Le deuxième bloc sélectionne le max de ces fonctions $f_i(x)$

$$\text{Class}(x) = i / f_i(x) = \max \{f_i(x)\}$$

La fonction probabiliste $f_i(x)$ est l'ensemble de formes pour lesquelles la classe appartenance est connue, cet ensemble est appelé la base d'apprentissage. A partir de là, on va établir une configuration du classificateur.

La probabilité de bon reconnaissance des entrées est appelée le taux d'apprentissage. On distingue deux techniques de discrimination.

Non paramétrique : la probabilité du vecteur x est définie par le comptage des éléments de la base d'apprentissage.

Paramétrique : l'estimation des paramètres (moyenne) à partir de la base d'apprentissage et où l'on cherche la classe ayant la plus grande probabilité.

Le classificateur traditionnel est basé sur les modèles statistiques, et les paramètres estimés de la base d'apprentissage [8].

II.2.2.2 Le classificateur neuronal

Les réseaux de neurones artificiels ANNs (Artificial Neural Networks) sont apparus comme des outils intéressants dans divers domaines industriels et de recherche. Ils sont capables de résoudre les problèmes non linéaires et complexes comme la reconnaissance des formes, la classification ou l'optimisation [10].

La classification est composée en deux blocs : le premier bloc calcule les fonctions de discrimination.

$$x \longrightarrow f_k(x) \quad f_k(x) \text{ un vecteur de } k \text{ composants}$$

et le deuxième bloc sélectionne le maximum. Dans le cas du classificateur neuronal basé sur l'apprentissage il permet de modifier les poids synaptiques pour approcher au mieux la probabilité $f_k(x)$, à base d'échantillonnage.

Donc on cherche à minimiser l'énergie définie par la distance moyenne entre $f_k(x)$ et la valeur désirée $d(x)$, après la classification on obtient la sortie correspondante à la classe de haute priorité, les autres classes ou les sorties sont à l'état bas.

Dans notre travail, nous avons utilisé pour la classification des signaux cardiaques. Le perceptron multicouche MLP (Multi Layer Perceptron) est un modèle non linéaire. La couche d'entrée est la première couche dont le nombre de neurones égal au nombre de caractéristiques spécifiques sélectionnées. La couche de sortie est la dernière couche qui détermine la sortie, le nombre de neurones dans cette couche dépend du nombre de classes désirées. Les couches intermédiaires sont ajoutées afin de renforcer la capacité d'apprentissage du réseau [10].

II.3. La classification neuronale des données médicales

II.3.1. classification des signaux cardiaques

Dans cette section, nous présentons l'aspect physiologique des sons cardiaques normaux et les méthodes de classification.

II.3.1.1. Anatomie de cœur

Le cœur est au centre du système circulatoire, c'est un muscle dont le travail consiste essentiellement à pomper le sang dans tout le système circulatoire.

Le cœur se compose de deux systèmes de pompage distincts, le côté droit et le côté gauche. Le côté droit reçoit du sang pauvre en oxygène provenant des veines et le pompe vers les poumons, où il capte

l'oxygène et se débarrasse du dioxyde de carbone. Le côté gauche du cœur reçoit du sang riche en oxygène des poumons et le pompe à travers les artères vers le reste du corps. Le cœur a quatre chambres séparées qui pompent le sang, deux du côté droit et deux du côté gauche : oreillette droite, ventricule droit, oreillette gauche et ventricule gauche [8].

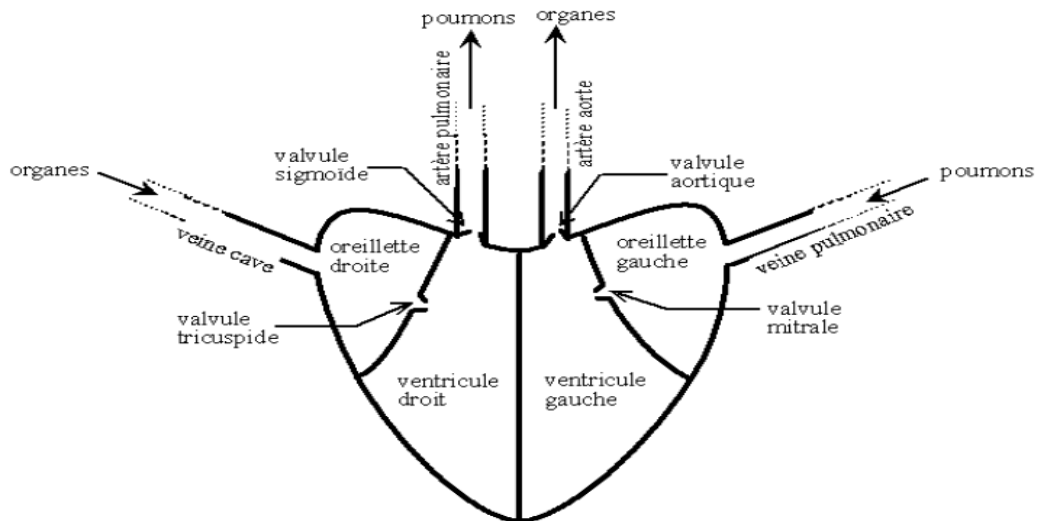


Figure II.3: anatomie du cœur

II.3.1.2. Le cycle cardiaque

Un cycle complet de battements du cœur comprend deux phases : la systole ou contraction et la diastole ou décontraction. Le sang appauvri en oxygène entre dans l'oreillette droite par deux grandes veines, en se contractant, l'oreillette l'envoie dans le ventricule, les valvules s'ouvrent, le ventricule se contracte à son tour expulse le sang par les artères pulmonaires vers les poumons, là il est rechargé en oxygène. Le sang enrichi revient dans l'oreillette gauche puis dans le ventricule gauche, la contraction du ventricule envoie le sang dans tout l'organisme par les artères Figure II.4, la contraction-décontraction des deux parties du cœur se déroule simultanément environ 70 fois par minute.

C'est en se refermant que les valvules [auriculo-ventriculaires]: tricuspide, mitrale et sigmoïdes (aortique et pulmonaire) émettent les bruits des battements.

La durée de tout le cycle cardiaque est d'environ 0.8s. Le cœur se relâche à peu près (0.4s). Un tel repos dans les intervalles entre les contractions est suffisant pour que la capacité de travail du muscle cardiaque se rétablisse tout le temps [11].

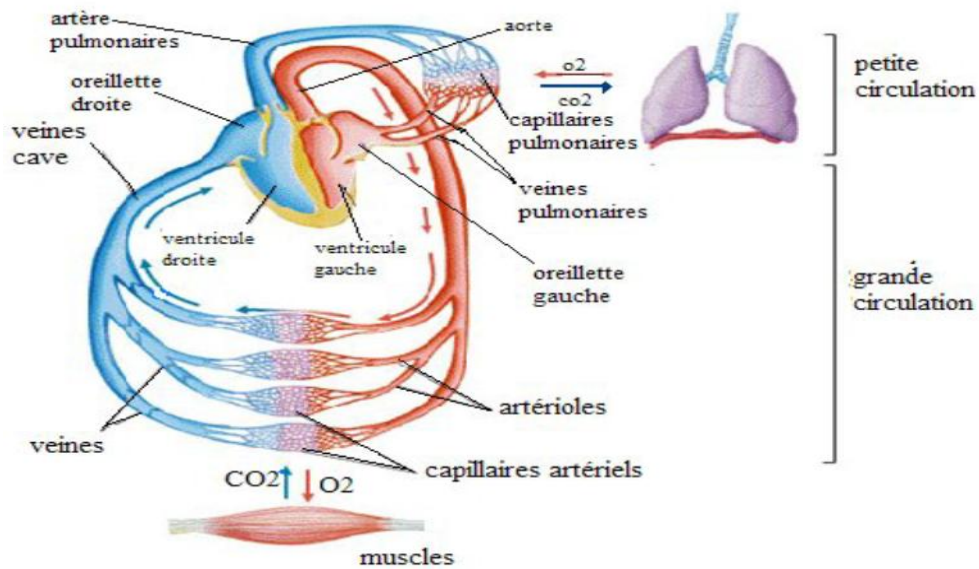


Figure II.4: Cycle cardiaque

II.3.1.3 L'électrocardiographie de la surface

L'activité électrique du cœur peut être recueillie par des électrodes placées à des endroits spécifiques sur la peau.

L'enregistrement de cette activité est appelé ECG ou l'électrocardiogramme, c'est une représentation graphique de l'activité électrique au niveau de cœur et un outil de diagnostic efficace pour détecter les anomalies des fonctionnements cardiaques [12][11].

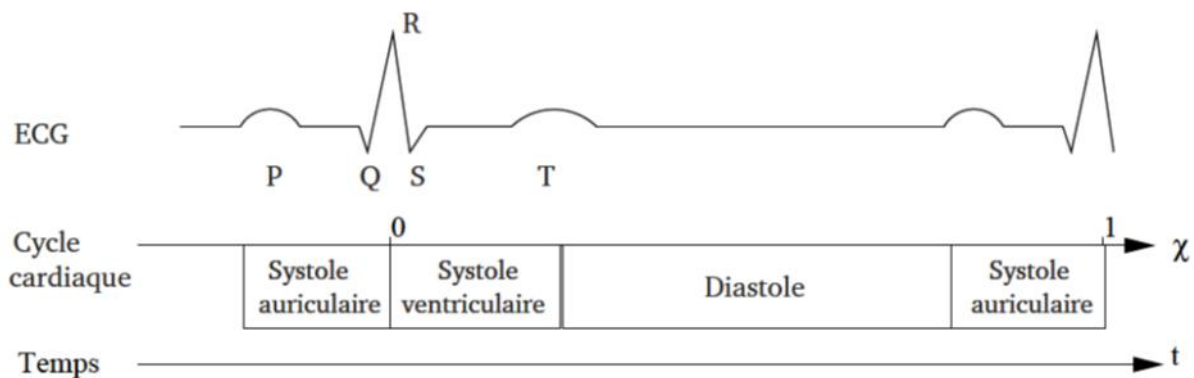


Figure II.5: les phases d'un battement de cœur, tracées dans l'ECG.

II.3.1.4 Classification et interprétation de l'ECG

L'analyse de l'ECG doit se faire dans le même ordre que le cycle cardiaque et se déroule comme suit :

-**L'onde P**: Elle est liée à la dépolarisation auriculaire, dans la condition normale hauteur à 2mm et durée à 0.12 secondes, et il n'existe qu'une seule avant le complexe QRS [14][13].

-**Le complexe QRS** : Elle correspond à l'activation et la dépolarisation ventricule, dans le cas normal le complexe QRS à une durée comprise entre 0.08- 0.095 secondes [15][14].

-**L'onde T**: Elle ne doit pas dépasser la moitié du complexe QRS, dans la condition anormale elle pourrait être grande, trop petite ou inversée [13].

-**L'onde U**: Cette onde n'est pas toujours visible, si elle peut être observée, elle se caractérise par une petite taille, le cas anormale serait d'avoir une onde U trop ample [13].

-**L'intervalle PP**: L'intervalle PP permet de mesurer la fréquence auriculaire. Si le rythme cardiaque est régulier l'intervalle PP est identique à l'intervalle RR [8].

-**L'intervalle PR**: Il correspond au temps de conduction auriculo-ventriculaire compris entre 0.12 et 0.2 secondes. Si l'espace PR est court ou lent alors il y'a une anomalie cardiaque [13].

-**L'intervalle RR**: L'intervalle RR permet de mesurer la fréquence ventriculaire ou fréquence cardiaque du cœur [8].

II.3.1.5. Arythmies cardiaques

L'arythmie représente l'absence de régularité des battements cardiaques, elle est souvent associée à un trouble de la conduction électrique.

Les troubles du rythme cardiaque sont dus essentiellement à des défaillances de certaines zones du cœur. Ces défaillances se manifestent par une déformation d'ECG qui est un moyen classique de dépistage de différentes arythmies [8].

La méthodologie suivi par les médecins est basé sur l'observation, l'expérience et la mesure des données.

- Localisation des ondes P, T, U, les complexes QRS.
- La morphologie (forme) des différentes ondes et complexes localisés.
- Calculer les amplitudes et les caractéristiques des différentes ondes et complexes.

Dans la terminologie médicale, un signal est dit normal si tous les paramètres cités ci-dessus sont normaux.

Dans le cas des troubles du rythme cardiaque et selon leur origine on définit les classes suivantes :

- ✓ Les troubles du rythme sinusal.
- ✓ Les troubles du rythme auriculaire.
- ✓ Les troubles du rythme ventriculaire.

a. Les troubles du rythme sinusal :

Se définissent par une onde P et un complexe QRS tous deux de morphologie normale mais les intervalles PP et / ou RR représentent des valeurs différentes des valeurs du signal normal. Les troubles du rythme sinusal sont la tachycardie sinusale et la bradycardie sinusale [12].



Figure II.6: bradycardie sinusale

b. Les troubles du rythme auriculaire :

Se définissent par l'onde P de morphologie anormale et un complexe de morphologie normale.

En plus de la morphologie anormale de l'onde P, les caractéristiques temporelles des différents intervalles : PP ; PR ; RR permettent de distinguer entre les extrasystoles auriculaires, la tachycardie auriculaire le flutter auriculaire et la fibrillation auriculaire.



Figure II.7: extrasystole auriculaire

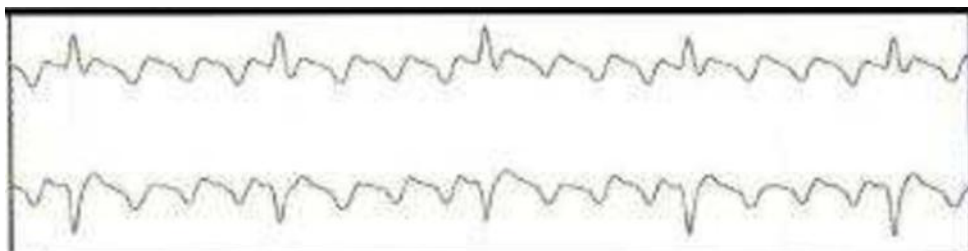


Figure II.8: flutter auriculaire

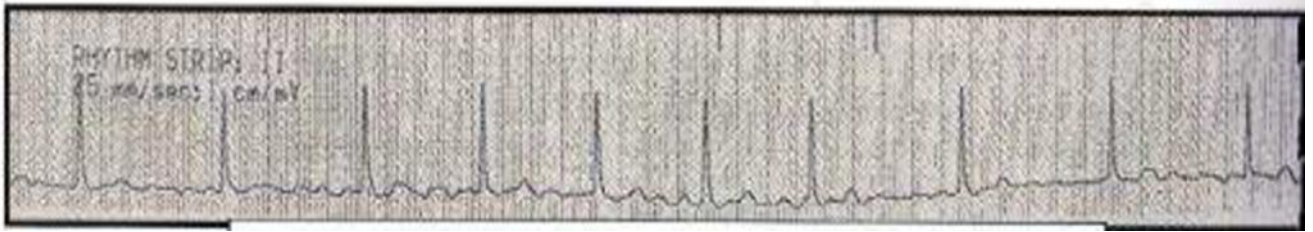


Figure II.9: fibrillation auriculaire

c. Les troubles du rythme ventriculaire :

Sont définis par une onde P de morphologie normale et un complexe QRS de morphologie anormale. De plus les connaissances des caractéristiques temporelles des différents intervalles PP ; PR et RR permettent de différencier entre les différentes anomalies.

Affectant le rythme ventriculaire, la tachycardie ventriculaire, le flutter ventriculaire et la fibrillation ventriculaire [8].



Figure II.10: extrasystole ventriculaire

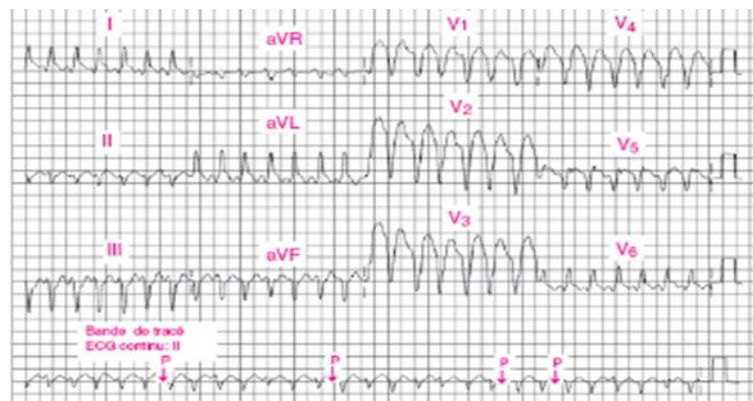


Figure II.11: tachycardie ventriculaire

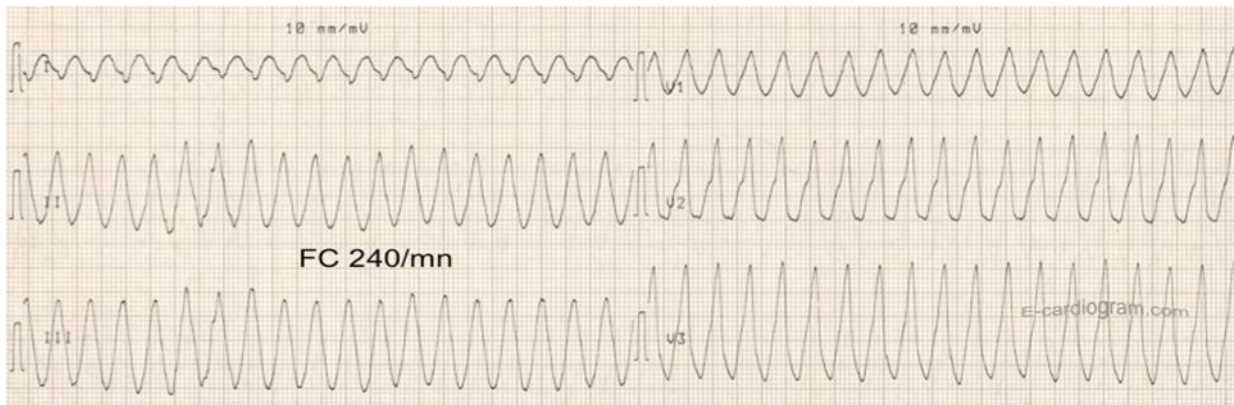


Figure II.12: flutter ventriculaire

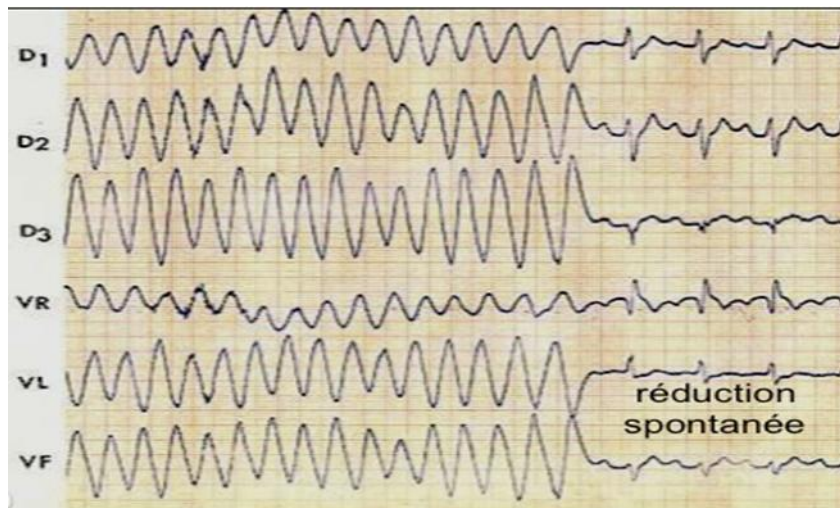


Figure II.13: fibrillation ventriculaire

II.3.2 les méthodes de classification automatique des signaux ECG

La plupart des travaux de recherche sont basés sur les méthodes des arbres et les méthodes statistiques, par la suite ces méthodes ont été développées telle que syntaxique, les systèmes experts, la logique flou, et enfin les réseaux de neurones, Figure II.16 [8].

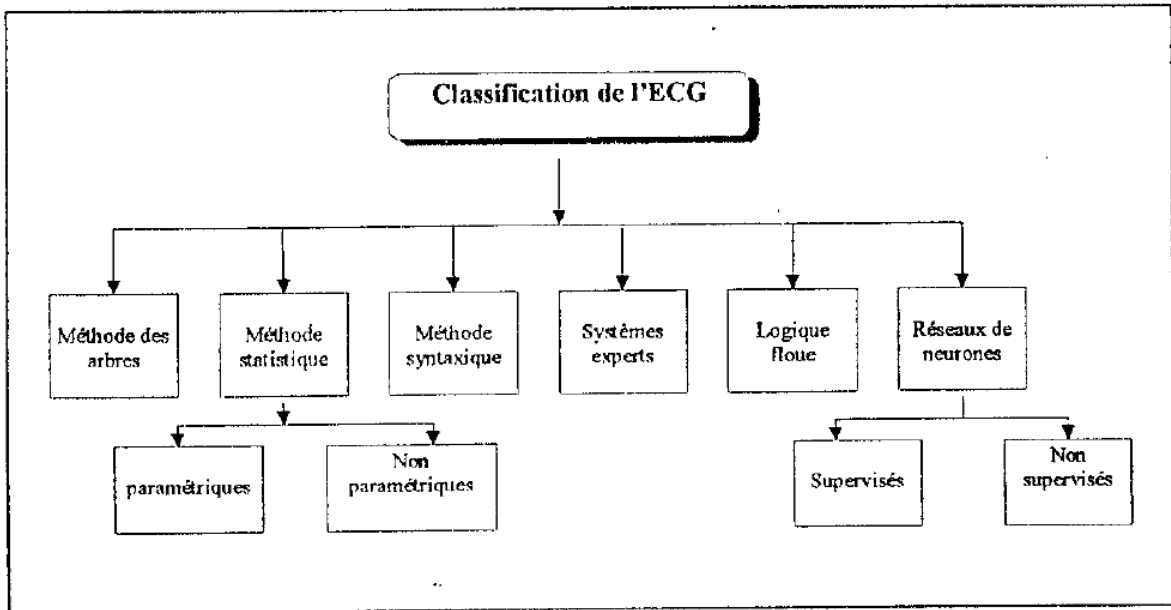


Figure II.14 : les différentes méthodes utilisées pour la classification d'ECG

II.3.2.1 Synthèse des différentes approches de classification ECG

A. Méthode des arbres

Cette méthode basée sur l'utilisation des arbres de décision, Les arbres de décision sont une famille de classifieurs qui prédit une classe par descente d'un arbre. Un arbre est composé de nœuds et de feuilles. Chaque nœud compare la valeur d'un paramètre d'entrée avec un ou deux seuils déterminés lors de l'apprentissage. Chaque nœud possède deux sorties qui indiquent le nœud suivant en fonction du résultat de la comparaison. Les nœuds finaux sont appelés feuilles et sont associés à une classe. La prise de décision d'un arbre est réalisée en descendant, ce dernier jusqu'à atteindre une feuille. La profondeur d'un arbre est définie comme le nombre maximum des nœuds à traverser avant d'arriver sur une feuille.

Les entrées de ces programmes sont caractérisées par des amplitudes des différentes ondes permettant de distinguer la nature du signal normal ou pathologique définie par des classes dans le système, ces programmes sont flexibles à toute modification [9][16].

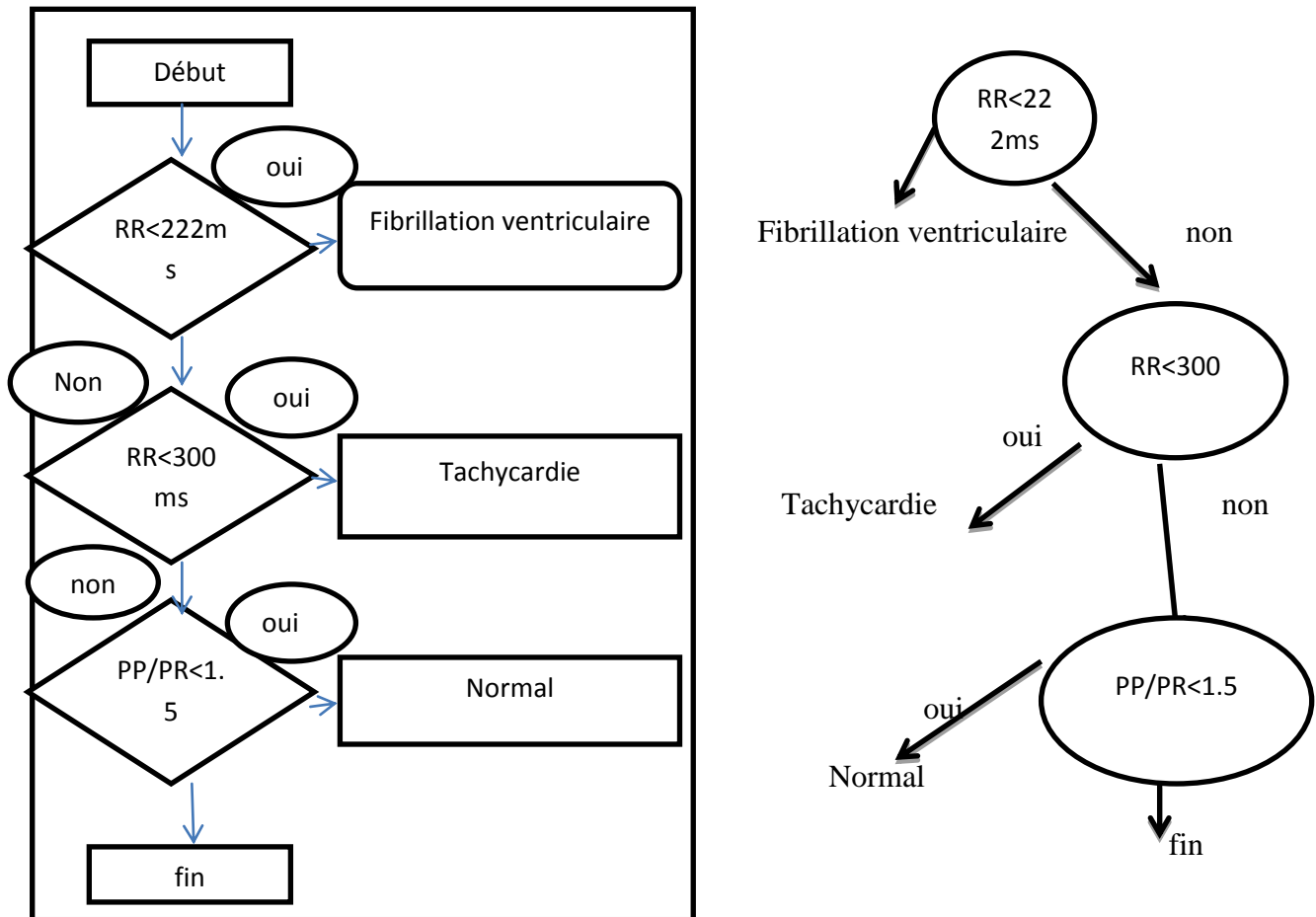


Figure II.15: méthode arbre

L'utilisation d'un unique arbre comme classifieur peut conduire à des problèmes comme :

- Le Sur-apprentissage (overfitting) ou la création d'arbres trop profonds. Le sur-apprentissage apparaît lorsque le classifieur s'appuie trop sur les données d'entraînement lors de l'apprentissage. Cela signifie que le classifieur, en plus de mémoriser les spécificités des classes, mémorise le bruit qui est spécifique aux données d'entraînement. Cela se traduit par une chute des performances de classification lors du test du classifieur. Pour limiter cet effet, il est possible d'utiliser des méthodes d'ensemble dont le but est de créer plusieurs arbres. Chaque arbre est entraîné avec un sous-ensemble des données d'entraînement afin de répartir les effets du sur-apprentissage sur les différents arbres [9].
- Les différents critères de diagnostic des médecins donnent une naissance à des systèmes commerciaux où l'utilisateur pouvait faire entrer ses propres préférences, systèmes non précis [16].

B. L'approche statistique

Cette approche s'appuie sur des calculs de probabilités et font des suppositions sur la distribution des données de manière à prédire la classe d'une nouvelle entrée.

Cette méthode utilise le traitement mathématique pour extraire les paramètres du signal cardiaque, en utilisant la règle de Bayes dans le processus de classification.

Cette règle permet de reconnaître si le patient qui a les symptômes S est atteint de la maladie D_i , parmi k maladies de l'ensemble de D_k , on peut exprimer cette règle par :

$$prob(D_i/s) = \frac{prob(s/D_i)prob(d_i)}{\sum_k prob(s/D_k)prob(D_k)}$$

D_k : les classes séparables.

Les travaux de Piblerger montrent la performance de son système amélioré de 20% à 40% par rapport à la méthode arbre.

L'utilisation de cette approche comme classifieur peut conduire à des problèmes comme :

- Nécessite une base de données large pour tester la procédure de classification, les paramètres statistiques qui sont préalablement estimés par l'apprentissage et l'âge du patient, le sexe, la taille, le prénom.
- Problème de séparation des classes lors de la présence de nombreuses anomalies [16].

C. Approche syntaxique

L'approche syntaxique, utilise des paramètres descriptifs liés à la nature même des formes du signal ECG. Elle suppose que l'ECG est composé par la structure de formes élémentaires ou primitives qui sont les pics et les segments. Ainsi les pics sont combinés pour former des complexes et les pics et les segments combinés pour former le cycle cardiaque.

Par la suite à chacune de ces primitives est attribué un symbole de l'alphabet $\sum \{K+ ; k- ; E ; \pi\}$ ou

$K+$ le pic positif, $k-$ le pic négative ; E une ligne droite et π un contour.

Dans une étude d'évaluation, Y.Susuki reporta que la sélection des primitives n'est pas une solution générale [8].

D. Les systèmes experts et la logique floue

D'après, Jos.L Willems l'utilisation clinique de ces approches a été limitée. Le problème de ces méthodes nécessite une reconnaissance des règles [8].

E. Réseaux de neurones

Aucune des méthodes citées précédemment n'a pu résoudre le problème de la classification du signal cardiaque. Ce sont des modèles utilisés pour le traitement des modèles linéaires.

Le réseau est composé des ensembles de couches. La couche entrée contient des éléments dans le vecteur des paramètres de signal suivi par la couche cachée qui effectue la somme de ses entrées et passe par la fonction d'activation afin d'obtenir un classifieur non linéaire et ensuite connecté à la couche cachée ou de sortie à des fonctions d'activation sigmoïde ou linéaire. Le nombre des neurones de couches cachées peut être égale au nombre des classes [8] [9].

Durant ces dernières années, les réseaux de neurones ont été utilisés pour la classification du signal cardiaque, leurs avantages sont :

- Habilité d'apprentissage et d'auto-organisation à partir d'exemples.
- Généralisation.
- Non nécessité de la connaissance de règles.
- Possibilité d'utiliser des modèles non linéaires [8].

a) Classificateur neuronal supervisé

Un système ICD « implantable cardioverter de fibrillation » est proposé par P.Leong il est un classificateur hybride comme montré dans la figure II.16, composé de deux classificateurs. Un classificateur temporel : basé sur la méthode des arbres, et un classificateur neuronal de morphologie basé sur l'algorithme de la rétro propagation du gradient.

La plupart des ICDs font la classification par la méthode des arbres en se basant sur la durée des différents intervalles RR ; PR ; PP.

Le système ICD propose pour classer les arythmies suivantes :

- la fibrillation ventriculaire
- la tachycardie ventriculaire
- la tachycardie supraventriculaire
- Le rythme sinusal normal.

Le classificateur neuronal de morphologie été conçu pour détecter la variation de morphologie dans la tachycardie ventriculaire, il est intégré sur un chip et utilise une implémentation analogique. Le classificateur temporel détecte les autres arythmies, et est implémenté en software [8].

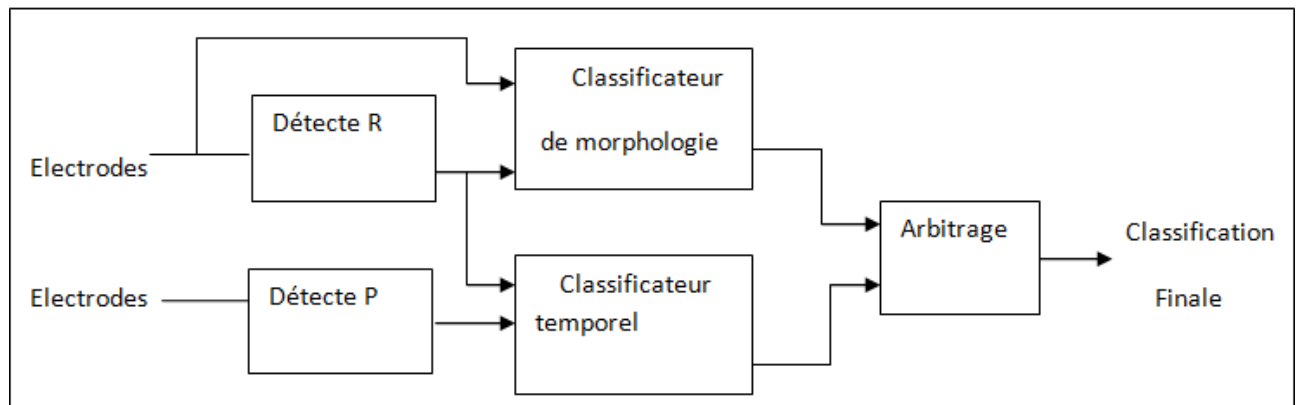


Figure II.16: classificateur neuronal supervisé

b) Classificateur neuronal non supervisé

Par exemple, un système a été proposé par P.voukydis pour la classification des arythmies cardiaques, la figure II.17 montre le système proposé. Il est composé de 2 sous classificateurs neuronales. Le premier classificateur est un réseau de neurones à treize entrées et trois sorties. Les entrées des réseaux décrivent l'amplitude et la durée du complexe QRS, les sorties du réseau permettent de distinguer entre un signal de morphologie normale ou anormale. La sortie du premier classificateur en combinaison avec la variation de l'intervalle RR appliquée à l'entrée du second classificateur pour identifier les arythmies suivantes :

- Rythme sinusal normal
- Tachycardie sinusal
- Fibrillation auriculaire lente
- Fibrillation auriculaire rapide
- Complexes tachycardies étroits
- Complexes tachycardies larges
- Rythme rapide-étroit
- Fibrillation ventriculaire

Les deux classificateurs sont basés sur l'apprentissage non supervisé en utilisant l'algorithme de Kohonen [8].

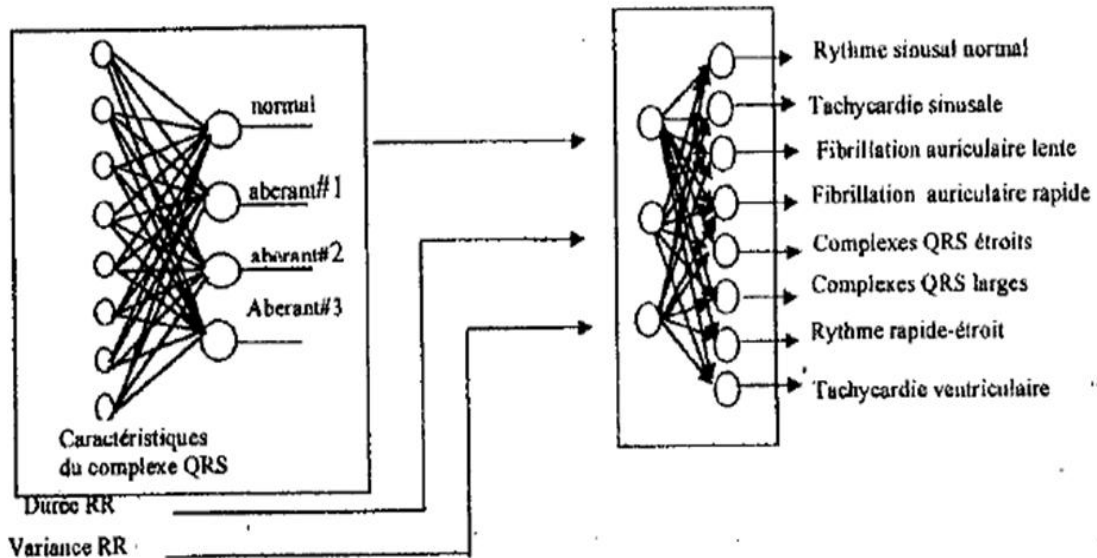


Figure II.17: classificateur neuronal non-supervisée

II.3.3 les différents types d'implémentation

Dans la section précédente, de nombreux travaux sur la classification neuronale ont été présentés et classés selon deux scénarios, logicielle et matérielle, afin d'aider l'expert dans son diagnostic médical. Ces approches ont été proposées selon le mode de traitement de l'information, soit d'une façon séquentielle ou parallèle.

II.3.3.1 Implémentation software

La plupart des classifieurs neuronaux sont développés sur des ordinateurs classiques, des DSP (Digital Signal Processor), des microcontrôleurs, et des circuits ASIC, à l'aide de différents langages (Matlab, Langage C, etc.) en vue d'une utilisation logicielle. La simulation de réseaux de neurones sur ces logiciels offrent à l'utilisateur des bibliothèques et un environnement flexible permettant de les adapter à leur applications. ASIC est une architecture spécialisée pour des applications particulières comme l'ASIP « application specific instruction set processor » par exemple codage des vidéos. Et permet Les applications implémentées sur Matlab application des RNN à l'alliage AL-ZN avec l'utilisation de réseaux de neurones comme outil de datamining. L'architecture interne de ces circuits est composée d'opérateurs arithmétiques et logiques, registres, modules dédiés au contrôle et synchronisation des opérations, et éventuellement de mémoires internes afin de stocker les données et les résultats. Ce type de conception est totalement axé sur les performances de classification, et la vitesse d'exécution des instructions, et il ne possède pas de réelle limitation sur l'utilisation mémoire ou encore sur le coût calculatoire des algorithmes.

Dans ces circuits, le programme de contrôle détermine les opérations à exécuter, offrant une grande flexibilité d'application. Le fonctionnement séquentiel est un inconvénient majeur. En effet, le processeur est constitué de telle façon qu'il exécute les différentes instructions de façon séquentielle. Pour résoudre ce problème on propose une implémentation hardware parallèle [16][9][17].

Exemple des outils pour réseaux de neurones :

- Neural network toolbox de Matlab permet de concevoir, mettre en application, visualiser, et simuler les réseaux de neurones avec l'utilisation de l'interface graphique « nstart » et des fonctions prédéfinies
- FANN « fast artificial neural network » est une librairie open-source de réseaux de neurones sur c++ qui permet l'entraînement de réseaux neuronal, le FANNOOL est un logiciel multi plateforme développé en c qui fournit un ensemble d'outils permettant l'utilisation de librairie FANN. Il propose différents algorithmes d'entraînement, plusieurs fonctions d'activations, et les dimensions de réseaux (le nombre de couches) [18].

II.3.3.2 Implémentation hardware

On distingue 2 grandes approches d'implémentation hardware parallèles des réseaux de neurones :

1 les neuro-calculateurs à application générale

Ce genre d'architecture est basé sur l'utilisation de calculateurs hôtes couplés à des cartes accélératrices agissant en co-processeurs ou processeurs de calcul arithmétique spécialisés [8].

Par exemple le clavier SwiftKey Neural Alpha est une application pour android et iphone qui s'appuie sur les réseaux de neurones artificiels pour prédire et corriger les frappes de manière plus intelligente.

Plus le nombre de processeurs augmente plus le délai de communication augmente. Pour résoudre ces problèmes on utilise l'implémentation VLSI (neuro-calculateurs à applications spécifiques).

2 Implémentation VLSI

L'implémentation VLSI est caractérisée par la rapidité de traitement et ensuite la possibilité de réaliser des systèmes portables.

On distingue 02 grandes approches d'implémentation VLSI (neuro-calculateurs à applications spécifiques) des réseaux de neurones.

- Implémentation qui intègre la phase d'apprentissage et la phase de test/généralisation dans un même circuit. Ce genre d'implémentation permet une flexibilité et une adaptabilité du circuit à plusieurs applications
- Implémentation intégrant seulement la phase de généralisation. Dans ce genre de circuits l'application réalisée en software permet de générer les poids synaptiques

L'implémentation VLSI des réseaux de neurones peut être analogique ou digitale. Les deux styles relatifs à un propre domaine d'application. Le choix d'un style se fait par rapport à la vitesse, la précision, la flexibilité, la reprogrammation, le transfert et la mémorisation de données.

L'implémentation analogique est facile à réaliser mais ses circuits sont sensibles aux bruits, à la température, aux variations de la tension d'alimentation, sont moins précis et consomment une grande surface.

Les circuits digitaux sont moins sensibles au bruit, aux variations de la température et à la tension d'alimentation, sont plus précis et permettent d'intégrer des réseaux de grande taille [8].

Parmi les circuits qui intègrent les réseaux de neurones, sont les circuits FPGA (Field Programmable Gate Arrays) dont XILINX ARTEX-7 et qui permettent de réaliser des circuits très flexibles.

II.4 Conclusion

Dans ce chapitre, nous retenons des points essentiels, pour la classification des arythmies cardiaques il faut prendre en considération la morphologie de l'onde P, le complexe QRS et les caractéristiques temporelles du signal cardiaque tel que l'intervalle PP, l'intervalle RR, et l'intervalle PR.

Dans notre travail, nous avons proposé le classificateur neuronal supervisé qui travaille sur le perceptron multicouche basé sur l'algorithme de la rétro propagation du gradient de l'erreur, permettant d'obtenir des plans de séparation de complexité arbitraire.

Sur la base de ces constatations, nous proposons dans le prochain chapitre un classificateur neuronal des arythmies cardiaques.

III.1 Introduction

Ce chapitre commence par le principe de la classification des signaux ECG par les réseaux de neurones. Par la suite, nous verrons en détail comment créer un modèle qui prédit les arythmies de chaque signal cardiaque en utilisant ce type de réseau. Et enfin, nous présentons les résultats obtenus sur la base de données QT Database (qtdb).

Pour cette application que nous traitons, l'étape de caractérisation est nécessaire pour la reconnaissance automatique des signaux cardiaques. Cette étape de caractérisation permet d'extraire les informations utiles du signal ECG et de le rendre facilement exploitable par des algorithmes de traitement automatique de données.

Dans ce travail, la caractérisation des signaux ECG réside dans la détection du complexe QRS de chaque signal cardiaque, afin d'extraire les paramètres représentatifs nécessaires qui permettront par la suite de bien reconnaître les signaux normaux et les signaux pathologiques.

III.2 classification des signaux ECG

L'objectif des systèmes intelligents est de détecter les situations anormales par l'analyse et la comparaison des signaux biomédicaux temporels, pour cela, on a trois étapes principales :

L'acquisition du signal ECG et des données de suivi du patient recueillies, passant par la partie traitement du signal cardiaque, et enfin par la phase de diagnostic pour détecter les pathologies [19][20][21] .

Les modules principaux choisis pour l'analyse des signaux ECG sont :

- Elimination de bruit du signal ECG
- Détection des ondes P, R, T ou le complexe QRS : le but du bloc traitement est la localisation des différentes ondes.
- Sélection ou extraction des paramètres : cette étape nécessite la sélection des paramètres les plus pertinents (la durée, la pente, l'amplitude.....)
- Classification ou reconnaissance : la classification dépend du développement du système intelligent qui permet de traiter les signaux.

III.3 Structure du classificateur neuronal des arythmies cardiaques

Nous proposons un réseau classificateur qui permet de classer les signaux selon les arythmies suivantes :

- Signal Normal Sinus Rhythm (NSRDB)
- Signal supraventriculaire (SV)
- Signal Cu-ventricular tachyaryhmia (CUDB)

Ce réseau est un classifieur neuronal permettant de classer les données en fonction des amplitudes des ondes P,Q,R,S,T des différents types de pathologie choisie.

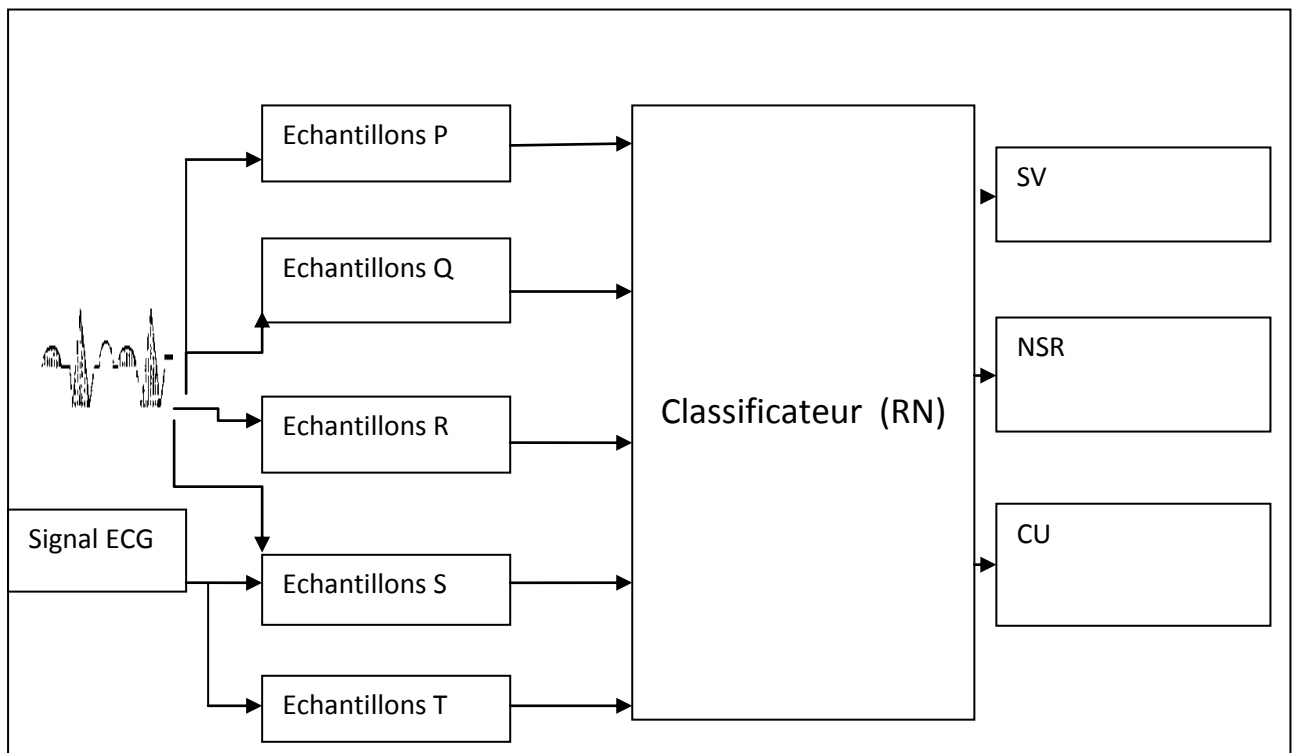


Figure III.1: Architecture du classifieur des arythmies

III.4 Présentation de la base de données

Nous avons téléchargé nos signaux ECG à partir de la base de données <<QT Database (qtdb)>> [29], conçue parmi les meilleures bases universelles, figure III.2.

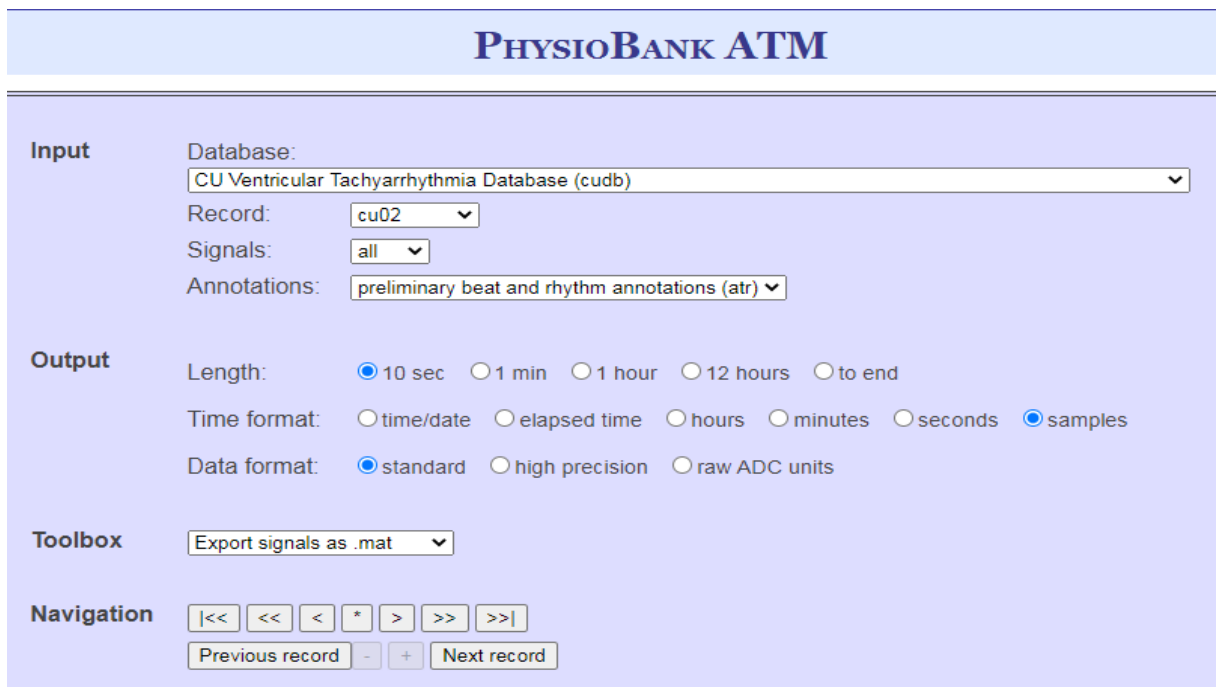


Figure III.2: Base de données

Nos enregistrements sont de 10 secondes échantillonnés à 2500 points par seconde.

III.5 Les anomalies cardiaques

Les anomalies cardiaques prises en considération dans les expériences et citées dans ce travail sont inscrites dans le tableau III.1 :

Référence	Noms d'enregistrement
Base de données de rythme sinusal normal (NSRDB)	16265, 16272, 16273, 16420, 16483, 16773, 16786, 17052, 18177, 16539, 16795, 18184
CU-tachyarythmie ventriculaire (CUDB)	Cu01, Cu03, Cu06, Cu07, Cu10, Cu11, Cu13, Cu14, Cu17, Cu18, Cu22, Cu23, Cu24, Cu28, Cu30, Cu32, Cu33
Arrythmie Supraventriculaire (SVDB)	800, 801, 804, 805, 806, 808, 809, 811, 820, 821, 824, 826, 829, 840, 841, 844, 846, 849, 852, 853, 856, 864, 871, 876, 877, 882, 886, 802, 807, 810, 825, 827, 887

Tableau III.1: la base de données.

III.5.1 MIT-BIH Normal Sinus Rhythm database

Cette base de données comprend 18 enregistrements ECG à long terme de sujets humains. Les sujets inclus dans cette base de données ne présentent aucune arythmie significative; ils comprennent 5 hommes de 26 à 45 ans et 13 femmes de 20 à 50 ans [29].

III.5.2 Cu ventriculaire tachyarrhythmia Database

Cette base de données comprend 35 enregistrements ECG de huit minutes de sujets humains qui ont connu des épisodes de tachycardie ventriculaire soutenue, de flutter ventriculaire et de fibrillation ventriculaire [29].

III.5.3 MIT-BIH Supraventriculaire Arrhythmia Database

Cette base de données comprend 78 enregistrements ECG d'une demi-heure choisis pour compléter les exemples d'arythmies supraventriculaires dans la base de données sur les arythmies du MIT-BIH [29].

III.6 Choix du vecteur d'entrée

Le choix du vecteur d'entrée de notre réseau est très important pour la bonne reconnaissance des pathologies cardiaques. En effet, la qualité de la classification dépend énormément de la pertinence des paramètres du vecteur d'entrée.

Les paramètres de caractérisation choisis sont les mêmes paramètres sur lesquels le cardiologue se base pour établir son diagnostic.

Nous avons déterminé de chaque signal les amplitudes des ondes P, Q, R, S, et T comme des caractéristiques. Ces vecteurs paramètres constituent le vecteur entrée de notre réseau classificateur [30] [31].

III.7 Prétraitement, traitement et implémentation software

Le but de l'implémentation est de déterminer pour le classificateur neuronal les paramètres suivants :

1. La taille de réseau de neurones
2. Les paramètres : le coefficient d'apprentissage et le facteur de momentum
3. La matrice des poids synaptiques.

III.7.1 Prétraitement du signal ECG

L'enregistrement ECG est généralement perturbé par différents bruits, ils peuvent en altérer plus ou moins l'information clinique.

Il est donc important de savoir quels sont les types de bruit dans un signal électrocardiogramme ECG [22].

III.7.2 Types de bruits présents dans le signal ECG

Lors de l'acquisition du signal ECG, des événements indésirables appelés artefacts peuvent apparaître sur le tracé électro cardiographique. La présence de ces bruits peut engendrer des erreurs dans le diagnostic.

Ces bruits peuvent être classés selon leurs origines en deux grandes catégories : bruits d'origine technique et bruits d'origine physique.

III.7.2.1 bruit d'origine technique

Les bruits d'origine technique sont les bruits qui sont causés par le matériel utilisé lors de l'enregistrement [22].

a. Le bruit du réseau 50Hz

Le bruit 50Hz est un bruit qui provient de l'alimentation par le réseau de distribution électrique. Il contamine le signal électro cardiographique ECG avec des oscillations dont l'harmonique fondamentale est à 50 Hz. Généralement, ce bruit est présent dans tous les enregistrements.

b. Les bruits dus au mauvais contact électrode-peau

Lorsque le gel entre l'électrode et la peau se sèche, cela peut provoquer un bruit qui provoque des changements brusques de l'amplitude du signal d'ECG. De plus, de la mauvaise conductivité entre les électrodes et la peau. Ce type de bruit est difficile à éliminer car son énergie se trouve dans la même gamme de fréquence que celle des complexes QRS.

c. Autres bruits

- Mouvements des câbles électriques
- La saturation des instruments de mesure
- Mauvaise qualité du câblage
- Port de vêtements synthétiques.

III.7.2.2 Bruits d'origine physique

Ces bruits engendrés par, soit des activités électriques du corps humain telles que les contractions musculaires, soit par les mouvements lors de la respiration [22].

a) Fluctuations de la ligne de base

La ligne de base est la ligne horizontale prise comme référence pour étudier la forme et l'amplitude des différentes ondes cardiaques.

Les fluctuations de cette ligne de base sont liées principalement aux mouvements du patient pendant sa respiration.

Généralement, Ces perturbations ne sont pas très gênantes pour l'analyse du signal ECG, car ils peuvent être filtrés puisque leur énergie se situe dans les basses fréquences.

b) Bruits dus au signal électromyogramme EMG

Ce bruit est dû à la contraction des tissus musculaires, il va être superposé sur le signal ECG comme des oscillations hautes fréquences.

Ces perturbations sont assez gênantes, elles peuvent noyer les ondes P et T et empêcher parfois la détection des pics R.

c) Autres artefacts d'origine physique

Le signal électro cardiographique ECG peut être affecté par certaines maladies comme l'hyperthyroïdie, l'ischémie et l'hypokaliémie. Ainsi que l'utilisation de certains médicaments peuvent modifier l'allure du tracé ECG, exemple la dioxine et la digitaline.

III.7.3 Traitement du signal ECG

Dans notre travail, on utilise l'outil Matlab pour réaliser la classification des signaux ECG à base des réseaux de neurones. MATLAB est un langage de calcul numérique constitué d'un noyau de routines graphiques et d'algorithmes de calculs préprogrammés.

Pour cette raison, nous avons suivi un organigramme de base pour le traitement des signaux, comme le montre la figure III.3. Nous avons utilisé les méthodes des filtres numériques.

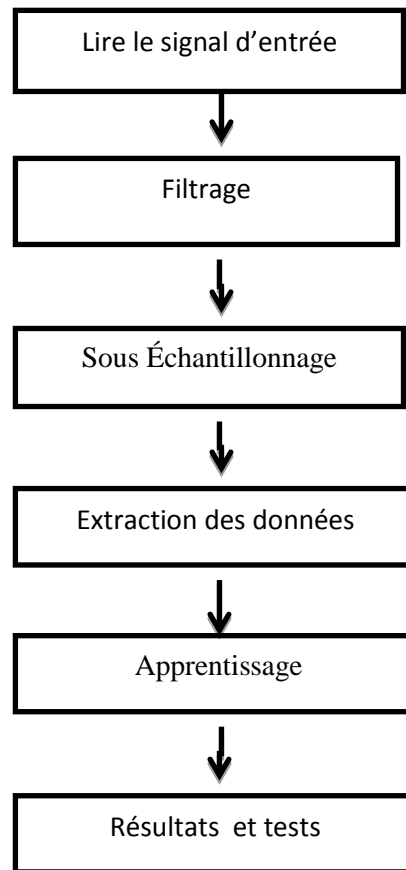


Figure III.3: Traitement du signal ECG

III.7.3.1 Filtrage numérique

A partir de la densité spectrale de puissance d'un signal électrocardiogramme, l'énergie du ECG répartie dans l'intervalle [2 HZ - 40HZ], recouvre des bruits cliniques typiques, situé dans les basses fréquences inférieures à 2HZ et supérieures à 50 Hz.

La solution de ce problème revient donc à filtrer le signal ECG par un filtre passe-bande avec une fréquence de coupure égale au deux fréquences précédentes.

Pour le choix du filtre il y'a des filtres analogiques et différents types de filtres digitaux qui ont été utilisés pour le filtrage du signal ECG tel que les filtres à réponse impulsionnelle finie (RIF) et les filtres à réponse impulsionnelle infinie(RII) [25][26][27][28].

- Le filtrage analogique du signal ECG présentait un inconvénient majeur : il distord la portion terminale du QRS et cache les ondes P et T.
- Les filtres à réponse impulsionnelle finie (RIF) ont une très bonne précision temporelle et un faible décalage de phase. Ils présentent néanmoins des effets de rebond au début et à la fin du

complexe QRS. Ces rebonds rendent impossible la détection du début et de la fin du complexe QRS.

- Les filtres à réponse impulsionnelle infinie (RII) sont simples à mettre en œuvre (exemple : formules Butterworth). Ils sont caractérisés par une bande de fréquence étroite. Leurs réponses impulsionnelles sont théoriquement infinies.

Dans le cadre de notre mémoire, le type de filtre utilisé est à réponse impulsionnelle infinie (RII), type Butterworth.

➤ Filtrage de secteur $f=50\text{Hz}$

Utilise un filtre numérique de butterworth passe bas de fréquence normalisé $f_c=0.5$ pour éliminer les fréquences supérieures à 50HZ.

➤ Filtrage de bruit de la ligne de base $f<1.5\text{HZ}$

Utilise un filtre numérique butterworth passe haut de fréquence normalisé $f_c=0.02$.

III.7.4 sous échantillonnage

Cette étape permet de faciliter la détection de l'amplitude des ondes des signaux traités pour extraire les données dont on aura besoin.

III.7.5 Implémentation software

Pour la classification des signaux ECG, nous avons utilisé le logiciel de calcul Matlab dans sa version 7.8. Afin d'établir notre réseau nous avons besoin d'une banque de données, celle-ci peut être exportée directement à partir d'un fichier Excel.

Après avoir exporté la banque de données, nous avons introduit la commande « nntool » dans la fenêtre de commande, une interface graphique apparaît, elle nous permet de créer un réseau, le visualiser, l'entraîner, le simuler, et exporter les valeurs de sortie. La simulation de notre RNA est effectuée en utilisant l'interface graphique « Neural Network Tool (nntool) » disponible sur Matlab. (Figure III.4).

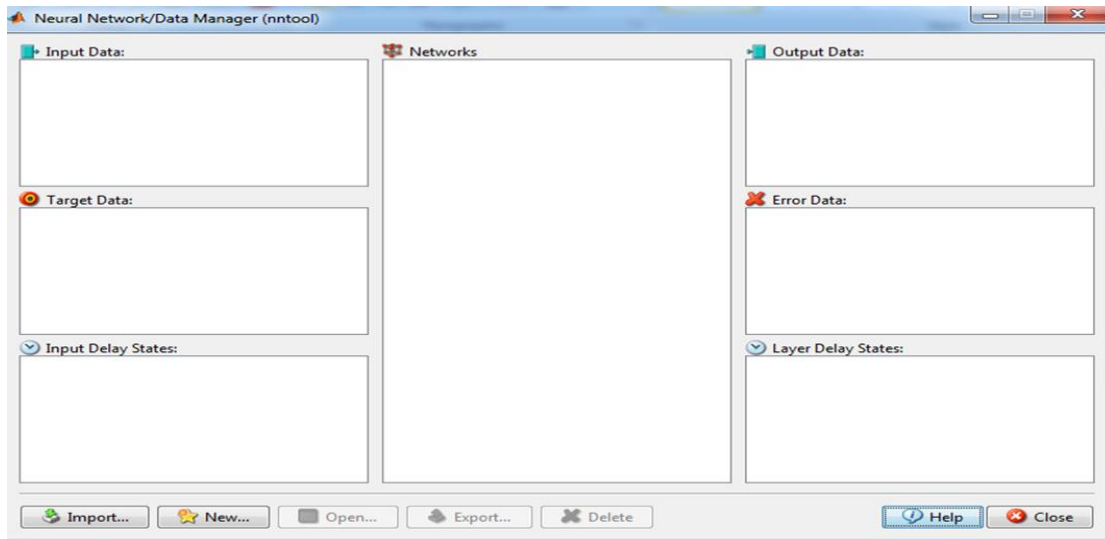


Figure III 4: Interface graphique « nntool »

III.7.5.1 Création des données

Avant de créer un réseau, il faut d'abord introduire les entrées qui sont dans notre cas les amplitudes des ondes de l'activité cardiaque, telle que les ondes P, Q, R, S et T et les propriétés qui doivent être atteinte par le réseau, dans notre cas est Normal Sinus Rhythm , Supraventricular Arrhythmia et Cu ventriculaire tachyarrythmia.

Pour cela on clique sur import (figure III.4), une interface graphique apparaît (figure III.5), elle nous permet d'introduire les entrées (input) et la valeur qui doit être atteinte par le réseau (target).

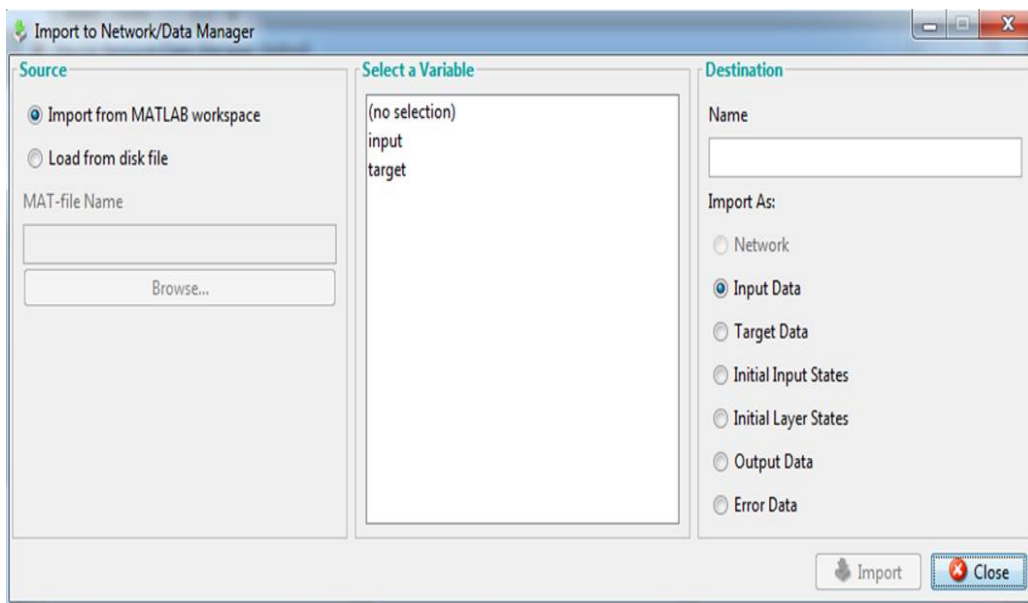


Figure III.5: extraction des données

III.7.5.2 Création du réseau

Pour générer un nouveau réseau, on clique sur New, une nouvelle fenêtre apparaît (figure III.5), on choisit le type de réseau préprogrammé, le feed-forward Backprop ainsi que les propriétés de convergence et les capacités d'approximation.

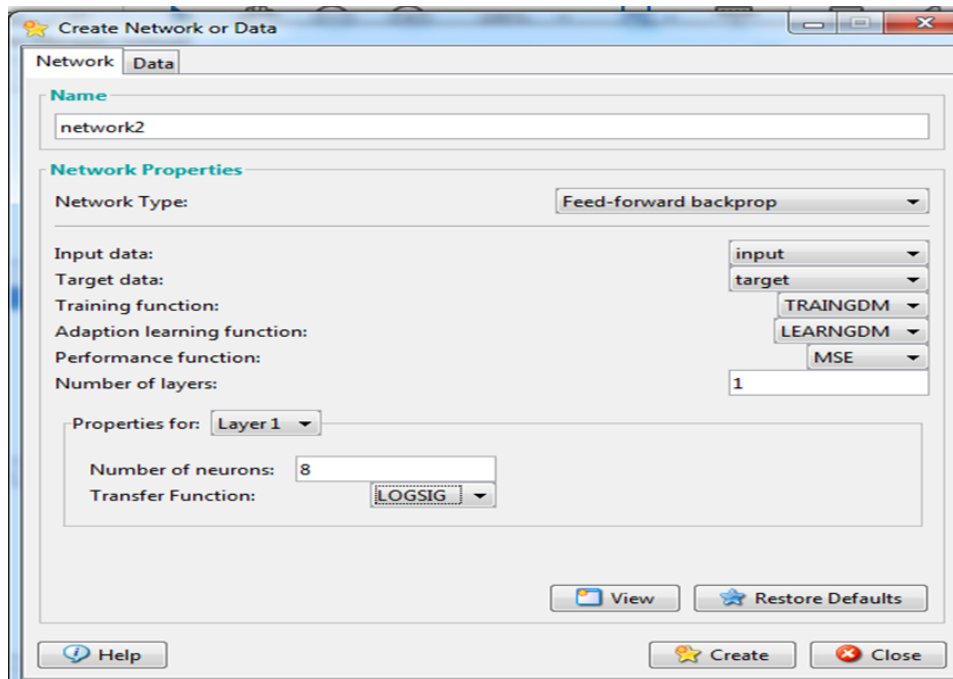


Figure .III.6: Paramètres du RNA dans le cas d'utilisation d'une seule couche cachée à 8 neurones pour la couche

On choisit la fonction d'entraînement «TRAININGDM» «Train gradient descent with momentum backpropagation», et comme fonction d'apprentissage «LEARNINGDM», et la fonction «MSE» «mean square error» comme fonction de performance.

Pour le nombre de couches cachées, il est à une couche et a pour fonction d'activation la tangente sigmoïde «LOGSIG», avec 8 le nombre de neurones.

Après avoir créé le réseau, il est possible de le visualiser en appuyant sur «create». La fenêtre suivante apparaît.

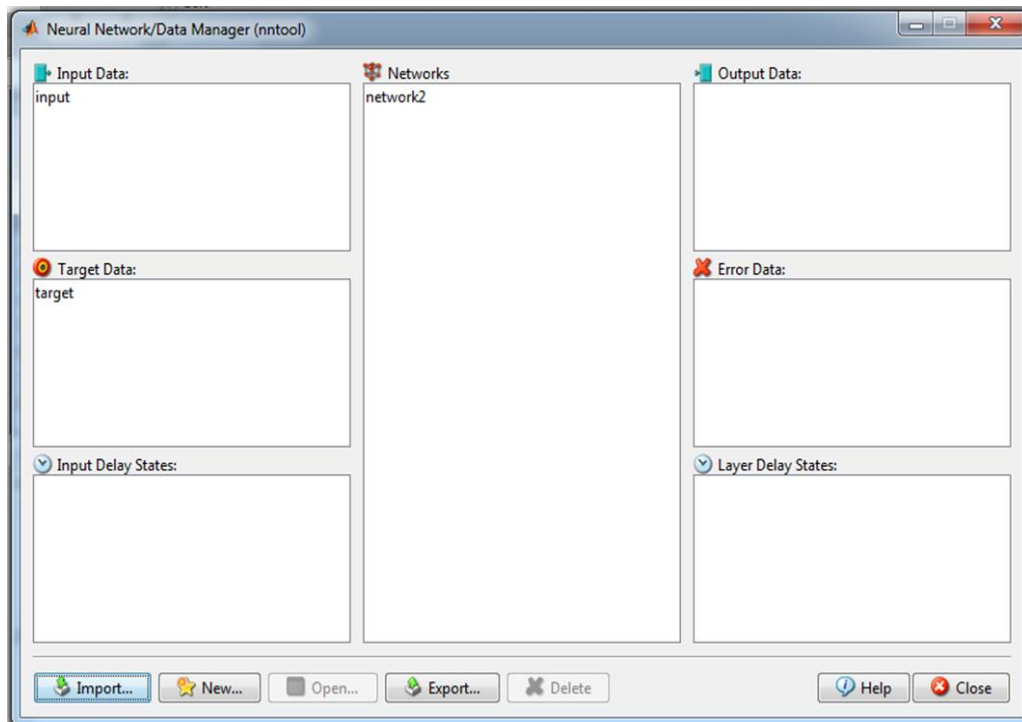


Figure III.7: Interface graphique « nntool », après la création du réseau appelée network2

III.7.5.3 diagramme de réseau de neurones

Le bloc diagramme du réseau de neurones est montré dans la figure III.8.

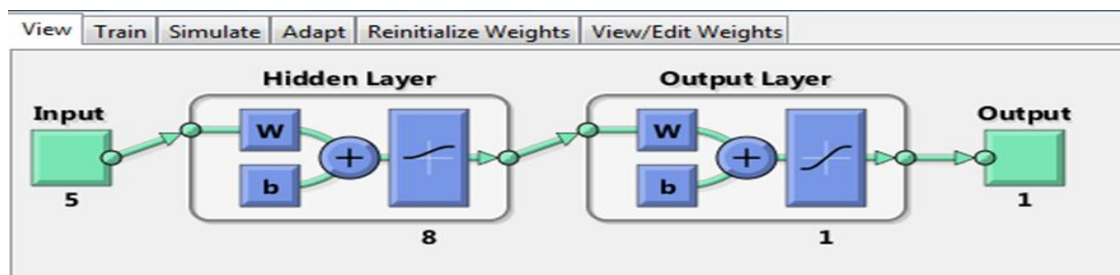


Figure III.8 : bloc diagramme de réseau de neurones

III.7.5.4 apprentissage, simulation et test

L'utilisation de notre système de RNA consiste en plusieurs étapes qui peuvent être résumées comme suit :

a) Apprentissage

Nous avons proposé une sortie désirée pour chaque vecteur d'entrée comme suit :

Entrées	Sorties
SVDB	1
NSRDB	5
CUDB	9

Tableau III.2 : représente les entrées et les sorties de système.

L'apport (input) de formation et la cible (Target) sont donnés comme entrant.

Dans l'onglet des performances d'entraînement présenté dans la figure III.9, l'époque est le nombre maximum d'itérations. On souhaite autoriser la formation, et l'objectif (goal) est défini selon la demande [32].

La performance minimale après le changement des poids est calculée pour de nombreux réseaux avec un nombre différent de neurones, ceux-ci peuvent être modifiés dans la création du réseau.

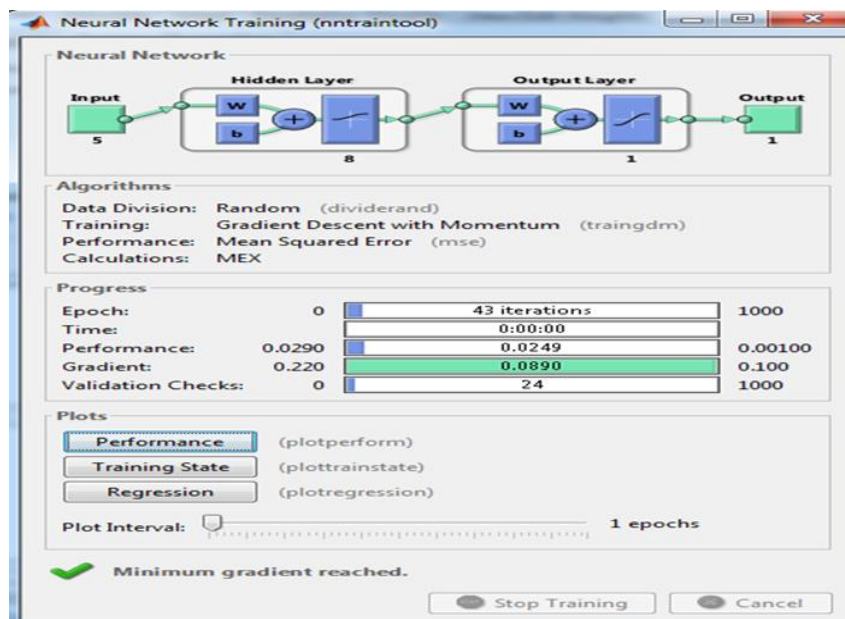


Figure III.9 : Fenêtre de l'apprentissage de l'outil" nntool"

b) Dimensionnement du système

Il n'existe pas de résultats théoriques, ni même de règle empirique satisfaisante, qui permet de dimensionner correctement un réseau de neurones en fonction du problème à résoudre [8].

Pour déterminer le nombre de neurones de la couche cachée nous avons procédé de la manière suivante :

- Préparer une base d'échantillons pour l'apprentissage.
- Fixer le nombre de neurones en entrées et en sortie
- Désigner un nombre de neurones arbitraires dans la couche cachée
- Fixer une erreur de très faible valeur.
- Fixer le coefficient d'apprentissage I_r et le facteur de momentum m_c à des valeurs aléatoires situées dans l'intervalle $[0\ 1]$.

Après plusieurs simulations, nous avons obtenu la convergence pour les paramètres suivants :

- Couche d'entrée : 5 neurones
- Couche cachée : 8 neurones
- Couche de sortie : 1 neurone
- coefficient d'apprentissage : $I_r=1$
- facteur de momentum : $m_c=0.9$
- nombre d'itérations : 1000
- les poids synaptiques (voir l'annexe A)

c) Performances du réseau

Les ordonnées de chaque courbe de la figure III.10, représentent les sorties du réseau (valeurs calculées) pour les entrées réservées à l'apprentissage, aux entrées réservées à la validation et aux entrées réservées au test. Les valeurs en abscisses représentent les valeurs désirées.

- Les cercles en noir représentant les valeurs désirées.
- Les droites tracées en continue de chaque courbe de la figure III.10, représentent l'approximation faite par le réseau.
- Les droites tracées en pointillée représentent la parfaite approximation.

Lorsque ces deux droites se confondent totalement, ou presque, nous parlons alors, d'une meilleure performance. La plupart des cercles (data) sont situés sur les deux droites [33].

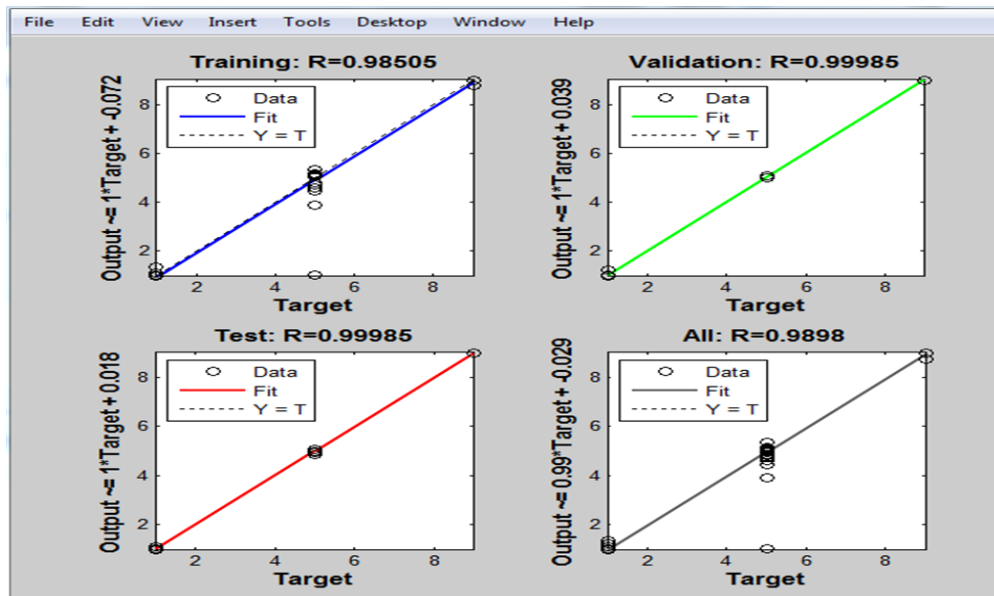


Figure III.10: Courbes de régression

La valeur de "R" représente le rapport entre les sorties du réseau calculées et les sorties désirées. Alors, nous pouvons dire, qu'un meilleur apprentissage donne des valeurs de R très proche de 1.

La courbe de training : donne $R=0,9850$ soit $R= 98,50\%$; les valeurs calculées à la sortie du RNA sont les mêmes que celles désirées des données réservées pour l'apprentissage.

La courbe de validation : donne $R=0,9998$ soit $R=99,98\%$; les valeurs calculées sont les mêmes que celles désirées des données réservées pour la validation.

La courbe de test : donne $R=0,9998$ soit $R=99,98\%$; les valeurs calculées sont les mêmes que celles désirées des données réservées pour le test.

La courbe de All : $R=0,9998$ soit $R=99,98\%$; les valeurs désirées sont les mêmes que celles calculées de la base de données totale.

La figure précédente montre que les valeurs de "R" obtenues sont très voisines de "1", ceci montre que la tâche de l'apprentissage a réussi.

d) résultats du test

Une fois le réseau de neurones créé, Les tests seront effectués afin de vérifier la qualité de classification du modèle neuronal en lui présentant de nouveaux exemples d'entrées, qui ne font pas partie de l'ensemble d'apprentissage afin de classer les entrées par rapport aux classes correspondantes.

Type de signal	Nombre de matrices
SV820	5*9
SV824	5*9
NSR16539	5*13
NSR16273	5*14
CU17	5*9
CU33	5*13

Tableau III. 3 : Types des signaux de test.

Pour cela, nous utiliserons la fenêtre de simulation figure III.11

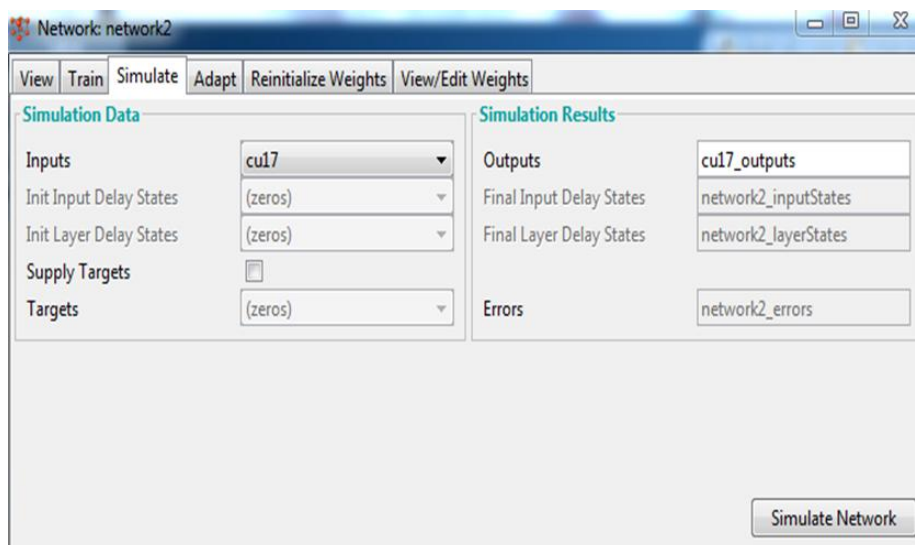


Figure III .11: Fenêtre de simulation

- Les résultats de la classification par le RNA sont donnés dans le tableau III.4.

SV820	[1 1 9 1 1 1 1 1 1]
SV 824	[6.5912 4.7789 4.9989 1 4.8962 4.9272 3.1318 4.8701 4.8964]
NSR16539	[4.118 4.6531 4.7927 4.6344 4.9331 4.2743 4.1535 4.8395 4.2107 4.7446 3.3275 4.9417 1.0001]
NSR16273	[9 8.9999 8.9879 9 8.9998 1.0031 9 8.8818]
CU17	[9 9 9 9 9 9 9 9 8.0619 8.9604 8.9906 7.7004]
CU33	[9 9 9 9 8.9992 9 9 9 9 9 9 9 9]

Tableau III.4: résultats de classification

III.7.5.5 Mesure des performances d'un classificateur

a) définition des performances d'un classificateur

Afin d'évaluer les performances de notre classificateur, on utilise des mesures de performances communes aux systèmes de diagnostic en général et qui sont [8] :

-valeur positive vrai(TP) : définie par le nombre des patients présentant une anomalie identifiée par le système de décision et coïncide avec celle identifiée par le médecin.

-valeur négative vrai(TN) : définie par le nombre de patients ne présentant pas une anomalie et le système de décision affirme le même résultat.

- valeur positive fausse (FP) : définie par le nombre de patients présentant une anomalie et le système n'identifie pas l'anomalie.

-valeur négative fausse(FN) : définie par le nombre de patients ne présentant pas une anomalie et le système identifie une anomalie.

$$\%CC= 100*(TP+TN)/N \quad (III_1)$$

$$\%FP= 100*FP/(TN+FP) \quad (III_2)$$

$$\%FN=100*FN/(TP+FN) \quad (III_3)$$

%CC= pourcentage de classification correcte

%FP= pourcentage de classification d'une anomalie

%FN= pourcentage de classification des signaux normaux

N= nombre de l'ensemble des vecteurs de test

$$SE= 100*TP'/(TP'+FN') \quad (III_4)$$

$$SP= 100*TN'/(TN'+FP') \quad (III_5)$$

$$PR= 100*TP'/(TP'+FP') \quad (III_6)$$

SE= la sensibilité

SP= spécificité « specificity »

PR= précision du système

D'après le tableau III.4, et les définitions précédentes, les mesures de performances sont donnés par :

TP=4

TN=2

FP=1

FN=0

N=7

Donc, les paramètres de performances de notre réseau sont :

%CC=85.71%

%FP=33.33%

%FN=0%

%SE=100%

%SP=66.66%

%PR=80%

III.8 Discussion et conclusion

Dans ce chapitre nous avons proposé une approche pour la classification des Arythmies cardiaques basé sur l'utilisation des amplitudes des ondes P ; Q ; R ; S et T.

Nous avons également présenté la conception software de ce classificateur sous Matlab. Le réseau avec l'algorithme de rétro-propagation a été vérifié avec une bonne fiabilité des performances.

Ce réseau peut être implémenté sur FPGA pour de meilleurs résultats.

IV.1 Introduction

L'objectif final de ce mémoire, est de réaliser un classificateur neuronal du type multicouche (MLP) sur un circuit FPGA. Cette implémentation est effectuée en vue d'une future exploitation dans un système embarqué capable de détecter les arythmies cardiaques.

Ce chapitre concerne l'implémentation sur FPGA du classificateur neuronal des arythmies cardiaques, sur FPGA. Les poids synaptiques déterminés dans la phase d'apprentissage du classificateur seront utilisés pour l'implémentation digitale.

Tout d'abord, nous présentons les circuits FPGA avec leur architecture, caractéristiques et domaines d'application.

Nous présentons, ensuite, les différentes méthodes de leur programmation, le langage VHDL, ainsi que l'outil de conception de XILINX, VIVADO.

IV.2 Les circuits FPGA

IV.2.1 Définitions

Les FPGA (Field Programmable Gate Arrays ou "réseaux logiques programmables") sont des composants entièrement reconfigurables ce qui permet de les reprogrammer à volonté afin d'accélérer notablement certaines phases de calculs. L'avantage de ce genre de circuit est sa grande souplesse qui permet de les réutiliser à volonté dans des algorithmes différents en un temps très court. Le progrès de ces technologies permet de faire des composants toujours plus rapides et à plus haute intégration, ce qui permet de programmer des applications importantes. Cette technologie permet d'implanter un grand nombre d'applications et offre une solution d'implantation matérielle à faible coût pour des compagnies de taille modeste pour qui, le coût de développement d'un circuit intégré spécifique implique un trop lourd investissement.

Grace à sa reconfiguration facile et illimitée, un circuit FPGA (Field Programmable Gate Array), est capable d'implémenter des circuits et systèmes complexes, pour différents domaine d'applications [34][35].

IV.2.2 Architecture des FPGA

Les circuits FPGA sont constitués d'une matrice de blocs logiques programmables entourés de blocs d'entrée/sortie programmable. L'ensemble est relié par un réseau d'interconnexions programmable.

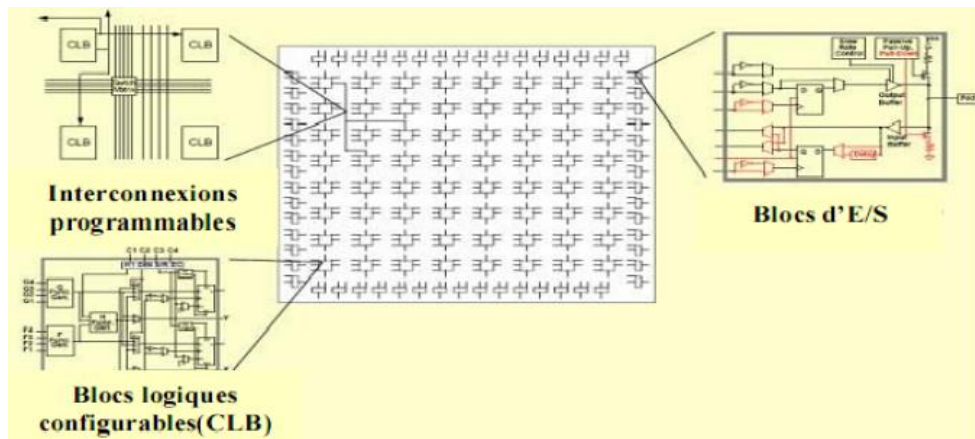


Figure IV.1: architecture interne d'un FPGA

Les FPGA sont bien distincts des autres familles des circuits programmables tout en offrant le plus haut niveau d'intégration logique.

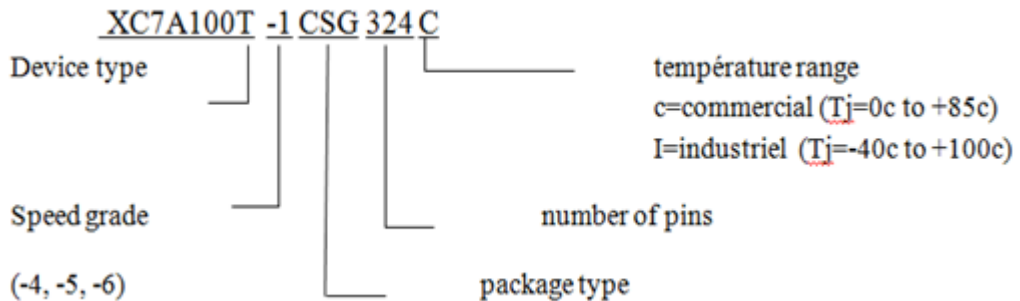
Il existe actuellement plusieurs fabricants de circuits FPGA, XILINX et ALTERA sont les plus connus [34].

IV .2.3 Structure de base

Un circuit intégré classique contient :

- des portes logiques.
- des connexions entre les portes logiques.
- des éléments de mémorisation (registres et/ou mémoires)
- des entrées/sorties.
- une (ou des) horloge(s).

Nomenclature des circuits FPGA



IV.2.4 composition de base d'une carte FPGA

La carte FPGA est composé à la base de :

- un réseau de blocs de logique programmable (Configurable Logic Block – CLB);
- un réseau d'interconnexions programmables
- des blocs d'entrée et de sortie extérieur (Input/Output Block – IOB)

Dans notre étude, la carte FPGA xc7A100T-1CSG324C est caractérisée par 210 IOBs (Input/Output Block – IOB) et 7925 CLBs (Configurable Logic Block – CLB) [35].

IV.2.5 Structure des blocs logiques programmables de notre carte

Un bloc logique programmable est composé de deux tranches (slices). Chaque tranche est reliée au réseau d'interconnexions. Elles sont reliées entre elles verticalement.

Une tranche comprend quatre tables de correspondance (Look-up Table –LUT) à 6 entrées (A6:A1) et deux sorties O6 et O5, une lut pouvant réaliser une fonction logique à 6 entrées ou deux fonctions logiques à 5 entrées. Elle est composée de Trois multiplexeurs (en gris, verticaux) pour réaliser des fonctions logiques de 7 ou 8 entrées, et des multiplexeurs programmables (en blanc) pour router les signaux à l'intérieur de la tranche, est constituée de portes logiques pour l'addition rapide et de huit éléments à mémoire qui sont :

- 4 (Au centre) sont toujours des bascules

Chapitre 4 Implémentation Hardware du classificateur neuronal des arythmies cardiaques

- 4 (À droite) peuvent être des bascules ou loquets.

Il existe aussi des tranches de type M qui peuvent implémenter de la mémoire et des registres à décalage [35].

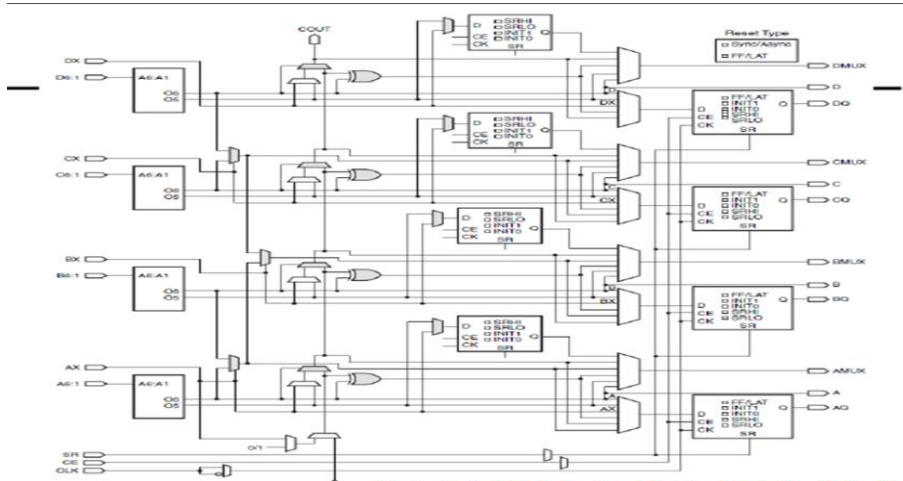


Figure IV.2: FPGA de Xilinx série 7 : tranche de type L (SLICEL)

IV.2.6 Technologie de programmation pour FPGA (ressource d'interconnexions)

Le routage est basé sur une architecture hiérarchique. Cette structure comporte plusieurs niveaux. Les quatre routages principaux sont :

- Routage local : Il permet l'interconnexion entre les LUTs d'un même CLB, entre ces LUTs et les matrices de routage global appelées GRM (Global Routing Matrix), et également, entre les CLBs horizontalement.
- Routage à usage général : Représenté par les canaux de routage horizontaux et verticaux qui relient les lignes et les colonnes de CLBs.
- Routage dédié : Chaque rangée de CLBs dispose de quatre longues lignes horizontales pouvant se relier pour réaliser les bus à trois états. Chaque CLB contient deux lignes verticales dédiées pour la propagation de la retenue.
- Routage global : Permet la distribution des signaux d'horloge ainsi que les signaux alimentant des grosses charges.

Chapitre 4 Implémentation Hardware du classificateur neuronal des arythmies cardiaques

Les interconnexions entre les blocs sont réalisés par la programmation, et il existe deux types d'interconnexion ; reprogrammable et programmable.

Les technologies de programmation qui sont utilisées actuellement sont:

- La technologie SRAM est la plus populaire pour les FPGAs, Une cellule de mémoire SRAM comporte 4 ou 6 transistors. Cette technologie est affectée par les radiations, donc elle n'est pas appropriée pour les applications spatiales.
- Les transistors EEPROM et EPROM
- Les antirusses

La compagnie xilinx offre des circuits FPGA reprogrammables, à l'aide d'un langage de programmation, on peut réaliser un circuit combinatoire ou séquentiel par la création des connexions entre les blocs logique. ce programme est chargée automatiquement dans la mémoire interne de la carte FPGA [35][36][8].

IV.2.7 Programmation des circuits FPGA

Pour programmer un FPGA, il faut commencer par décrire le design en utilisant un langage de description matériel (VHDL, Verilog, ...).

Le synthétiseur va ensuite générer la liste d'interconnexion qui permettra de simuler le placement de tous les composants au sein du FPGA, d'effectuer le routage entre les différentes cellules logiques et de calculer le délai des différents signaux [34].

IV.2.7.1 Langage VHDL

Ce langage a été développé au début des années 80 par le département USA de la défense. Il a été standardisé en 1987 par IEEE. Actuellement, il est le plus utilisé par les développeurs. Afin d'écrire un programme en VHDL, des logiciels, tels que : Xilinx de la société Xilinx ou Quartus de la société Altera sont généralement utilisés. Le code VHDL est très modulaire ce qui permet de réutiliser les fonctions déjà écrites et d'insérer des modules standard. Cependant, il demeure lourd et complexe pour quiconque n'est pas familier au VHDL. On note aussi que la simulation d'un code

Chapitre 4 Implémentation Hardware du classificateur neuronal des arythmies cardiaques

VHDL est trop fastidieuse. A cet effet, Xilinx commercialise un outil de conception de haut niveau nommé « System Generator » (SG).

VHDL peut être utilisé pour plusieurs objectifs : documentation, simulation, synthèse, preuve formelle [36].

IV.2.7.2 Structure générale (concept de base)

Déclaration des bibliothèques dont lequel sont stockés des unités de conception.

Déclaration de l'entité et architecture, les vues externe et interne de paquetage et les configurations, les unités de conception contiennent des descriptions d'action séquentielle concurrentes faisant intervenir des objets appartenant aux trois classes : constantes, variables, et signaux.

IV.2.7.3 Simulation en VHDL

On peut citer deux types de simulation :

a) Simulation fonctionnelle

Elle permet la vérification globale de l'algorithme programmé, dans ce cas les contraintes physiques des composants ne sont pas prises en compte. Les simulateurs de Xilinx ou Modelsim de la compagnie « montor graphic » peuvent être utilisés.

b) Simulation temporelle

Elle consiste à faire une simulation en prenant compte les contraintes de fonctionnement ainsi que les contraintes des composants à utiliser. Après cette vérification le programme peut être implanté sur FPGA et tester.

IV. 2.7.4 Synthèse

La synthèse permet de générer automatiquement à partir du code VHDL un schéma de câblage permettant la programmation du circuit cible.

IV.3 Implémentation sur FPGA du classificateur neuronale

IV.3 .1 Architecture du neurone

A partir des données précédentes, on propose l'architecture pour un neurone, comme le montre la figure IV.3. Ce dernier est composé de :

- Un compteur : son rôle est de compter le nombre d'opérations réalisé par le neurone.
- Un multiplexeur : pour sélectionner chaque entrée avec son propre poids.
- Rom : une mémoire qui stocke les poids synaptiques, elle est propre pour chaque neurone.
- Multiplieur : il sert à faire la multiplication des poids synaptiques avec les valeurs d'entrées. Cette opération est nécessaire pour le calcul de la somme pondérée. Les valeurs calculées dans ce multiplieur sont transmises de sa sortie vers l'entrée de l'accumulateur.
- Registre : sert à stocker le résultat du multiplieur.
- Accumulateur : C'est l'élément qui effectue la somme pondéré. L'accumulateur se compose d'un additionneur et d'un registre.
- Lut : son rôle est de prendre la somme pondérée calculée par le neurone, et l'appliquer à la fonction sigmoïde pour transmettre une valeur d'activation à la sortie du neurone.

Les entrées sont appliqués au neurone via un multiplexeur à huit bits, le compteur sélectionne le poids synaptique approprié à l'entrée du neurone, le résultat du produit est stocké dans un registre, ensuite dans un accumulateur qui fait l'addition et stocke le résultat, cette somme pondérée adressée à une position dans la lut permet d'activer une valeur en sortie.

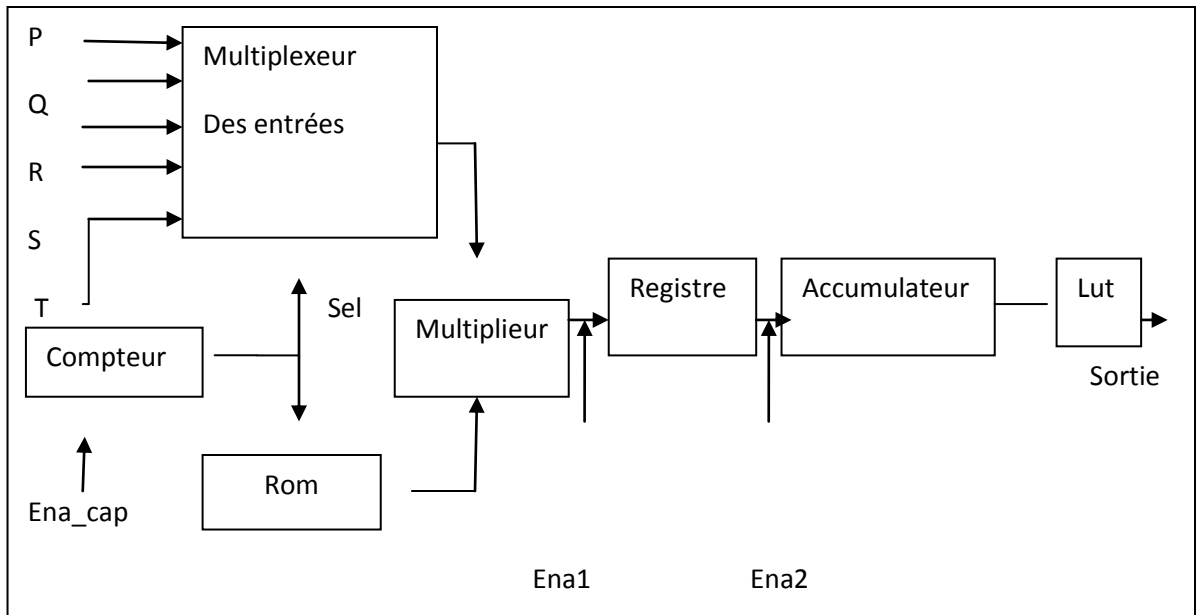


Figure IV.3: architecture du neurone

Ena1 : enable (load) de registre

Ena2 : enable (load) d'accumulateur

IV.3.2 Architecture du réseau de neurones

Après la simulation du neurone, on peut créer un réseau multicouche qui permet de relier les neurones entre eux par les connexions des poids entre eux. Et chaque neurone a son propre composant, chaque ROM est égale au nombre de nœuds relative à la couche d'entrée. Pour la même couche il y'a un calcul parallèle et entre les couches le calcul se fait en série. Ce réseau est commandé par l'unité de contrôle qui gère le fonctionnement du réseau comme présenté dans la figure (IV.4).

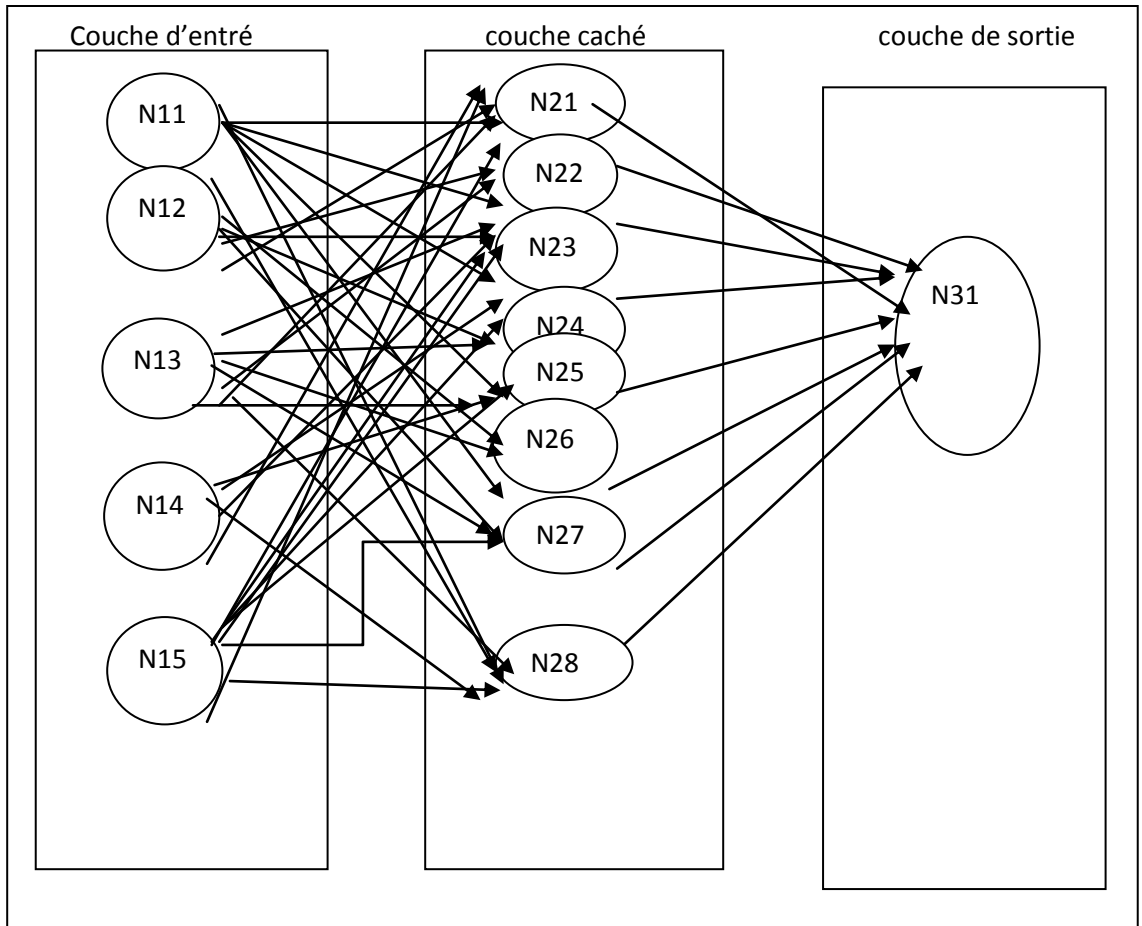


Figure IV.4 : le réseau de classificateur

IV.3.3 L'outil de conception de Xilinx : Vivado

Xilinx Vivado est l'environnement de développement logiciel de Xilinx. Il offre un ensemble complet d'outils familiers et puissants, de bibliothèques et de méthodologies.

Ce logiciel permet essentiellement d'effectuer les différentes étapes de la conception et de la réalisation de circuits numériques sur FPGA. Il permet entre autres de faire la description, la simulation, la synthèse et l'implémentation d'un circuit, puis la

programmation d'un circuit FPGA Vivado qui doit être installé sur le PC de l'utilisateur [35].



Figure IV.5: XILINX VIVADO

IV.4 Description en VHDL

IV.4.1 Description d'un Neurone

La description VHDL d'un neurone nécessite la description préalable des composants : Le compteur à une sélection de trois bits, le multiplexeur de huit entrées, chaque entrée est à huit bit, le multiplieur à dix huit bit, l'accumulateur à dix huit bit, les mémoires Rom qui contiennent les poids synaptiques à dix bit et une lut à des valeurs sur huit bit (voir ANNEX B).

L'architecture du neurone présentée dans la figure IV.3 peut être décrite en VHDL comme suit :

Chapitre 4 Implémentation Hardware du classificateur neuronal des arythmies cardiaques

```
Project Summary x  neuron.vhd x
D:/projetvivado_neurone/neurone/1neurone.srcs/sources_1/new/neurone.vhd
34 --use UNISIM.VComponents.all;
35 entity neurone is
36     Port ( clk,ena,ena1,ena2 : in STD_LOGIC;
37           e1,e2,e3,e4,e5 : in STD_LOGIC_VECTOR (3 downto 0);
38           sortie : out STD_LOGIC_VECTOR (7 downto 0));
39 end neurone;
40 architecture Behavioral of neurone is
41 component mmultiplicieur is
42     Port ( x : in STD_LOGIC_VECTOR (3 downto 0);
43           y : in STD_LOGIC_VECTOR (9 downto 0);
44           q : out STD_LOGIC_VECTOR (13 downto 0));
45 end component;
46 component compteur is
47     Port ( clk,ena : in STD_LOGIC;
48           q : out STD_LOGIC_VECTOR (2 downto 0));
49 end component;
50 component mux_8bit is
51     Port ( e1,e2,e3,e4,e5 : in STD_LOGIC_VECTOR (3 downto 0);
52           sel : in STD_LOGIC_VECTOR (2 downto 0);
53           u : out STD_LOGIC_VECTOR (3 downto 0));
54 end component;
55 component rom is
56 Port (
57     sel : in STD_LOGIC_VECTOR (2 downto 0);
58     data_out : out STD_LOGIC_VECTOR (9 downto 0));
59 end component;
60 component registre is
61     Port ( clk : in STD_LOGIC;
62           l: in std_logic;
63           d : in STD LOGIC VECTOR (13  downto 0);
64           g :out STD _LOGIC _VECTOR (13  downto 0));
65 end component;
66 component accumulateur2 is
67     Port ( clk,ena2: in STD_LOGIC;
68           m : in STD_LOGIC_VECTOR (13 downto 0);
69           r : inout STD_LOGIC_VECTOR (13 downto 0));
70 end component;
71 component lut is
72     Port ( adress : in STD_LOGIC_VECTOR (13 downto 0);
73           clk:in STD_LOGIC;
74           data_out : out STD_LOGIC_VECTOR (7 downto 0));
75 end component;
76 signal t1: std_logic_vector(2 downto 0);
77 signal t2: std_logic_vector(3 downto 0);
78 signal t3: std_logic_vector(9 downto 0);
79 signal t4: std_logic_vector(13 downto 0);
80 signal t5: std_logic_vector(13 downto 0);
81 signal t6: std_logic_vector(13 downto 0);
```

```
83 u0 : compteur port map(clk,ena,t1);
84 u1 : mux_8bit port map(e1,e2,e3,e4,e5,t1,t2);
85 u2 : rom port map(t1,t3);
86 u3 : mmultiplicieur port map (t2,t3,t4);
87 u4 : registre port map(clk,ena1,t4,t5);
88 u5 : accumulateur2 port map(clk,ena2,t5,t6);
89 u6 : lut port map(t6,clk,sortie);
90 end Behavioral;
```

Figure IV.6 : programme VHDL d'un neurone

IV.4.1.1 RTL analyse d'un neurone

La description RTL (Register Transfer Level description) permet de vérifier l'analogie de notre description VHDL avec l'architecture proposée.

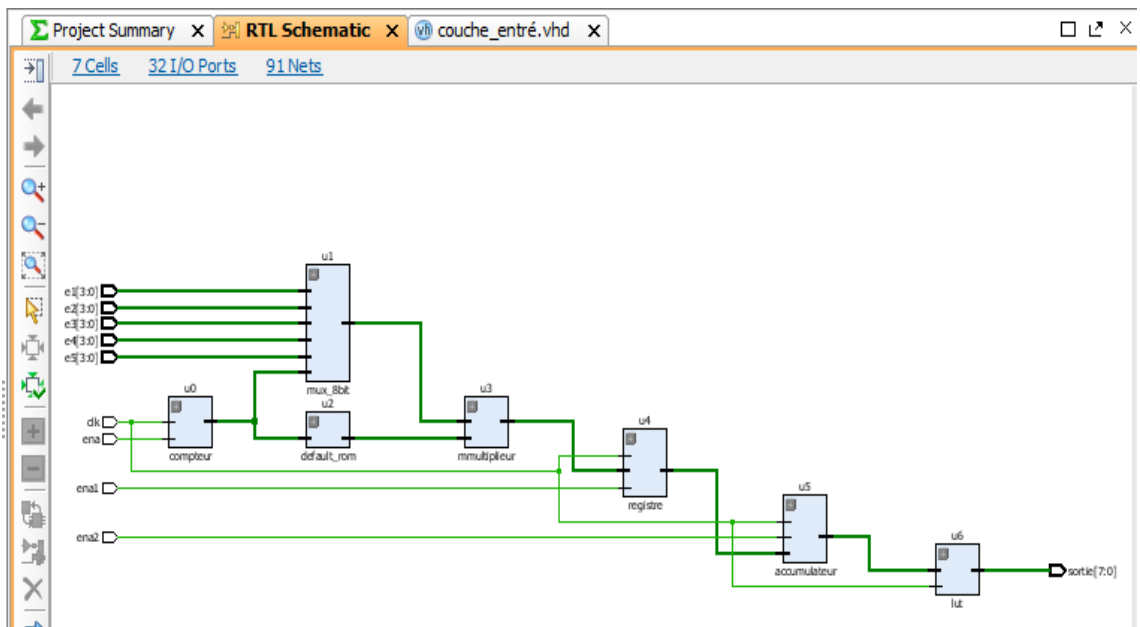


Figure IV.7: RTL analyse d'un neurone

Chapitre 4 Implémentation Hardware du classificateur neuronal des arythmies cardiaques

Blocs d'E/s	44
Interconnexions programmables	103
cellule logiques	7

Tab IV.1 : Table de résultats de l'analyse RTL

IV.4.1.2 Synthèse d'un neurone

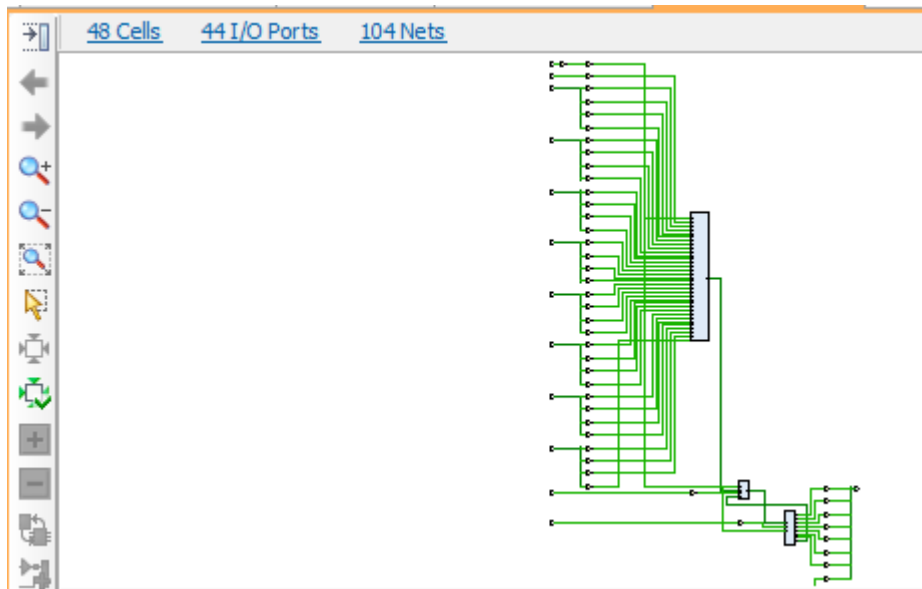


Figure IV.8: synthèse d'un neurone

Blocs d'E/s	44
Interconnexions programmables	104
Cellules logiques	48

Tab IV.2 : table des résultats de la synthèse

IV.4.1.3 implémentation d'un neurone

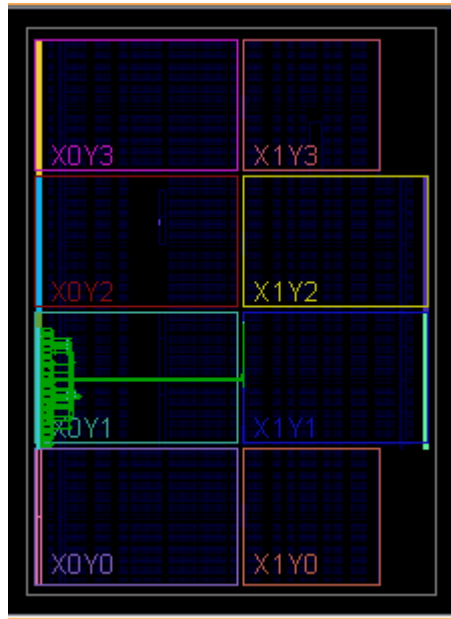


Figure IV.9: implémentation d'un neurone

Dans notre étude, la carte FPGA XC7A100T-1CSG324C est caractérisée par 210 I/OBs (Input/Output block-IOB) et 7925 CLBs (configurable logic block CLB), nombre de cellule logique 101440.

A partir de résultats précédents on peut dire que le neurone peut être implémenté sur le circuit ARTIX-7.

IV.4.2 description d'une couche cachée

La couche cachée composée de huit neurone chaque d'elle à cinq entrées correspond aux les paramètres morphologiques p, q, r, s, t. (voir l'annexe B)

IV.4.2.1 RTL analyse d'une couche cachée

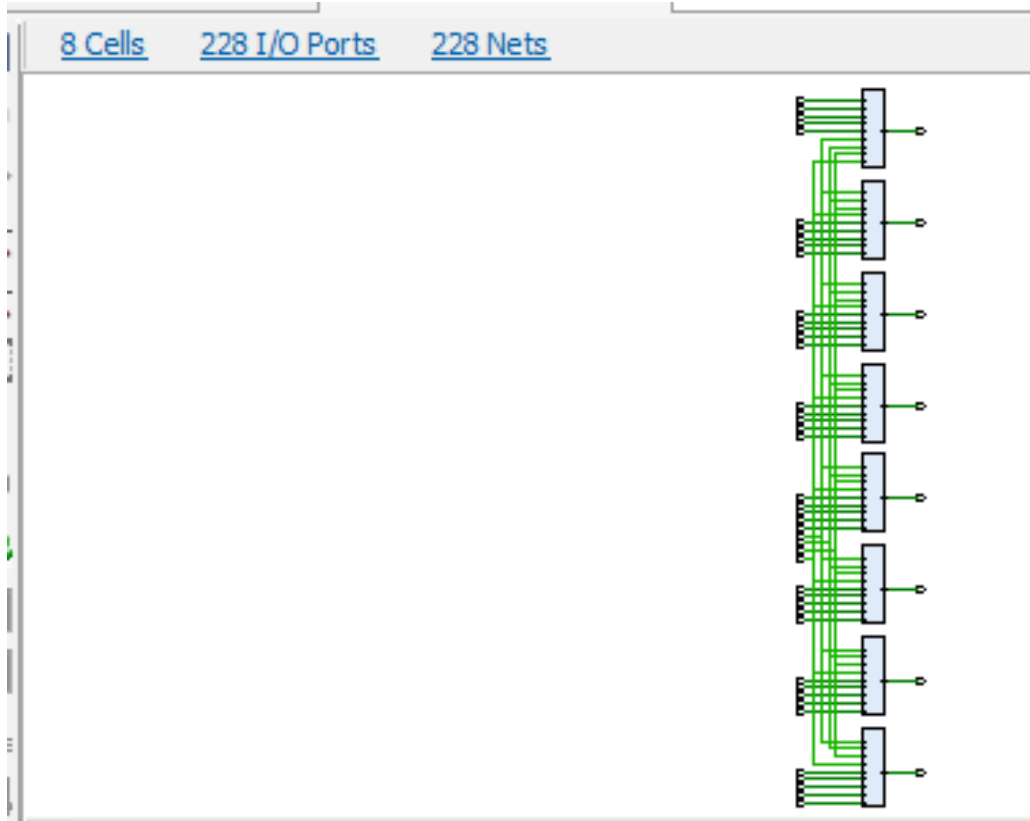


Figure IV.10: RTL analyse de la couche cachée

Blocs d'E/s	288
Interconnexions programmables	228
cellule logiques	8

Tab IV.3 : table RTL analyse de la couche cachée

IV.4.2.2 Synthèse d'une couche cachée

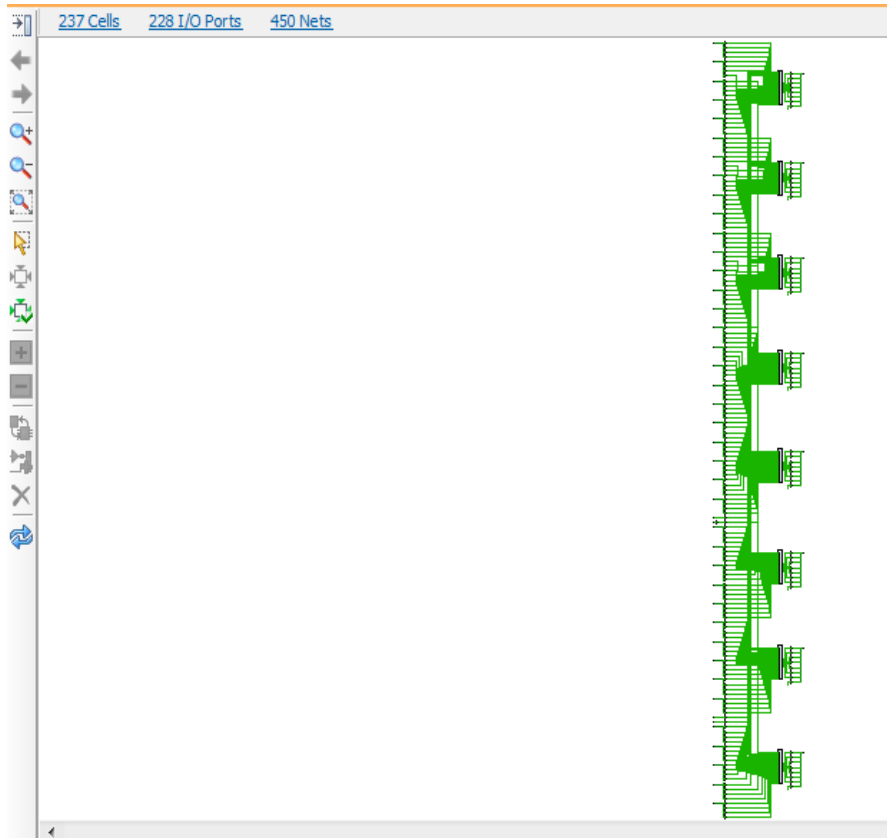


Figure IV.11: résultat de synthèse de la couche cachée

Blocs d'E/s	228
Interconnexions programmables	450
cellule logiques	237

Tab IV.4 : table résultat de la synthèse de la couche cachée

IV .4.2.3 Implémentation d'une couche cachée

A partir les résultats de la synthèse de la couche cachée, le nombre de blocs d'entrées sorties est supérieur a sel dans la carte FPGA artex7, donc ne peut pas voir l'implémentation.

Nous avons passé à la couche de sortie qui rassemble les sorties de la couche cachée pour nous pouvons implémenter le réseau.

IV.4.3 description d'une Couche de sortie

La couche de sortie contient un neurone à huit entrées à huit bit et une sortie de huit bit.

IV.4.4 description d'une couche cachée et couche de sortie

IV.4.4.1 RTL analyse

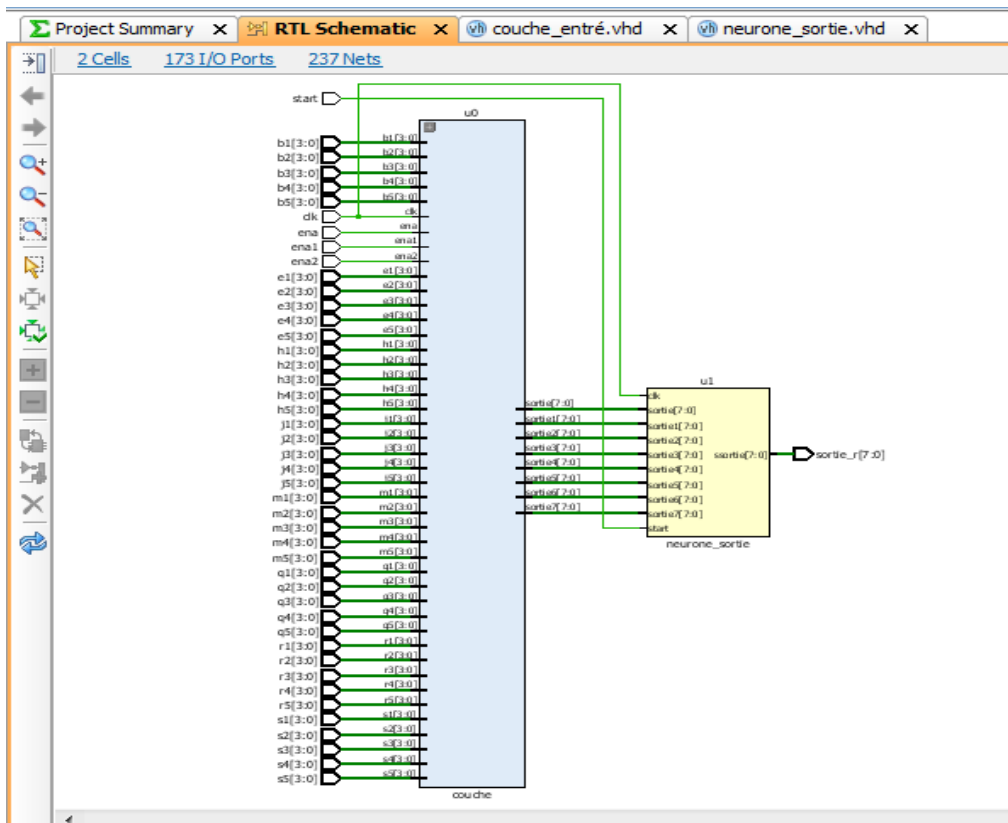


Figure IV.12: résultat de RTL analyse de la couche cachée et couche de sortie

Chapitre 4 Implémentation Hardware du classificateur neuronal des arythmies cardiaques

Blocs d'E/s	173
Interconnexions programmables	237
cellule logiques	2

Tab IV.5 : table résultat de RTL analyse de la couche cachée et couche de sortie

IV.4.4.2 Synthèse

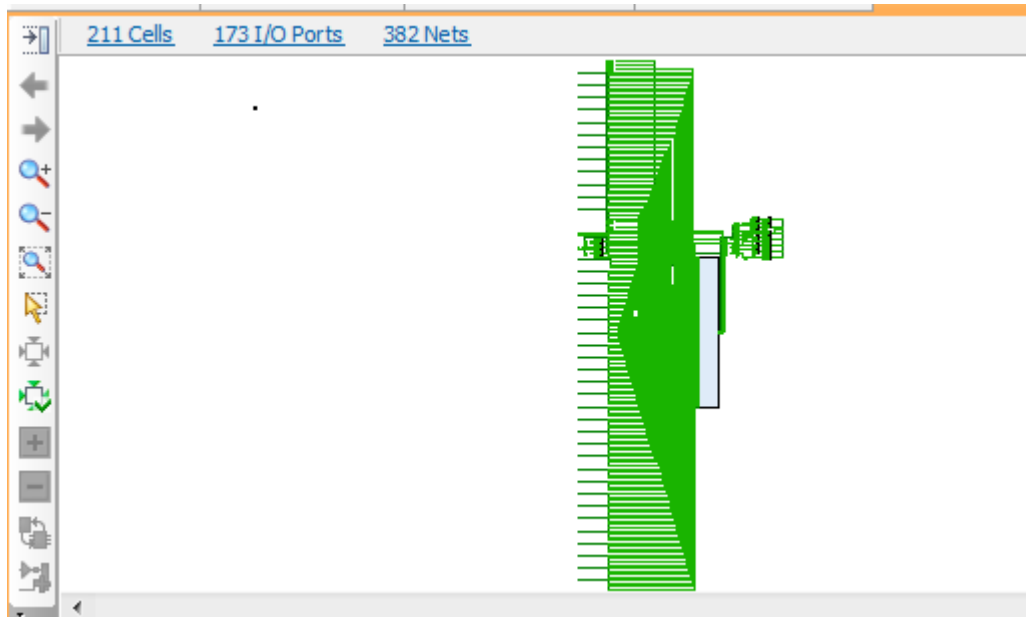


Figure IV.13: résultat de synthèse de la couche cachée et la couche de sortie

Blocs d'E/s	173
Interconnexions programmables	382
cellule logiques	211

Tab IV.6 : table résultat de la synthèse de la couche cachée et couche de sortie

D'après les résultats de la synthèse de la couche cachée et la couche de sortie, on peut dire qu'on peut implémenter notre réseau sur la carte artex7.

IV.4.4.3 Implémentation



Figure IV.14: résultat de l'implémentation de la couche cachée et couche de sortie

IV.5 Conclusion

La conception des réseaux de neurones est très complexe et nécessite beaucoup de temps pour sa mise en œuvre. Dans ce chapitre nous avons proposé une architecture générale d'un neurone artificiel et celle des couches qui constitue le réseau. L'outil de synthèse permet de réduire considérablement ce dernier. Après la simulation et le test on a pu généraliser ce neurone à un réseau de neurones multicouche pour le classificateur étudié.

Le travail effectué dans ce mémoire se rapporte à la conception et l'implémentation en technologie FPGA d'un classificateur neuronal des arythmies cardiaques.

Le projet initial proposé était une implémentation sur FPGA d'un nœud de capteur intelligent, qui se résume par une intégration du réseau de neurones dans l'architecture du système NEO430 où il s'agissait de créer une interface de communication entre ce système déjà existant et le réseau de neurones, l'application du réseau étant dédiée au classifieur d'arythmies cardiaques.

Dans la première partie de ce projet, nous avons fait une étude théorique et technique sur les réseaux de neurones et les arythmies cardiaques, et sur les approches pour la classification des arythmies, nous avons déduit les points suivants :

1. La classification des arythmies cardiaques doit prendre en charge les caractéristiques morphologiques du signal cardiaque tel que les ondes P, Q, R, S, T.
2. Les réseaux de neurones sont très appropriés pour la classification du signal cardiaque, vu les principales propriétés qu'ils présentent à savoir :
 - Habilité d'apprentissage et l'auto organisation à partir des exemples.
 - Possibilité de lire des formes en réponse à des nouvelles non apprises.

Dans la deuxième partie, nous nous sommes intéressés à l'étude des circuits FPGA, leurs structures, le langage VHDL, et l'outil xilinx vivado.

Toutefois, un ensemble de méthodes et d'algorithmes sont développés compte tenu de l'importance de ce signal et son exploitation en routine clinique dans le diagnostic des cas pathologiques cardiaques. Cette étude s'inscrit dans cette problématique et propose un classificateur des arythmies cardiaques par application des réseaux de neurones.

Dans cette étude nous avons proposé une approche pour la classification des arythmies cardiaques suivantes : (NSRDB) : signal normal sinus rhythm ; (SV) : signal supraventriculaire ; (CUDB) : signal ventricular tachyaryhmia

Nous avons étudié dans cette approche le signal cardiaque et nous avons extrait l'amplitude des ondes du signal afin de créer notre base de données et d'échantillons.

Après l'apprentissage, le système donne un taux d'apprentissage de 85,71% et de précision de 80%.

Les résultats justifient :

L'activité cardiaque qui constitue l'un des plus importants paramètres déterminant l'état d'un sujet. Elle se traduit par l'apparition des plusieurs ondes sur le tracé de l'électrocardiogramme.

L'analyse du signal ECG et l'identification de ses paramètres qui constituent une étape primordiale pour le diagnostic.

Après la phase de prétraitement qui représente le filtrage des signaux, nous sommes passées à l'étape de l'extraction semi-automatique des paramètres, à savoir l'extraction des paramètres du signal cardiaque manuellement des différents patients de la base de données «QT-Database».

Pour l'apprentissage, nous avons proposé un réseau basé sur l'algorithme de rétro propagation du gradient simulé sous matlab, la phase de test a donné de bons résultats mais il réside une variation au cours du temps.

Ensuite nous passons à l'étape de création du réseau de neurones sous xilinx vivado en langage VHDL à partir des poids et de la taille du réseau déterminés dans la phase d'apprentissage.

On peut dire que le but de notre projet a été atteint, les tests sous Matlab ont donné de bons résultats et les résultats de simulation obtenus sous l'environnement xilinx vivado ont été satisfaisants.

Ce travail nous a été bénéfique et nous a permis d'apprendre et de découvrir le domaine de la recherche et plus particulièrement le domaine de l'intelligence artificielle et nous a permis aussi de maîtriser l'outil xilinx vivado, facile à la conception et la mise en œuvre des systèmes électroniques et embarqués.

Comme perspectives

Pour un système automatique précis dans le domaine médicale, nous proposons un classificateur qui contient en entrée tous les paramètres du signal ECG: morphologique comme les amplitudes des ondes P,Q,R,S,T,U et temporelles comme les intervalles PP,RR,PR

Ces paramètres devraient être extraits par des algorithmes plus précisément pour permettre l'utilisation directe du système.

Pour le réseau, les poids synaptiques w_{ij} et w_{jl} obtenus après apprentissage sont :

✓ Valeurs réelles :

$W_{ij} = [1.1381 \ 1.1789 \ 2.9826 \ 5.2304 \ 1.1841;$
 $-4.2804 \ 12.318 \ -0.90475 \ 6.3484 \ -13.4099;$
 $2.7474 \ 0.30472 \ 0.24121 \ 2.6883 \ -1.2804;$
 $5.7919 \ 2.268 \ -1.1911 \ 1.7038 \ 1.2912;$
 $5.3947 \ -0.1517 \ -3.8611 \ 2.1053 \ -4.6343;$
 $-2.4575 \ 6.654 \ -7.681 \ -1.2654 \ -6.8935;$
 $0.88577 \ -0.63415 \ 4.51 \ -0.87512 \ -0.57134;$
 $-1.454 \ -1.6044 \ -1.4346 \ 4.2941 \ 6.2789]$

$W_{jl} = [-0.043812 \ -5.9592 \ 0.74235 \ 3.8803 \ -10.3457 \ -10.3027 \ 0.10963 \ 4.7324]$

✓ Le biais $B(1)$ et $B(2)$ obtenus après apprentissage sont :

$B(1) = [-1.4567;$

$-11.4424;$

$-2.7631;$

$-2.5768;$

$-7.9935;$

$0.4007;$

$2.0389;$

$-5.1327]$

$B(2) = [10.2226]$

Compteur

```

21
22 library IEEE;
23 use IEEE.STD_LOGIC_1164.ALL;
24 USE IEEE.std_logic_arith.all;
25 use IEEE.std_logic_unsigned.all;
26 -- Uncomment the following library declaration if using
27 -- arithmetic functions with Signed or Unsigned values
28 --use IEEE.NUMERIC_STD.ALL;
29
30 -- Uncomment the following library declaration if instantiating
31 -- any Xilinx leaf cells in this code.
32 --library UNISIM;
33 --use UNISIM.VComponents.all;
34 entity compteur is
35     Port ( clk,ena : in STD_LOGIC;
36           q : out STD_LOGIC_VECTOR (2 downto 0));
37 end compteur;
38 architecture Behavioral of compteur is
39 signal count: std_logic_vector(2 downto 0):="000";
40 begin
41 process (clk)
42 begin
43 if clk='1' and clk'event then
44 if ena='1' then
45 count<=count+"001";
46 end if;
47 end if;
48 end process;
49 q<=count;
50 end Behavioral;

```

Programme VHDL d'un compteur

Multiplexeur 8 bits des entrées

```

34 entity mux_8bit is
35     Port ( e1,e2,e3,e4,e5: in STD_LOGIC_VECTOR (3 downto 0);
36           sel : in STD_LOGIC_VECTOR (2 downto 0);
37           u : out STD_LOGIC_VECTOR (3 downto 0));
38 end mux_8bit;
39
40 architecture Behavioral of mux_8bit is
41 begin
42 process (e1,e2,e3,e4,e5)
43 begin
44     case sel is
45         when "000"=> u <= e1;
46         when "001"=> u <= e2;
47         when "010"=> u <= e3;
48         when "011"=> u <= e4;
49         when "100"=> u <= e5;
50         when others=> u<="0000";
51     end case;
52     END PROCESS;
53 end Behavioral;

```

Programme VHDL d'un Multiplexeur

La rom

```

Project Summary x rom.vhd x
D:/projetvivo_1neurone/1neurone/1neurone.srscs/sources_1/new/rom.vhd
27 -- arithmetic functions with Signed or Unsigned values
28 --use IEEE.NUMERIC_STD.ALL;
29
30 -- Uncomment the following library declaration if instantiating
31 -- any Xilinx leaf cells in this code.
32 --library UNISIM;
33 --use UNISIM.VComponents.all;
34
35 entity rom is
36 Port (
37     sel : in STD_LOGIC_VECTOR (2 downto 0);
38     data_out : out STD_LOGIC_VECTOR (9 downto 0));
39 end rom;
40 architecture Behavioral of rom is
41 type rom is array (4 downto 0) of std_logic_vector(9 downto 0);
42 signal ROM_data:rom:=("0000001011","0001111011","0000000011",
43 "0000010110","0000000001");
44 begin
45 data_out<=ROM_data(conv_integer (sel));
46 end Behavioral;

```

Programme VHDL d'une ROM

Multiplieur

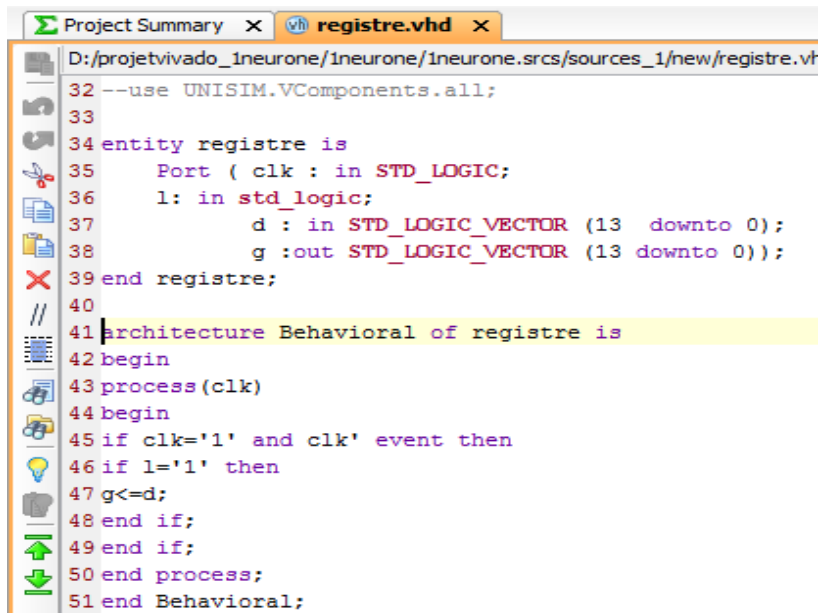
```

Project Summary x mmultiplieur.vhd x
D:/projetvivo_1neurone/1neurone/1neurone.srscs/sources_1/new/mmultiplieur.vhd
27 USE IEEE.NUMERIC_STD.ALL;
28 -- Uncomment the following library declaration if using
29 -- arithmetic functions with Signed or Unsigned values
30 --use IEEE.NUMERIC_STD.ALL;
31
32 -- Uncomment the following library declaration if instantiating
33 -- any Xilinx leaf cells in this code.
34 --library UNISIM;
35 --use UNISIM.VComponents.all;
36
37 entity mmultiplieur is
38     Port ( x : in STD_LOGIC_VECTOR (3 downto 0);
39           y : in STD_LOGIC_VECTOR (9 downto 0);
40           q : out STD_LOGIC_VECTOR (13 downto 0));
41 end mmultiplieur;
42
43 architecture Behavioral of mmultiplieur is
44 begin
45 q<=x*y;
46 end Behavioral;

```

Programme VHDL d'un multiplieur

Registre



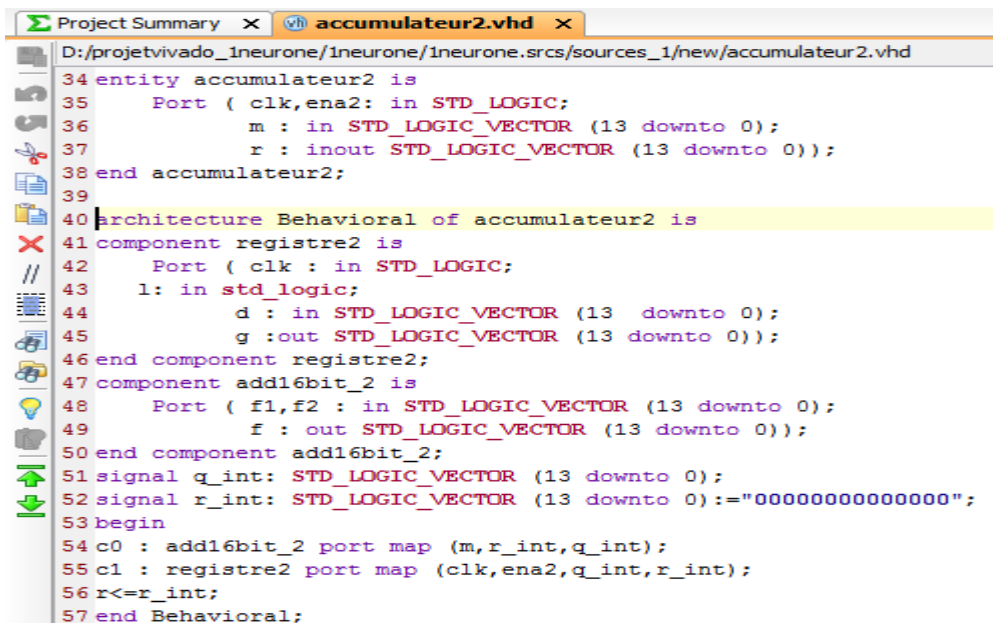
```

Project Summary x vh registre.vhd x
D:/projetvivado_1neurone/1neurone/1neurone.srscs/sources_1/new/registre.vf
32 --use UNISIM.VComponents.all;
33
34 entity registre is
35     Port ( clk : in STD_LOGIC;
36           l: in std_logic;
37           d : in STD_LOGIC_VECTOR (13 downto 0);
38           g :out STD_LOGIC_VECTOR (13 downto 0));
39 end registre;
40
41 //
42 architecture Behavioral of registre is
43 begin
44 process(clk)
45 begin
46 if clk='1' and clk' event then
47 if l='1' then
48 g<=d;
49 end if;
50 end if;
51 end process;
52 end Behavioral;

```

Programme VHDL d'un registre

Accumulateur



```

Project Summary x vh accumulateur2.vhd x
D:/projetvivado_1neurone/1neurone/1neurone.srscs/sources_1/new/accumulateur2.vhd
34 entity accumulateur2 is
35     Port ( clk,ena2: in STD_LOGIC;
36           m : in STD_LOGIC_VECTOR (13 downto 0);
37           r : inout STD_LOGIC_VECTOR (13 downto 0));
38 end accumulateur2;
39
40 architecture Behavioral of accumulateur2 is
41 component registre2 is
42     Port ( clk : in STD_LOGIC;
43           l: in std_logic;
44           d : in STD_LOGIC_VECTOR (13 downto 0);
45           g :out STD_LOGIC_VECTOR (13 downto 0));
46 end component registre2;
47 component add16bit_2 is
48     Port ( f1,f2 : in STD_LOGIC_VECTOR (13 downto 0);
49           f : out STD_LOGIC_VECTOR (13 downto 0));
50 end component add16bit_2;
51 signal q_int: STD_LOGIC_VECTOR (13 downto 0);
52 signal r_int: STD_LOGIC_VECTOR (13 downto 0):="00000000000000";
53 begin
54 c0 : add16bit_2 port map (m,r_int,q_int);
55 c1 : registre2 port map (clk,ena2,q_int,r_int);
56 r<=r_int;
57 end Behavioral;

```

Programme VHDL d'un accumulateur.

Add_16bit

```

D:/projetvivoado_1neurone/1neurone/1neurone.srcs/sources_1/new/add16bit_2.vhd
34 entity add16bit_2 is
35     Port ( f1,f2 : in STD_LOGIC_VECTOR (13 downto 0);
36           f : out STD_LOGIC_VECTOR (13 downto 0));
37 end add16bit_2;
38 Architecture Behavioral of add16bit_2 is
39 component add1bit_2 is
40     Port ( a,b : in std_logic;
41           res : out std_logic);
42 end component add1bit_2;
43 begin
44 U0: add1bit_2 port map (f1(0),f2(0),f(0));
45 U1: add1bit_2 port map (f1(1),f2(1),f(1));
46 U2: add1bit_2 port map (f1(2),f2(2),f(2));
47 U3: add1bit_2 port map (f1(3),f2(3),f(3));
48 U4: add1bit_2 port map (f1(4),f2(4),f(4));
49 U5: add1bit_2 port map (f1(5),f2(5),f(5));
50 U6: add1bit_2 port map (f1(6),f2(6),f(6));
51 U7: add1bit_2 port map (f1(7),f2(7),f(7));
52 U8: add1bit_2 port map (f1(8),f2(8),f(8));
53 U9: add1bit_2 port map (f1(9),f2(9),f(9));
54 U10: add1bit_2 port map (f1(10),f2(10),f(10));
55 U11: add1bit_2 port map (f1(11),f2(11),f(11));
56 U12: add1bit_2 port map (f1(12),f2(12),f(12));
57 U13: add1bit_2 port map (f1(13),f2(13),f(13));
58

```

Programme VHDL d'un add_16bit

Add_1bit

```

add_16bit.vhd x add_1bit.vhd x compteur.vhd x
D:/projetvivoado_1neurone/1neurone/1neurone.srcs/sources_1/new/add_1bit.vhd
26 -- Uncomment the following library declaration if using
27 -- arithmetic functions with Signed or Unsigned values
28 --use IEEE.NUMERIC_STD.ALL;
29
30 -- Uncomment the following library declaration if instantiating
31 -- any Xilinx leaf cells in this code.
32 --library UNISIM;
33 --use UNISIM.VComponents.all;
34
35 entity add_1bit is
36     Port ( a,b : in std_logic;
37           res : out std_logic);
38 end add_1bit;
39 architecture Behavioral of add_1bit is
40 begin
41     res<=a or b ;
42 end Behavioral;
43

```

Programme VHDL d'un add_1bit

Lut

```
Project Summary x vh lut.vhd x
D:/projetvivado_1neurone/1neurone/1neurone.srcs/sources_1/new/lut.vhd
33 --use UNISIM.VComponents.all;
34
35 entity lut is
36     Port ( adress : in STD_LOGIC_VECTOR (13 downto 0);
37           clk:in STD_LOGIC;
38           data_out : out STD_LOGIC_VECTOR (7 downto 0));
39 end lut;
40 architecture Behavioral of lut is
41 type rom is array (31 downto 0) of std_logic_vector(7 downto 0);
42 signal ROM_data:rom:=("00000000","00000000","00000000","00000000","00000000",
43 "00000001",
44 "00000100",
45 "00001011",
46 "00011010",
47 "00011110",
48 "00101000",
49 "00110010",
50 "00111100",
51 "01001001",
52 "01011000",
53 "01011111",
54 "01100010",
55 "01100011",
56 "01100011","01100011","01100011","01100011","01100011",
57 "01100011","01100011","01100011","01100011","01100011","01100011","01100011",
58 "01100011","01100011",
59 others=> ("01100100"));
60 begin
61 data_out<=ROM_data(conv_integer(adress));
62 end Behavioral ;
```

Programme VHDL d'une lut

- [1]: A. Bensmail. Implémentation des réseaux de neurones sur FPGA par l'utilisation de la reconfiguration dynamique. Mémoire de master en systèmes embarqués, ENST, 2016.
- [2]: Y.Djeriri. Les réseaux neurones artificiels. UDL-SBA-2017.
- [3]: N.izeboudjen « plateforme pour l'implémentation des réseaux de neurones sur FPGA application à l'algorithme de rétro propagation du gradient ». Thèse de doctorat, ENP, 2014.
- [4]: PEOI, Ian's Web Page ; <https://www.peoi.org/Courses/Coursesfr/neural/neural3.html>, date de consultation : mars 2020
- [5]: Yann MORÈRE, Ian's Web Page ; <http://www.morere.eu/spip.php?article19>, date de consultation : avril 2020
- [6]: F.TSCHIRHART. RESEAUX DE NEURONES FORMELS APPLIQUES A L'INTELLIGENCE ARTIFICIELLE ET AU JEU. Mémoire de recherche, ESGI, Paris, 2009.
- [7]: G. Dreyfus. «Réseaux de neurones : Méthodologie et applications », Eyrolles, Paris, ISBN : 2212114648, 9782212114645, 2002.
- [8]: N. Izeboudjen. Conception et implémentation en FPGA d'un classificateur neuronal des arythmies cardiaques. Thèse de magister en électronique, ENP, 1999.
- [9] : P. Chollet. Traitement parcimonieux de signaux biologique. THÈSE DE DOCTORAT / IMT Atlantique sous le sceau, l'Université Bretagne en Electronique École Doctorale Mathématiques et STIC, 2017.
- [10]: A.BABA HAMED. Vers un modèle De Classification Neuronale Des Données Médicales A Base De La Technologie FPGA. Thèse de doctorat en Génie Biomédical, Université Aboubakr Belkaïd Tlemcen, 2017.
- [11]: A .Mokadam . Segmentation et Classification des signaux non-stationnaires. Application au traitement des sons cardiaques et à l'aide au diagnostic. Thèse de doctorat en Traitement du Signal, de l'Université de Haute Alsace, 2012.
- [12]: M. Zaamouche. Classification des battements cardiaques en utilisent les réseaux de neurones profonds. Mémoire de master académique en instrumentation, université 8 mai 1945-Guelma, 2019.

- [13]: M. Nait-hamoud. Segmentation et classification du signal ECG par les SVM flous multi-classes. Thèse de magistère en informatique, université cheikh labri Tbessa-Tébessa, 2010.
- [14]: Z. Zidelmal. Reconnaissance d'arythmies cardiaques par support vector machines (SVMS). Thèse de doctorat en Electronique, université mouloud Mammeri Tizi Ouzou, 2012.
- [15]: R. Halimi, Y. Hammouya. Classification d'un signal ECG par RNA (RBF). Mémoire de master en télécommunications, université Kasdi Merbah Ouargla, 2019.
- [16]: P. Borne, M. Benrejeb, J. Hggège, « les réseaux de neurones : Présentation et applications », Editions Technip, Paris, ISBN : 978-2-7108-0896-1, ISSN : 1152-0647, 2007.
- [17]: D .SAPTONO .Conception d'un outil de prototypage rapide sur le FPGA pour des applications de traitement d'images. Thèse de doctorat DE L'UNIVERSITE de BOURGOGNE en Instrumentation et Informatique de l'Image .2011.
- [18]: N. boufala. Implémentation matérielle de réseau de neurones probabiliste pour une interface myélectrique embarquée .master automatique Bejaia, en 2012.
- [19]: A. Belgacem. Classification des signaux EGC avec un système-multi-agent neuronale. Magister en informatique, Tlemcen, 2012.
- [20]: L. Manai, R. Bouzid. Analyse et classification des signaux ECG sous MATLAB et implémentation d'une solution embarquée. Conférence international en automatique & traitement de signal (ATS 2017), vol 24 PP 13_17. Robotique, Informatique et Systèmes Complexes (RISC), ENIT, université El Manat Tunis.
- [21]: R.BENALI « Analyse du signal ECG en vue de la reconnaissance de pathologies cardiaques » thèse de doctorat, université de Tlemcen avril 2013.
- [22]: M.Boukhatem. Calcul du rythme cardiaque par la détection des pics R pour un signal ECG, master en génie électrique, université de boumerdes, 2016.
- [23]: M. ZAAMOUCHE. « Classification des battements cardiaques en utilisant les réseaux de neurones profonds», magister en instrumentation, 2019.
- [24]: K. Bensafia « Transmission sans fils, par voie GSM, et traitement du signal ECG», thèse de magister, 2018.
- [25] : P. JIAPU, willis J. Tompkins « transections on bimédical engineering », vol.BME 32.NO, 1985.

- [26] : A.Rene . « LA TRANSFORMATION EN ONDELETTES.Universite Pierre et Marie Curie.
- [27]: P.Valérie Application de la théorie des ondelettes. Institut National Polytechnique de Grenoble .Enseignement UNESCO Traitement du signal et des images numériques, Tunis, ENIT, 14-18 mars 2005.
- [28]: M.BHOURI, R.FERCHICHI. simulation-signal biomédicale, projet tutoriel, 2017.
- [29] : PhysioNet, Ian's Web Page ; <https://archive.physionet.org/cgi-bin/atm/ATM>, date de consultation : juillet 2020.
- [30]: R.Halimi, Y.Hammouya. Classification d'un signal ECG par RNA (RBF), master en génie Telecommunications, université Kasdi Merbah Ouargla, 2019.
- [31]: M.HENDEL, A.BENYETTOU, H.KHELIL. Classification des arythmies cardiaques par réseaux de neurones artificiels, 5th International Conference : Sciences of Electronic, Technologies of Information and Telecommunications March 22-26, 2009 – TUNISIA, 5 pages, Setif, 2009.
- [32]: PRIYA « identification of individual melodies using artificial neural networks classifier », master of engineering, wireless communication, ECED Thapar University, 2015.
- [33]: B.CHEFRI.ETUDE DU COMPORTEMENT NON LINEAIRE ET DE L'ENDOMAGEMENT SOUS SOLlicitation THERMO MECANIQUE DES STRUCTURES MECANIQUE. Magistère en génie mécanique université de BATNA, 2014.
- [34] : A.GUELLAL « Les circuits FPGA: description et applications ». Attaché de recherche, Division: Energie solaire Photovoltaïque,Equipe : Economie et Maîtrise de l'Energie,N°24.
- [35]: « Compréhension – interpréter l'information », Ian's Web Page ; http://fr.wikipedia.org/wiki/Taxonomie_de_Bloom, date de consultation : septembre 2020.
- [36]: B, Benguettaf, R. Zaoui. « Implémentation des réseaux de neurones artificiels (RNA) sur "FPGA" pour le diagnostic des defaillances de la machine asynchrone », magister en commande électrique, magister ,2007.