REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE

MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE

UNIVERSITE Dr. YAHIA FARES - MEDEA –



FACULTE DES SCIENCES

DEPARTEMENT DE MATHEMATIQUES ET D'INFORMATIQUES

Projet de fin d'études

Présenté pour l'obtention du Diplôme de Master en Informatique

Option : Ingénierie des Systèmes et Technologie du Web

La détection de la maladie du cancer à partir des Données des puces à ADN à l'aide de L'apprentissage profond

Préparé par :

- Melle Korteby Amira
- Melle Amarsetti Asmaa

Dirigé par :

MrSahmadi Ibrahim

2020-2021

Remerciement :

Tout d'abord Nous tenons à remercier Dieu tout puissant et miséricordieux, qui nous a donné la force et la patience d'accomplir ce Modeste travail.

Un profond respect et un remerciement particulier à notre encadreur Mr : ***SAHMADI Brahim****Pourses conseils, sa disponibilité et sa patience, Sa grande connaissance dans le domaine, ainsi que son expérience, ont joué un rôle Important dans la conception de ce travail.*

Nos vifs remerciements vont également aux membresDu jury pour l'intérêt qu'ils ont porté à notre recherche enAcceptant d'examiner notre travail Et de l'enrichir parLeurs propositions.

Enfin nous tenons également à remercier tous les enseignants et le corps administratif de la faculté de Sciences à l'université de Médéa.

« Asmaa &Amira »

Dédicaces :

Je dédie ce travail :

À l'homme de ma vie, celui qui s'est toujours sacrifié pour me voir réussir, à toi mon père que nulle dédicace ne peut exprimer mes sincères sentiments.

À la lumière de mes jours, la source de mes efforts, ma vie et mon bonheur, maman que j'adore énormément.

À Mon frère Oussama, ma sœur Anfel, et ma chère tante khadidja pour leur amour et soutien

À mon fiancé Ramzi pour son encouragement et ses aides.

À mon binôme et ma sœurAmira le travail avec toi est un plaisir.

À ma chère copine Katrennada et à toute ma grande famille.

Merci à tous ceux qui ont cru en moi.

Asmaa …



Dédicaces:

Je dédie ce travail :

À Mes chers parents, que nulle dédicace ne peut exprimer mes sincères sentiments, pour leurs patience illimitée, leurs encouragements, leurs aides, en témoignage de mon profond amour et respect pour leurs grands sacrifices.

À Mes adorables sœurs : Meriem et Zahida pour leur amour et soutien qu'elles trouvent ici l'expression de ma haute gratitude.

À mes petits poussin Youcef et Ahmed et leur gentil papa Tewfik.

À mon binôme et ma sœur Asmaa le travail avec toi est un plaisir.

À ma chère copine KatrennadaEt à tous ceux que j'aime.

Merci à tous ceux qui ont cru en moi.

Amira ...



Résumé :

Actuellement, le cancer est la principale cause de mortalités dans le monde. D'après les statistiques de l'OMS (organisation mondiale de la santé), environ 1 décès sur un totale de 6 est dû à un cancer, Ainsi d'ici 2030 le nombre de nouveaux cas de cancer estimés par an pourrait augmenter jusqu'à 25 millions.Cependant, le diagnostic précoce peut sauver d'innombrables vies et économiser des milliards de dollars. A cet effet les chercheurs dans le domaine de la bio-informatique ont optés à la techniquedes puces à ADN qui contiennent des données d'expression génique qui vont être analysées et traitées à travers l'apprentissage profond. Dans ce contexte les objectifs de cette thèse ont pour but de développer un modèle d'apprentissage profond en s'appuyant sur le***CNN*** et ses architectures ***VGG16, Xception***avec l'application de la méthode ***DeepInsight***, ce modèle a la capacité de classer les gènes stockés dans des puces ADN, et gérer le problème de la grande dimensionnalité des caractéristiques grâce à la méthode ***CFS***, ce modèle a été testé sur quatre datasets Colon cancer, prostate cancer, leukimia cancer, Cns cancer.

Les mots clés :cancer, bio-informatique, puce à ADN, CNN,CFS, VGG16, Xception,et Deepinsight.

Abstract:

Currently, cancer is the leading cause of most deaths in the world. According to World Health Organization statistics, about 1 in 6 total deaths is due to cancer, so by 2030 the number of estimated new cancer cases per year could increase to 25 million, However, early diagnosis of cancer can save countless lives and billions of dollars, indeed, researchers in the field of bioinformatics have opted for DNA chips that contain gene expression data to be analyzed and processed through deep learning. In this context, the objectives of this thesis were to develop a deep learning model based on the CNN and its VGG16, Xception architectures with the application of the DeepInsight method, this model has the ability to classify the genes stored in DNA chips, and manage the problem of high dimensionality of the features through the CFS method.this model was tested on four datasets Colon cancer, prostate cancer, leukimia cancer, Cns cancer.

Keywords: cancer, bioinformatics, DNA chip, CNN, CFS, VGG16, Xception, Deepinsight.

ملخص:

يعتبر السرطان حاليا السبب الرئيسي لمعظم الوفيات في جميع انحاء العالم , وفقالإحصاءات المنظمة العالمية للصحة فان حوالي واحد من كل ست حالات وفاة هي ناتجة عن السرطان, و يمكن ان يرتفع عدد الحالات الجديدة المقدرة سنويا الى 25مليونا و ذلك بحلول سنة 2030. ولهذا السبب فان التشخيص المبكر للسرطان يمكن ان ينقذ العديد من الأرواح و يوفر مليارات الدولارات, لهذا الغرض لجأ الباحثون في مجال المعلوماتية الحيوية الى شرائح الحمض النووي التي تحتوي على بيانات التعبير الجيني التي سيتم معالجتها و تحليلها من خلال تقنية التعلم العميق.الهدف الرئيسي من هذه الاطروحة هوتطوير نموذج التعلم العميق على أساس CNN وهياكلهاVGG16 ، Xceptionمع استعمال خاصيةتسمح بتحويل المعلومات الرقمية المخزنة في الشرائح الى صور DeepInsight، بالإضافة الى الاعتماد على طريقة دقيقةCFS تسمح باختيار الخائص المهمة فقط من اجل تصنيفها الى قسمين، جينات مريضة وأخرى سليمة.تم اختبار هذا النموذج على أربع مجموعات بيانات: سرطان القولون، سرطان البروستات، سرطان اللوكيميا، سرطان الجهاز العصبي المركزي.

الكلمات المفتاحية: سرطان، المعلوماتية الحيوية، شرائح الحمض النووي، بيانات التعبير الجيني، التعلم العميق.

Table des matières :

Remerciments I
Dedicaces II
Résumé II
Tables des figures IV
Tables des tableaux IIV
Introduction générale 10
Chapitre 01 : Technologie des puces ADN 26
1 Introduction: 15
2 La bio-informatique: 15
2.1 Définition : 15
2.2 Le principe de la bio-informatique et ses champs d'application : 15
3 La technologie des puces à ADN : 16
3.1 Définition : 16
3.2 Le principe de fonctionnement : 16
3.3 Les phases d'analyse d'une puce à ADN : 18
3.3.1 La préparation de la cible et l'hybridation : 18
3.3.2 L'acquisition et analyse des images : 19
3.3.3 La normalisation des données : 20
3.3.4 La Présentation des données de puces à ADN : 20
4 Les types des puces d'ADN : 21
4.1 Les macro-array ou filtre à haute densité : 21
4.2 Les micro-array : 21
4.3 La puce à oligonucléotides : 21
5 L'utilisation des puces à ADN : 21
5.1 Le diagnostic de la maladie : 22
5.2 La découverte des médicaments : 22
6 Les puces à ADN et le cancer : 22
6.1 La maladie du cancer : 22
6.2 L'apport des puces à ADN en cancérologie : 23
7 Conclusion : 24
Chapitre 02 : Apprentissage profond 26
1 Introduction : 26
2 Définition de l'apprentissage profond: 26
2.1 La classification de l'apprentissage profond : 27

2.1.1 Apprentissage profond supervisé : 27
2.1.2 Apprentissage profond non supervisé : 28
2.1.3 Apprentissage profond semi supervisé : 28
2.2 Les domaines d'applications de l'apprentissage profond : 28
2.2.1 Le Robotique : 28
2.2.2 L'agriculture : 29
2.2.3 La santé : 29
2.2.4 La détection de malwares ou de fraudes : 30
2.3 Les architectures de l'apprentissage profond : 31
2.3.1 L'auto-encoder : 31
2.3.2 Réseau de neurones récurrent (Recurrent Neural Network) : 32
2.3.3 Réseau de neurones récursif (Recursive Neural Network) : 32
2.3.4 Réseau de neurones convolutif (Convolutional Neural Network) : 33
2.3.5 La partie d'extraction des caractéristiques : 34
2.3.6 La partie de la classification : 36
2.3.7 Exemples d'architectures du CNN : 37
3 Apprentissage par transfert : 41
3.1 Définition : 41
3.2 Apprentissage par transfert sur des images : 41
3.3 Avantage de l'apprentissage par transfert : 42
4 Les étapes d'application d'un modèle d'apprentissage profond : 43
4.1 Le prétraitement des données : 43
4.1.1 La standardisation des données : 43
4.1.2 La normalisation des données : 44
4.2 L'équilibre des données : 44
4.2.1 Le sous-échantillonnage « Undersampling » : 45
4.2.2 Le sur-échantillonnage « Oversampling » : 45
5 La transformation des données non images en format image : 45
5.1 La méthode Deepinsight : 45
5.1.1 Définition : 45
5.1.2 Principe de fonctionnement : 45
5.1.3 L'implémentation de DeepInsght : 46
5.2 La technique de réduction de dimensionnalité T-SNE : 47
5.2.1 Définition : 47
5.2.2 Principe de fonctionnement : 47
5.2.3 L'intérêt de la réduction de dimensionnalité : 48

5.3 La normalisation des images : 48
5.3.1 La première normalisation : 48
5.3.2 La deuxième normalisation : 48
6 Les mesures d'évaluations: 49
6.1 La matrice de confusion : 49
6.2 La précision : 50
6.3 Le rappel : 50
6.4 F mesure : 50
6.5 Accuracy : 50
7 Conclusion 50
Chapitre 03 : Conception de la méthode proposée 51
1 Introduction : 51
2 L'approche proposée : 51
3 Schéma de l'approche proposé : 51
4 Les Jeux de données : 53
4.1 Définition : 53
4.2 Description des jeux de données utilisées : 53
4.2.1 Colon cancer : 53
4.2.2 Leukemia cancer : 53
4.2.3 Prostate cancer : 53
4.2.4 Tumeur du système nerveux central : 54
5 Les étapes d'application de l'approche : 54
5.1 Le prétraitement des données : 54
5.2 La normalisation des données : 54
5.3 Le filtrage des gènes: 55
5.4 La séparation des données : 56
5.4.1 Le Train Set : 56
5.4.2 Le Test Set : 56
5.5 L'équilibre des données : 57
6 L'implémentation de la méthode Deepinsight : 58
6.1 La Normalisation : 58
6.2 L'installation de transformateur des images : 58
6.3 L'entraînement du transformateur : 58
6.4 La matrice de densité: 60
6.5 L'ajustement des pixels : 60

6.6 Transformation en image: 62
6.7 Affichage des images d'entraînement : 62
6.8 La transformation des données de test : 63
7 La Description des Modèles implémentées : 63
7.1 Description du modelé 1D : 63
7.1.1 Colon : 63
7.1.2 Prostate : 63
7.1.3 Leukimia : 64
7.1.4 CNS : 64
7.2 Description du modèle 2D: 69
7.2.1 Architecture VGG16 : 69
7.2.2 Architecture Xception : 69
7.2.3 Modèle CNN simple : 73
7.3 Conclusion: 75
Chapitre 04 : Implémentation et résultats 76
1 Introduction : 76
2 Outils et environnement : 76
2.1 Matériel utilisé : 76
2.2 Language de programmation: 76
2.2.1 Python : 76
2.2.2 Caractéristiques : 76
2.3 Environnement d'implémentation : 77
2.3.1 Google colaboratory : 77
2.4 Les modules utilisés : 78
2.4.1 Tensorflow : 78
2.4.2 Keras : 78
2.4.3 Scikit-learn : 78
2.4.4 Numpy : 79
2.4.5 Pandas : 79
2.4.6 IO : 79
2.4.7 imbalanced-learn : 79
2.4.8 Matplotlib : 79
3 Résultats et discussion : 80
3.1 Structure des modèles utilisés : 80
3.2 Résultats obtenus : 85

3.2.1 Colon dataset : 85
3.2.2 Modèles CNN 2D : 86
3.3 Prostate dataset : 87
3.3.1 Modèle CNN 1D : 87
3.3.2 Modèle 2D : 88
3.4 Leukimia dataset : 89
3.4.1 Modèle CNN 1D : 89
3.5 CNS Dataset : 91
3.5.1 Modèle CNN 1D : 91
3.5.2 Modèle CNN 2D : 91
4 Comparaison avec les classicateurs classiques : 93
4.1 Machine à vecteurs de support (SVM) : 93
4.2 Random forest: 93
4.3 Discussion: 94
5 Conclusion : 95
Conclusion générale 96
Bibliographies 98
Webographies 101

Liste des figures :

Figure I.1: principe des puces ADN 17
Figure I.2: étapes des analyses d'une puce à ADN 18
Figure I.3: hybridation d'une puce à ADN 18
Figure I.4: acquisition d'image dans une analyse par puce à ADN 19
Figure II.1 : Les sous-bronches de l'intelligence artificielle 27
Figure II.2 : les différentes couches d'un réseau neuronal 28
Figure II.3 : l'apprentissage profond en robotique 30
Figure II.4 : l'apprentissage profond en agriculture 30
Figure II.5 : l'apprentissage profond et la santé 31
Figure II.6 : l'apprentissage profond et la détection des malwares 31
Figure II.7 : Les parties des autoencodeurs 32
Figure II.8 : le réseau de neurones récurrent 33
Figure II.9 : le réseau de neurones récursif 34
Figure II.10 : les couches d'un réseau de neurones convolutif 35
Figure II.11 : fonctionnement des couches convolutives 36
Figure II.12 : la différence entre max-pooling et average-pooling 36
Figure II.13 : La fonction d'activation ReLU 37
Figure II.14 : la couche fullyconnected et tles fonctions d'activation 38
Figure II.15 : les différentes couches de GoogleNet 39
Figure II.16 : les couches de l'architecture Xception 40
Figure II.17 : Raccourcis entre les blocs de convolution 40
Figure II.18 : Les couches de vgg16 41
Figure II.19 : Les avantages de l'apprentissage profond 42
Figure II.20 : Les étapes de la méthode DeepInsight 47
Figure II.21 : La matrice de confusion 51
Figure III.1 : Schéma des étapes de l'approche proposée 52
Figure III.2 : La séparation des données 56
Figure III.3 : L'emplacement des gènes de l'ensemble d'entrainement 59
Figure III.4: La matrice de densité pour chaque ensemble de données 60
Figure III.5: Ajustement des pixels 61
Figure III.6: Les images d'entrainement pour chaque ensemble de données 63
Figure III.7: Description du modèle 1D colon 64
Figure III.8: Description du modèle 1D prostate 66
Figure III.9: Description du modèle 1D Leukimia 67
Figure III.10: Description du modèle 1D CNS 68
Figure III.11: Les couches du modèle Vgg16 69
Figure III.12: Les couches du modèle xception 70
Figure III.13:schéma du Modèle 2D Xception 72
Figure III.14: Schéma du Modèle 2D vgg16 73
Figure III.15: Description du modèle 2D simple 74

Figure IV.1: Graph colon 1D 85
Figure IV.2 : Graph colon 2D 86
Figure IV.3: Graph prostate 1D 87
Figure IV.4: Graph prostate 2D 88
Figure IV.5: Graph leukimia 1D 89
Figure IV.6: Graph Leukimia 2D 90
Figure IV.7: Graph CNS 1D 91
Figure IV.8: Graph CNS 2D 92

Liste des tableaux :

Tableau I.1 : Présentation des données avec une matrices..... 20

Tableau III.1 : Description des ensembles de données 54

Tableau III.2 :Prétraitement des données..... 57

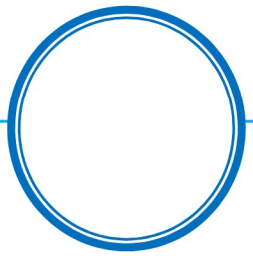
Tableau IV.1 : Colon dataset..... 80

Tableau IV.2: Prostate dataset..... 81

Tableau IV.3: Leukimia dataset 82

Tableau IV.4: CNS dataset..... 83

Tableau IV.5: Comparaison de l'accuracy entre les modèles 94



Introduction Générale

Introduction Générale :

De nos jours, L'évolution de l'informatique et des technologies de stockage a vu une explosion de grands volumes des données. Il est maintenant possible d'analyser des quantités énormes de données de dimension élevée grâce à l'augmentation des performances des ordinateurs. La bio-informatique est un domaine récent, qui fournit également des données de très grande dimension où il n'est pas rare d'avoir à manipuler plusieurs milliers d'attributs. Afin d'appliquer efficacement les techniques d'analyse ou d'apprentissage dans ce domaine, il est nécessaire de réduire la dimension des données en sélectionnant les attributs les plus intéressants pour le problème étudié. **[1]**

Actuellement, La technologie des puces à ADN permet de mesurer simultanément les niveaux d'expression de gènes au sein d'échantillons de tissus dans des conditions expérimentales données. L'utilisation de cette technique a permet de diagnostiquer plusieurs maladies dans le domaine de la Bio-informatique depuis les années 90, ou le professeur et chercheur ***Todd Golub*** a montré dans son article « ***Moleculare classification of cancer*** » que la puce à ADN est la seule technologie qui a permis de discriminer deux formes de leucémie dans le domaine oncologique. Et comme les données analysées présentent plusieurs milliers d'attributs, parmi les quelques 7000 gènes testés sur les puces pour cette expérience, un petit nombre de gènes (environ 50) apparaît comme très important pour la reconnaissance des deux formes de la maladie. Donc, il est nécessaire de proposer des méthodes innovantes pour la réduction de dimension et la sélection des gènes les plus pertinents pour accomplir la tâche de classification. **[2]**

Dans le domaine informatique, Les algorithmes populaires d'analyse des données non structurées sont basés sur la conception de réseaux de neurones, plus précisément sur l'apprentissage profond. Ce dernier est une branche de l'apprentissage automatique qui utilise une architecture basée sur les réseaux de neurones pour construire des modèles sophistiqués capables de comprendre des données complexes. Cette capacité a permis aux algorithmes DL de surpasser les algorithmes traditionnels dans plusieurs domaines tels que la vision par ordinateur, la classification d'images et la reconnaissance vocale. **[3]**

La problématique :

Selon l'organisation mondiale de la santé, Le cancer est la deuxième principale cause de décès dans le monde après les maladies cardiovasculaire, **environ 10 millions de morts par an à l'échelle mondiale**, **[4]** et généralement un décès sur un totale de six décès est dû à un cancer. Cependant, L'identification précoce et la prédiction d'un type de cancer sont très importantes pour la santé des patients, le traitement sera plus efficace et on pourra sauver plusieurs vies.

Dans le cadre de l'identification du cancer, La technologie des puces à ADN a permis d'améliorer considérablement l'observation précise de milliers de gènes du cancer grâce aux données d'expression génétique, cependant le grand nombre des caractéristiques (gènes) et la petite taille d'échantillons (personnes) dans une puces ADN peut provoquer une sélection des gènes bruyante et non informative pour la détection des cellule cancéreuse et la classification de leur type avec une faible précision. **[5]**

De nombreuses techniques d'apprentissage automatique ont obtenu une bonne précision dans la classification du cancer. Cependant, ces approches présentent encore des problèmes sur les données génétiques obtenues à partir des puces ADN à cause de leur grande complexité. L'incapacité de certains algorithmes ML dans l'utilisation des données non structurées a limité leur utilité dans la classification du cancer. Les techniques modernes basées sur l'apprentissage profond se sont avérées très réussi et très adaptés à traiter une variété de données structurées et non structurées comprenant des images, du son, des vidéos, du texte et des données relatives à des maladies, Cette capacité a permis aux algorithmes d'apprentissage profond de jouer un rôle important dans la détection précoce du cancer par l'identification et la classification des gènes informatifs. Ces gènes peuvent ensuite être utilisés pour prédire les classes des échantillons à tester. **[3]**

Les objectifs :

L'objectif principal de notre travail est de réaliser un modèle performant et puissant qui consiste à faire une classification des données génétiques afin de prédire la classe de nouveaux exemples (Tumeur, Normal) en s'appuyant sur l'apprentissage profond.

Nous proposons à travers cette thèse une implémentation d'un modèle CNN qui est un algorithme très utilisé dans le domaine de la bio-informatique grâce à sa capacité de faire une classification précise. Cette implémentation sera développée sur deux dimensions différentes (1D et 2D), avec l'utilisation de la méthode DeepInsight qui sert à transformer les expressions génétiques de chaque échantillon en une image correspondante.

Afin d'évaluer notre méthode, nous allons réaliser une série d'expérimentations sur des bases confirmant l'intérêt d'utiliser cette méthode dans la sélection des gènes.

Organisation de la thèse :

Cette thèse est composée de quatre chapitres dont nous donnons une brève description dans les lignes suivantes :

Le Premier chapitre :

On va commencer par une brève présentation du domaine de la bio-informatique, dont on va citer le principe de ce domaine et ses principaux champs d'applications, nous présenterons également la technologie des puces à ADN, en se focalisant sur leurs principes de fonctionnement, leurs étapes d'analyse et les domaines d'utilisation de cette technologie. Pour finir nous parlerons de l'utilisation des puces à ADN en oncologie et leur apport dans ce domaine.

Le deuxième chapitre :

On va consacrer cette partie pour l'apprentissage profond et ses différentes architectures, commençons par une présentation de ce domaine, ses différentes classifications (supervisé, semi supervisé et non supervisé), ainsi que ses différentes architectures. Puis nous présentons ses divers domaines d'utilisation et enfin on va montrer aussi les mesures d'évaluation des modèles.

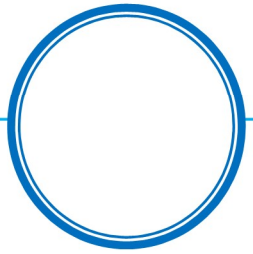
Troisième chapitre :

Dans ce chapitre, nous allons présenter notre approche proposée pour la détection du cancer, en se basant sur des jeux de donnée qui contiennent des informations génétiques obtenues à partir des puces à ADN, dont on va décrire les différentes étapes et méthodes suivie pour la construction de notre modèle qui sera basé sur l'architecture convolutive CNN, et ce dernier sera appliquer en deux exemplaires « dimension un 1D et dimension deux 2D ».

Quatrième chapitre :

Finalement, pour le dernier chapitre on va présenter notre environnement de travail, ainsi que les outils utilisés et on discutera nos résultats obtenus à partir de l'approche proposé dans le chapitre précédent.

Une conclusion générale terminera notre thèse, qui va contenir les perspectives ouvertes de ce travail de recherche.



CHAPITRE I

Technologie des puces à ADN

1 Introduction :

La technologie des puces à ADN est une technique qui a évolué depuis leur apparition dans les années 1990 pour devenir capable de tester actuellement plusieurs millions de marqueurs en une seule expérience. Leur usage en cancérologie a permis d'analyser différentes collections de tumeurs humaines avec le support de plateformes dédiées.

Dans ce chapitre on va présenter le domaine de la bio-informatique en se focalisant sur les principes, les concepts et les différentes étapes d'analyses en utilisant la technologie des puces à ADN, ainsi que l'apport de cette technique dans les problèmes cancérologique.

2 La bio-informatique:

2.1 Définition :

La bio-informatique est une discipline à l'interface de la biologie et de l'informatique, qui recouvre l'ensemble des technologies et des méthodes permettant de collecter, stocker, analyser et interpréter les données biologiques, elle est désignée sous deux aspects :

- ***Le développement d'infrastructures et d'outils « Bioinformatics » :*** c'est des systèmes de stockages de données et de visualisation, elle se rapproche plus d'un travail d'ingénierie en infrastructure et logiciel.
- ***Le développement et la mise en œuvre des méthodes mathématiques et informatiques « computational biology » :*** c'est une discipline scientifique qui consiste à modéliser et à analyser des systèmes biologiques par des modèles informatiques, en empruntant d'ailleurs de nombreuses approches aux mathématiques et à la physique.[6]

2.2 Le principe de la bio-informatique et ses champs d'application :

Le domaine de la bio-informatique fournit des bases de données centrales, accessibles mondialement, qui permettent aux scientifiques et aux chercheurs de présenter, rechercher et analyser une information biologique à travers l'informatique. Elle propose des logiciels de stockages et de traitement d'une grande quantité de données biologiques, ainsi que des outils de modélisation, visualisation, exploration et de l'interprétation de ces données.

La bio-informatique est constituée par l'ensemble des concepts et des techniques nécessaires à l'interprétation informatique de l'information biologique :

- **La *bio-informatique des séquences :*** elle traite l'analyse de données issues de l'information génétique contenue dans la séquence de l'ADN. Cette branche s'intéresse en particulier à l'identification des ressemblances entre les séquences, à l'identification des gènes ou de régions biologiquement pertinentes dans l'ADN en se basant sur l'enchaînement de leurs composants élémentaires (nucléotides, acides aminés).
- ***La bio-informatique structurale :*** elle traite la reconstruction, la prédiction ou l'analyse de la structure 3D ou du repliement des macromolécules biologiques (protéines, acides nucléiques), au moyen d'outils informatiques.
- ***La bio-informatique des réseaux :*** elle s'intéresse aux interactions entre gènes, protéines, cellules, organismes, en essayant d'analyser et de modéliser les comportements collectifs des ensembles élémentaires du vivant. [7]

Grâce à ces nouvelles connaissances, les chercheurs ont la possibilité de faire de nouvelles découvertes scientifiques. Des découvertes qui peuvent par exemple améliorer la qualité de vie de personnes malades grâce à la mise en place de nouveaux traitements médicaux plus efficaces.

3 La technologie des puces à ADN :

3.1 Définition :

Une puce à ADN est un ensemble de molécules d'ADN fixées en rangées ordonnées sur une petite surface qui peut être du verre, du silicium ou du plastique. Cette biotechnologie récente permet d'analyser le niveau d'expression des gènes dans une cellule, un tissu, un organe, un organisme ou encore un mélange complexe, à un moment donné et dans un état donné par rapport à un échantillon de référence.[8]

3.2 Le principe de fonctionnement :

La technologie des puces à ADN (ou « DNA microarrays » en anglais) a été initialement mise au point au milieu des années 1990, suite au séquençage des premiers génomes, afin d'étudier simultanément l'expression de tous les gènes d'un organisme. Son concept repose sur un procédé multidisciplinaire qui intègre la biologie, la chimie des acides nucléiques, les nanotechnologies, l'analyse d'images et la bio-informatique.

Une puce à ADN est une technologie miniaturisée de biologie moléculaire constitué d'un support solide, souvent une lame de verre, sur lequel sont fixés jusqu'à plusieurs millions de fragments ADN simple brin appelés sondes. Leur principe est alors basé sur la capacité de reconnaissance et d'hybridation entre les sondes et leurs cibles simple brin dans l'échantillon à analyser.[9]

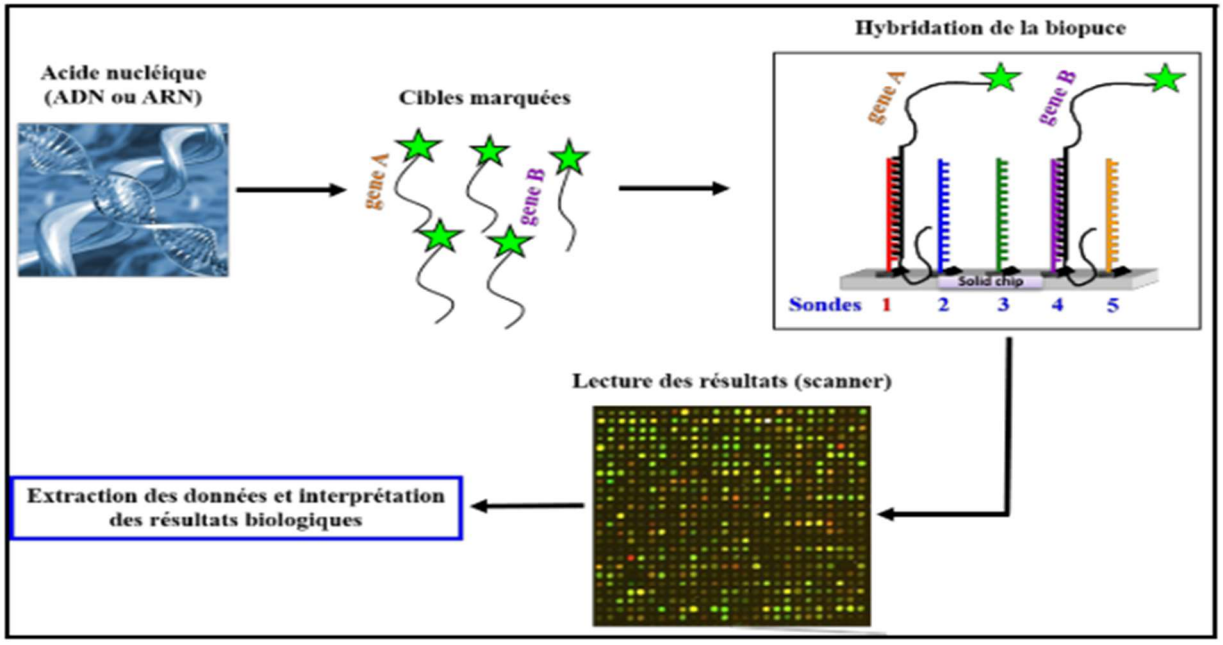


Figure I.1: principe des puces ADN

Les sondes spécifiques fixées sur le support solide de la bio-puce s'hybrident spécifiquement avec les cibles par complémentarité des bases. Les cibles extraites des échantillons biologiques à analyser sont marquées à l'aide d'un fluorochrome.

Après, l'étape d'hybridation des cibles et la lecture de la bio-puce avec un scanner, l'image obtenue est alors analysée permettant les interprétations biologiques.

3.3 Les phases d'analyse d'une puce à ADN :

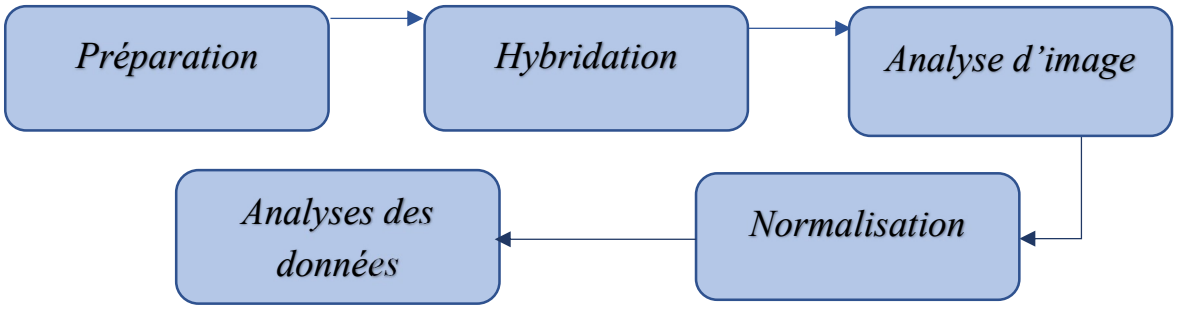


Figure I.2: étapes des analyses d'une puce à ADN

3.3.1 La préparation de la cible et l'hybridation :

Pour comparer les niveaux d'expression dans deux échantillons biologiques ou deux conditions (référence et pathologique) on réalisera les étapes suivantes :

La première étape : c'est la préparation du génome dans ces deux échantillons c'est-à-dire l'extraction des ARNm d'un échantillon biologique à analyser.

La deuxième étape : consiste à marquer les deux échantillons par des substances fluorescentes (Cy3 et Cy5) c'est-à-dire qu'une culture est marquée avec un fluorochrome vert, tandis que la seconde est marquée avec un fluorochrome rouge.[1]

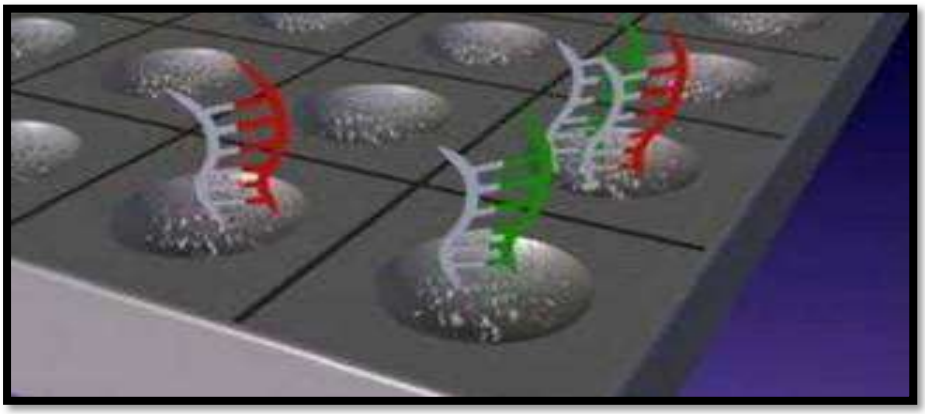


Figure I.3: hybridation d'une puce à ADN

Le processus d'hybridation qui consiste à mélanger et placés les ADN marqués sur les sondes de puces à ADN, est ensuite réaliser dans une station fluidique (four) dans une durée de

10 à 17 heures et a une température de 60 degrés (le fragment d'ADN simple brin peut reconnait son brin d'ADN complémentaire à cette température)

3.3.2 L'acquisition et analyse des images :

Suite à l'hybridation, L'obtention des images sera réalisée par la lecture des puces sur des scanners de haute précision, adaptés aux marqueurs utilisés (Cy3 et Cy5). Le procédé de détection combine deux lasers, pour exciter les fluorochromes.

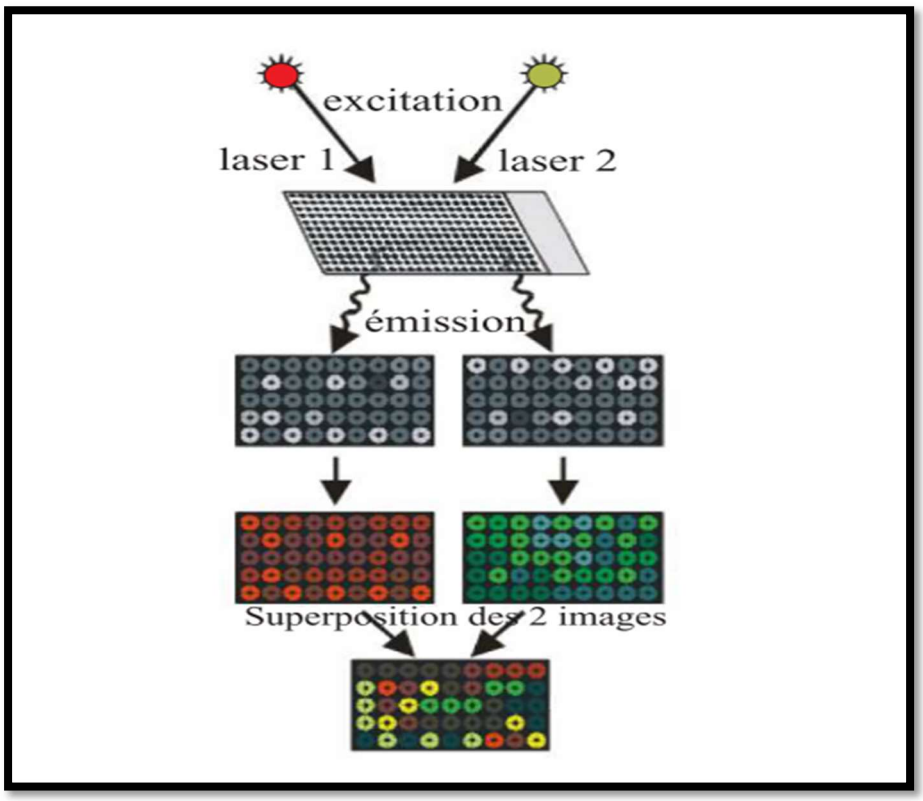


Figure I.4: acquisition d'image dans une analyse par puce à ADN

On obtient alors deux images dont le niveau de gris représente l'intensité de la fluorescence lue. Si on remplace les niveaux de gris par des niveaux de vert pour la première image et des niveaux de rouge pour la seconde, on obtient en les superposant une image en fausses couleurs composées de spots allant du vert au rouge quand un des fluorophores domine, en passant par le jaune (même intensité pour les deux fluorophores). Le noir symbolise l'absence de signal.

Les images sont traitées par des logiciels d'analyse qui permettent de mesurer la fluorescence de chaque spot sur la lame (estimant les niveaux d'expression pour chacun des gènes présents sur la puce), mais aussi de relier chaque sonde à l'annotation correspondante

(nom de gêne, numéro de l'ADNc utilisé, séquence de l'oligonucléotide, etc.). Ainsi, pour chaque spot, l'intensité de chaque marqueur est calculée puis comparée au bruit de fond.[1]

3.3.3 La normalisation des données :

Cette étape consiste à ajuster l'intensité globale des images acquises sur chacun des deux canaux rouge et vert, de manière à corriger les différences systématiques entre les échantillons sur la même lame, qui ne représentent pas de variations biologiques entre les échantillons et qui tendent à déséquilibrer le signal de l'un des canaux par rapport à l'autre.

La normalisation est effectuée à partir de toutes les sondes présentes sur le support pour éliminer les différences entre les différentes puces liées aux variations de quantité de départ, aux biais de marquage ou d'hybridation et aux variations du bruit de fond.[1]

3.3.4 La Présentation des données de puces à ADN :

Les données recueillies après la normalisation pour l'étude d'un problème donné sont regroupées sous forme d'une matrice avec une ligne par couple (gène, sonde) et une colonne par échantillon [1]. Chaque valeur de mij est la mesure du niveau d'expression du i−ème gène dans le j−ème échantillon, où i = 1, . . M et j = 1, . . N.

Gènes	**Echantillon 1**	**Echantillon 2**	**. . . . .**	**Echantillon N**
Gène 1	m11	m12	. . . . .	m1N
Gène 2	m21	m22	. . . . .	m2N
Gène 3	m31	m32	. . . . .	m2N
. . .	. . .	. . .	. . .	. . .
Gène M	mM1		. . . . .	mMN

Tableau I.1: Présentation des donnée avec une matrice

4 Les types des puces d'ADN :

On distingue plusieurs types de puces selon la densité des spots, le mode de fabrication, la nature des fragments fixés à la surface, les méthodes d'hybridation et l'objectif du projet **[10]**

4.1 Les macro-array ou filtre à haute densité :

Principe : un dépôt direct de l'ADN sur le support avec une seule condition expérimentale par puce.

Nombre de fragments déposé : 2400 fragments d'ADN.

Marquage : marquage radioactive et le criblage par excès de cibles.

4.2 Les micro-array :

Principe : un dépôt direct de l'ADN sur le support avec deux conditions expérimentales par puce.

Nombre de fragments déposé : 10000 fragments d'ADN.

Marquage : par fluorescence

4.3 La puce à oligonucléotides :

Principe : une synthèse au milieu des sondes Oligo nucléotidiques sur une surface solide, avec une condition expérimentale par puce.

Nombre de fragments déposé : jusqu'à 4,2 millions oligonucléotides.

Marquage : par fluorescence.

5 L'utilisation des puces à ADN :

Les puces à ADN apportent une solution innovante aux anciens problèmes de la détection, l'identification et au typage des bactéries dans un échantillon, Les scientifiques peuvent maintenant comprendre les fonctions des gènes de n'importe quel organisme en dépistant des milliers de gènes et de leurs produits en même temps.

Cette innovante technique est employée surtout dans le domaine médical pour le traitement des maladies cardiovasculaire, infectieuses et les différentes inflammatoires organiques. Leur application peut être classée par catégorie dans deux types principaux :

5.1 Le diagnostic de la maladie :

La puce est employée pour comparer les tissus ou les cellules malades aux tissus sains, afin de détecter les caractéristiques et les gènes responsables d'une maladie particulière, elle est très utilisée pour connaître la condition d'une maladie, le type d'une tumeur ainsi que différente autre factorisation qui permet le bon diagnostic d'un certain nombre de maladies, spécialement le cancer.

5.2 La découverte des médicaments :

Suite au diagnostic, les puces peuvent être employées pour l'étude des anticorps, ainsi que les microorganismes des bactéries et des virus détectés, afin de découvrir des antibiotiques et des vaccins conforme à la maladie. **[11]**

6 Les puces à ADN et le cancer :

6.1 La maladie du cancer :

Le cancer est une pathologie caractérisée par la présence d'une ou plusieurs tumeurs malignes formées à partir d'une transformation cellulaire tumorale qui se traduit notamment par une perte de contrôle du cycle cellulaire ou l'instabilité génétique d'une cellule initialement normale (anomalies cytogénétiques). **[12]**

Cette maladie est considérer comme la deuxième principale cause de décès dans le monde après les maladies cardiovasculaire par les statistiques de l'organisation de la santé mondiale « **OMS** », **environ 10 millions de morts par an à l'échelle mondiale**, et généralement un décès sur six décès totaux est dû à un cancer **[4]**

En 2018**,** les études épidémiologiques relatives au cancer en Algérie ont révélé que le pays enregistrait 41 870 nouveaux cas de cancer par an, dont 25 122 chez la femme et 16 748 chez l'homme. L'étude a également révélé que les projections à 2025 donnaient une augmentation du nombre des cas pouvant atteindre 70 556 nouveaux cas par an. **[13]**

Dans ces dernières années le nombre des gènes cancéreuse est en rapide augmentation et les méthodes anciennes utilisées depuis le début de la mise en œuvre du génie génétique pour leur exploitation ne suffisent pas à traiter ce grand nombre d'informations, en effet le séquençage des génomes fournit aux expérimentateurs un nombre important et considérable de séquences qui peuvent être utilisées comme sondes génétiques. **[14]**

6.2 L'apport des puces à ADN en cancérologie :

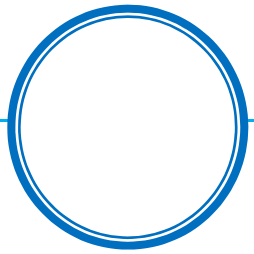
L'identification précoce et la prédiction d'un type de cancer sont très importantes pour la santé du patient. Récemment, les données d'expression génétique des puces d'ADN ont été utilisées pour un diagnostic efficace et précoce du cancer.

Les puces à ADN en cancérologie sont des outils d'une nouvelle discipline de recherche l'oncogénétique. Ces puces permettent de détecter à l'échelle du génome entier les lésions de l'ADN (amplifications, délétions et autres mutations), de préciser les interactions protéines-ADN dans les régions promotrices et quantifient les niveaux d'ARN (puces d'expression, puces pour microRNA). L'analyse des données générées nécessite des moyens statistiques spécifiques aux grandes bases de données et est devenue une discipline à part entière de la bio-informatique.

L'objectif de ces analyses est de mettre en évidence de nouvelles cibles thérapeutiques et de fournir un nombre limité de marqueurs biologiques pertinents pour la classification ou le pronostic des cancers. Ces marqueurs sont accessibles aux techniques de routine des laboratoires de biologie clinique (PCR, immun dosages...) ou d'anatomie pathologie (immunohistochimie). Néanmoins la réduction des couts de production des puces rend possible aujourd'hui l'utilisation de puces dédiées dans les laboratoires d'analyses pour détecter des signatures moléculaires basées sur 50 à 100 gènes ou pour la détection rapide de mutations de suppresseurs de tumeurs. **[15]**

7 Conclusion :

Dans cette partie on a parlé en général de la bio-informatique et ces différents champs d'application, on se focalisant sur la technique des puces ADN en cancérologie, leurs principes de fonctionnement et leurs différentes étapes d'analyse.



Chapitre II

Apprentissage profond

1 Introduction :

Depuis 2012 les algorithmes basés sur l'apprentissage profond semblent prêt à résoudre des différents problèmes d'une manière précise et rapide grâce à l'avancée des performances de calcul d'ordinateurs, dans plusieurs domaines : la reconnaissance faciale, l'autonomie des voitures ainsi que la recherche dans le domaine médicale. Dans ce chapitre en va présenter les différentes architectures d'apprentissage profond, leur classification, et ses divers domaines d'utilisation, en montrons aussi les mesures d'évaluation de ces modèles.

2 Définition de l'apprentissage profond:

L'apprentissage profond ou **le Deep Learning** en anglais, est une forme d'intelligence artificielle et un dérivé de l'apprentissage automatique, fondée sur des approches mathématiques qui s'appuient sur des réseaux de neurones artificiels s'inspirant du cerveau humain. Les origines du Deep Learning remontent aux origines de Machine Learning dans les années 1950, où le mathématicien britannique **Alan Turing** s'intéresse aux machines qui peuvent comprendre et apprendre**. [16]**

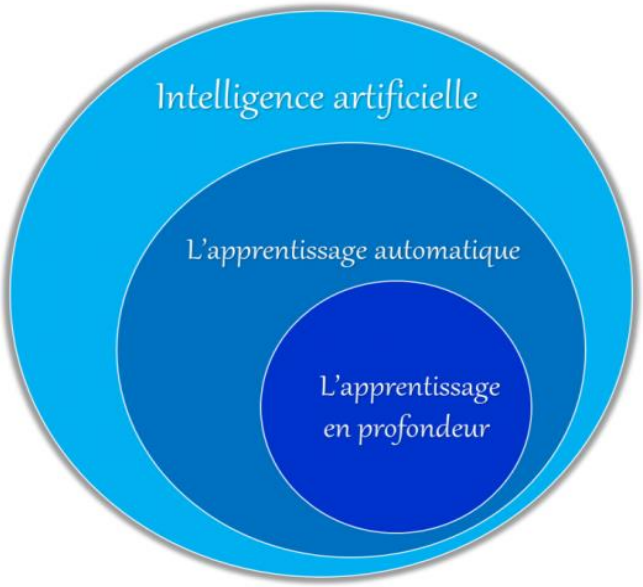


Figure II.1 : Les sous-bronches de l'intelligence artificielle

Contrairement aux algorithmes classiques de Machine Learning dont la capacité d'apprentissage est limitée quelle que soit la quantité de données acquise, Les systèmes de Deep Learning n'ont pas réellement de limite en termes de capacité d'apprentissage. Plus ils reçoivent de données sur lesquels s'entraîner, plus ces systèmes améliorent leurs performances.**[17]**

Une fois que les machines ont acquis assez d'expérience grâce au Deep Learning, elles peuvent être utilisées pour des tâches spécifiques et elles peuvent établir des relations et identifier des causes qui restent indétectables par l'homme.

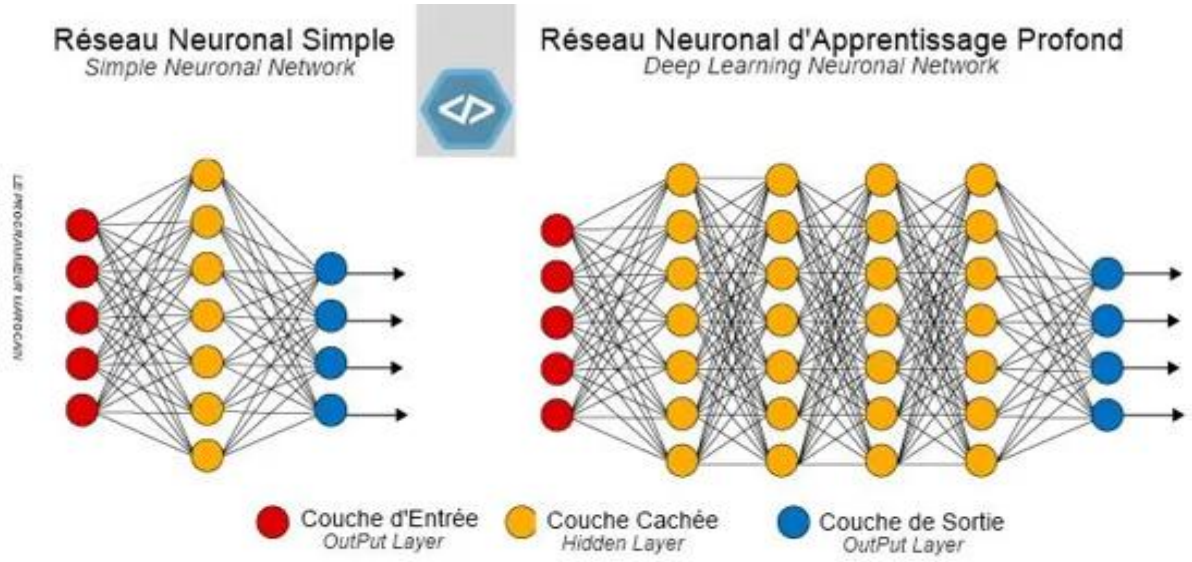


Figure II.2 : les différentes couches d'un réseau neuronal

Un réseau de neurones d'apprentissage profond est constitué de trois couches principales :

- ***La couche d'entrée :*** *(Input layer)* cette couche reçoit les données d'entrée.
- ***La couche cachée :*** *(Hidden layer)* cette couche effectue des calculs mathématiques.
- ***La couche de sortie :*** *(output layer)* cette couche renvoie les données de sorties.

2.1 La classification de l'apprentissage profond :

L'apprentissage profond peut être classé en trois classes principales en fonction des objectifs pour lesquels elles ont été conçues :

2.1.1 Apprentissage profond supervisé :

L'apprentissage supervisé est une technique d'apprentissage qui utilise des données étiquetées, après une formation réussite le modèle pourra obtenir les bonnes réponses aux questions de l'environnement. Il existe différents apprentissages supervisés pour un apprentissage profond : les réseaux de neurones profonds *(DNN),* les réseaux de neurones convolutifs *(CNN),* les réseaux de neurones récurrents *(RNN),* ainsi que la mémoire a long et court terme *(LSTM)* et les unités récurrentes fermées *(GRU).* **[22]**

L'entraînement par l'apprentissage supervisé est facile et efficaces grâce aux données déjà étiquetées. Les labels permettent de valider le modèle en le testant sur des données étiquetées et en comparant les résultats prédits et les sorties réelles.

2.1.2 Apprentissage profond non supervisé :

Les systèmes d'apprentissage non supervisés sont ceux qui peuvent faire une prédiction sans la présence d'étiquettes de données. Dans ce cas, le modèle apprend la représentation interne ou des fonctionnalités importantes pour découvrir des relations ou une structure inconnues dans les données d'entrée. Souvent, le regroupement, la réduction de la dimensionnalité et les techniques génératives sont considérés comme des approches d'apprentissage non supervisé.

Les auto-encodeurs, machines Boltzmann restreintes *(RBM)*, Long short-terme memory *(LSTM*) sont également utilisés pour l'apprentissage non supervisé dans de nombreux domaines d'application. **[22]**

2.1.3 Apprentissage profond semi supervisé :

L'apprentissage semi-supervisé est un apprentissage basé sur des jeux de données partiellement étiquetés, son principe est soit, de modifier, soit de réorganiser les hypothèses effectuées sur le modèle à partir des données d'apprentissage. **[22]**

2.2 Les domaines d'applications de l'apprentissage profond :

De nos jours l'apprentissage profond nous aide toujours à découvrir et à réaliser des nouvelles avancées technologiques, Les champs d'applications du Deep Learning sont très nombreux, On cite parmi eux les domaines suivants :

2.2.1 Le Robotique :

L'évolution de l'intelligence artificielle et précisément le Deep Learning permettra aux robots du futur de devenir des meilleurs assistants pour l'homme. Ils ne seront pas uniquement utilisés pour comprendre les questions et y répondre, comme c'est le cas actuellement pour certains robots**.**



Figure II.3 : l'apprentissage profond en robotique

2.2.2 L'agriculture :

Le Deep Learning permet déjà aux agriculteurs de déployer des équipements capables de repérer et de différencier les plantes cultivées et les mauvaises herbes. Cette capacité permet aux machines de désherbage d'épandre des herbicides exclusivement sur les mauvaises herbes tout en épargnant les autres plantes.

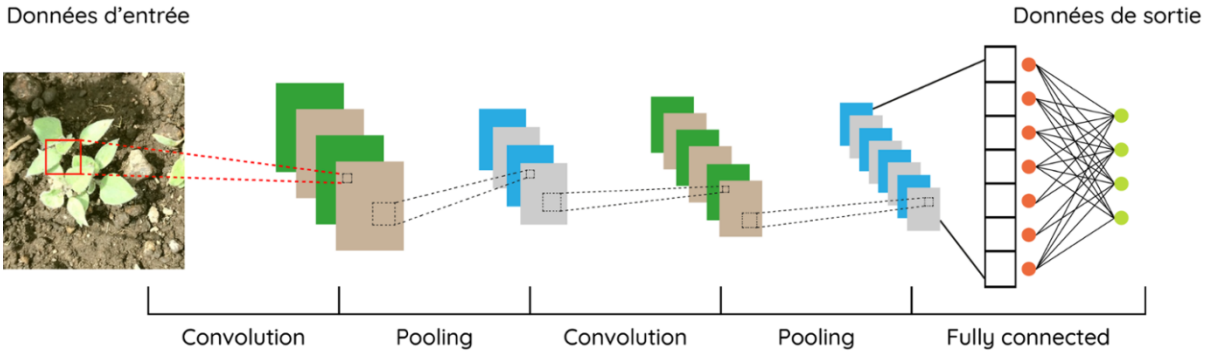


Figure II.4 : l'apprentissage profond en agriculture

2.2.3 La santé :

L'intelligence artificielle et en particulier l'apprentissage profond vise à créer des logiciels qui pourraient remplacer l'intelligence humaine dans le domaine de la santé afin d'aider les professionnels et les chercheurs à améliorer leur efficacité, leur productivité et leur constance dans la qualité des soins apportés aux patients.



Figure II.5 : l'apprentissage profond et la santé

2.2.4 La détection de malwares ou de fraudes :

Les algorithmes de Deep Learning sont particulièrement adaptés à la lutte contre la fraude, en offrant comme principal avantage l'adaptation rapide a d'autres types de documents ou parcours, en plus le secteur bancaire est au centre des flux de big data, qu'ils soient financiers ou non et l'application de Deep Learning permet aux banques d'améliorer grandement leurs techniques de traitement de données, et indirectement de multiplier les bénéfices.



Figure II.6 : l'apprentissage profond et la détection des malwares

2.3 Les architectures de l'apprentissage profond :

Il existe deux considérations majeures en Deep Learning : le traitement multicouches non linéaire et le formulaire d'apprentissage peut être supervisé ou non supervisé, les architectures populaires pour la création des modèles d'apprentissage sont :

2.3.1 L'auto-encoder :

C'est une sorte de réseau de neurones artificiels plus précisément, une sorte de perceptron qui a une couche cachée ou les entrées et les sorties sont identiques et similaires. L'architecture d'un auto-encodeur est constituée des couches de neurones de codage qui compriment les données d'origines d'entrée pour permettre d'avoir une représentation comprimée des données. Ainsi que des couches de décodage essayent de reconstruire les données d'origines à partir des données comprimées**.**

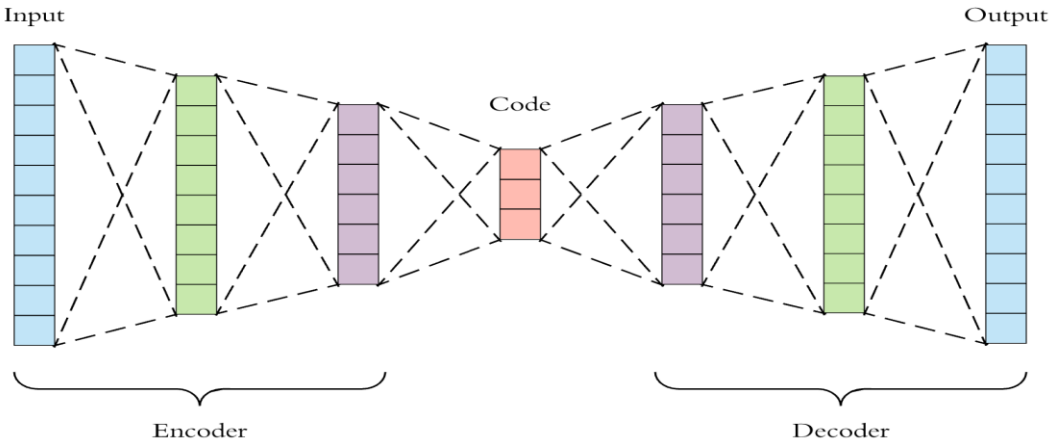


Figure II.7 : les parties d'un auto-encodeur

L'encodeur est constitué par un ensemble de couches de neurones, qui traitent les données afin de construire de nouvelles représentations dites "encodées". À leur tour, les couches de neurones du décodeur, reçoivent ces représentations et les traitent afin d'essayer de reconstruire les données de départ. Les différences entre les données reconstruites et les données initiales permettent de mesurer l'erreur commise par l'auto-encodeur. L'entraînement consiste à modifier les paramètres de l'auto-encodeur afin de réduire l'erreur de reconstruction mesurée sur les différents exemples du jeu de données. **[18]**

2.3.2 Réseau de neurones récurrent (Recurrent Neural Network) :

Les RNN sont des réseaux de neurones dans lesquels l'information peut se propager dans les deux sens, y compris des couches profondes aux premières couches, ces réseaux possèdent des connexions récurrentes au sens où elles conservent des informations grâce à leur mémoire qui mémorise toutes les informations sur ce qui a été calculer, ils peuvent prendre en compte à un instant t un certain nombre d'états passés.

Une couche de type RNN prend en entrée une séquence de tableau (une matrice de taille (T x M)) et retourne en sortie un vecteur de taille R qui est une représentation compressée de la totalité de la séquence de caractères.

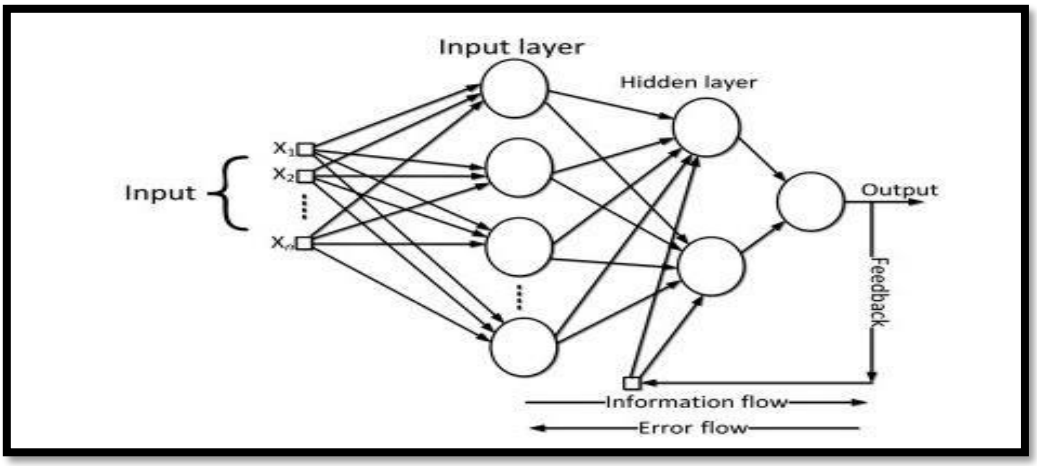


Figure II.8 : le réseau de neurones récurrent

Les RNN sont particulièrement adaptés aux applications faisant intervenir le contexte, et plus particulièrement au traitement des séquences temporelles comme l'apprentissage et la génération de signaux, Aussi il est utilisé pour les reconnaissances, en modélisation de comportement d'un grand nombre de types différents. **[19]**

2.3.3 Réseau de neurones récursif (Recursive Neural Network) :

Les modèles adaptatifs non-linéaires qui peuvent apprendre des informations détaillées et structurées sont appelés réseaux de neurones récursifs, ils sont efficaces dans le traitement du langage naturel pour l'apprentissage des séquences et des structures des arbres, principalement des phrases et des phrases basées sur l'incorporation de mots, ils sont que des connexions entre les neurones, ces derniers sont établis dans des cycles dirigés et ils comprennent une classe d'architectures qui peuvent fonctionner avec des entrées structurées.

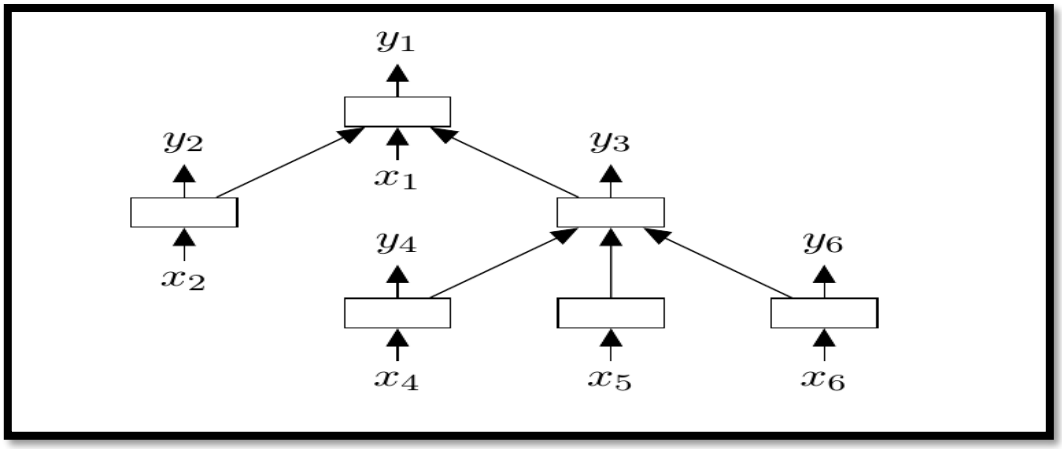


Figure II.9 : le réseau de neurones récursif

Ces modèles ne sont cependant pas encore universellement reconnus, La principale explication en est son ambiguïté sous-jacente. Non seulement parce qu'il s'agit de structures très complexes pour la recherche d'informations, mais aussi en raison d'une période d'apprentissage informatique coûteuse. **[20]**

2.3.4 Réseau de neurones convolutif (Convolutional Neural Network) :

Un réseau de neurone convolutifs (Convolutional Neural Network ou CNN en anglais) est un type particulier de réseau de neurones qui se base sur l'opération de convolution, il est particulièrement adapté à la reconnaissance d'images. Les CNNs se composent de deux parties, la première est appelée la partie convolutif et la deuxième est appelée la partie de classification, Ils se composent également de quatre types de couches : la couche de con- volution, la couche de pooling, la couche de correction (ReLU) et la couche fully-connected. La figure suivante montre une vue générale de CNN : **[21]**

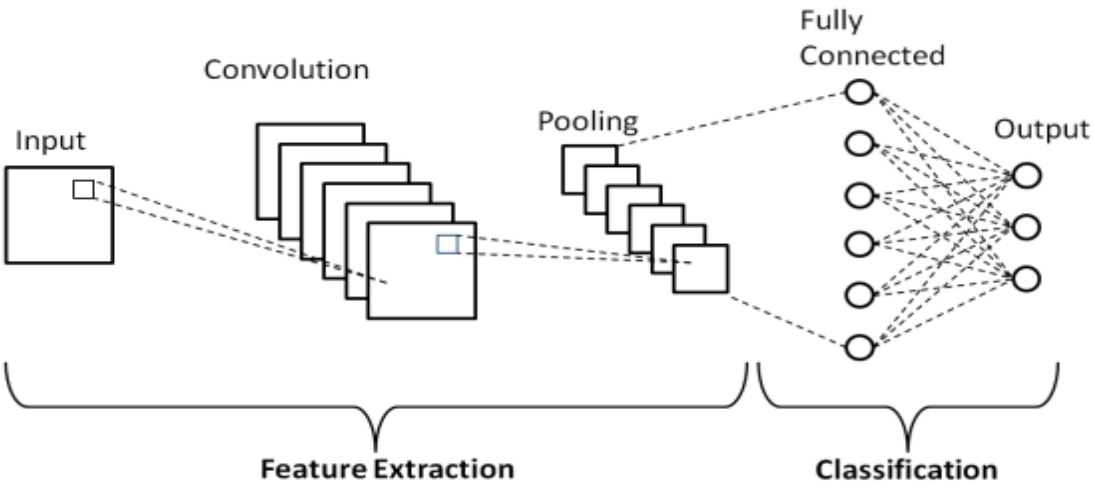


Figure II.10 : les couches d'un réseau de neurones convolutif

2.3.5 La partie d'extraction des caractéristiques :

Son objectif est d'extraire des caractéristiques propres à chaque image, elle est composée de trois couches essentielles :

2.3.5.1 Couche de convolution :

Cette couche est la composante clé du CNN, son but est l'extraction de caractéristiques des images les compressant de façon à réduire leur taille initiale. En résumé, l'image fournie en entrée passe à travers une succession de filtres, créant par la même occasion de nouvelles images appelées cartes de convolutions. Enfin, les cartes de convolutions obtenues sont concaténées dans un vecteur de caractéristiques appelé « code CNN ».

Elle est composée de trois paramètres essentiels :

- **La Profondeur de la couche *(Kernel size)*** : c'est le nombre de noyaux de convolution
- **Le pas *(stride)* :** il contrôle le chevauchement des champs récepteurs. Plus le pas est petit, plus les champs récepteurs se chevauchent et plus le volume de sortie sera grand.
- **La marge a 0 *(zéro padding)*** : Cette marge permet de contrôler la dimension spatiale du volume de sortie.

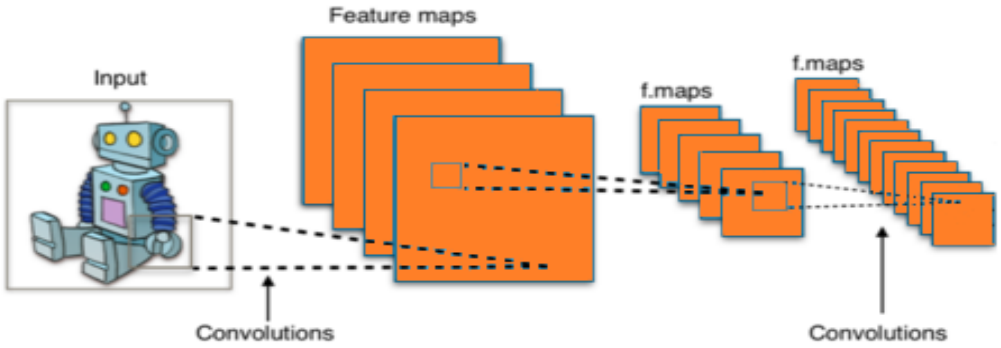


Figure II.11 : fonctionnement des couches convolutives

Des filtres sont appliqués à chaque image utilisée pour l'apprentissage à différentes résolutions, et la sortie de chaque image convolutée est utilisée comme entrée de la couche suivante. Dans la terminologie du réseau convolutif, le premier argument de la convolution est souvent appelé l'entrée (input) et le second argument comme noyau (kernel). La sortie est parfois appelée la carte des caractéristiques (feature map).

2.3.5.2 Couche de pooling :

Cette couche consiste à améliorer l'efficacité du réseau et réduire progressivement la taille spatiale des images en condition de préserver leurs caractéristiques importantes. Pour cela on découpe l'image en cellules régulières (des cellules carrées de petite taille pour ne pas perdre trop d'informations), puis on garde au sein de chaque cellule la valeur maximale (Max pooling) ou on calcule la moyenne des valeurs de la zone (average pooling), comme il est montré dans la figure suivante :

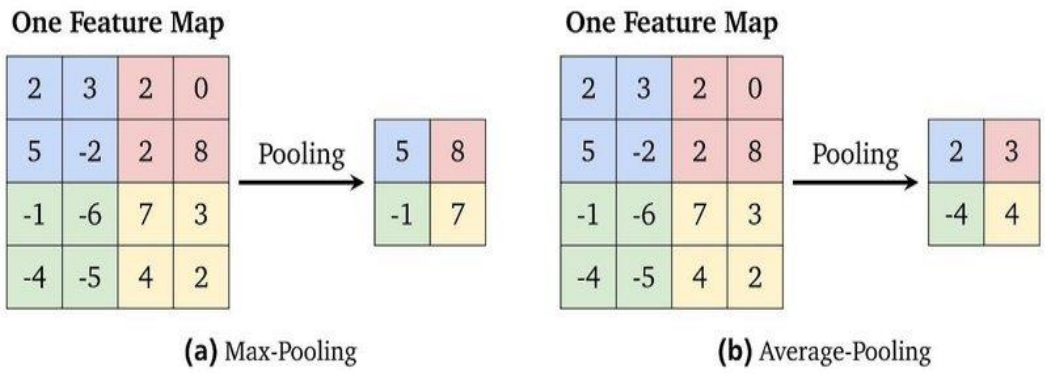


Figure II.12 : la différence entre max-pooling et average-pooling

Enfin, la couche de pooling permet de réduire le nombre de paramètres et de calculs dans le réseau, afin d'améliorer l'efficacité du réseau et d'éviter le sur-apprentissage.

2.3.5.3 Couche de correction ReLU :

Cette couche améliore l'efficacité du traitement en intercalant entre les couches de traitement une couche qui va opérer une fonction mathématique (fonction d'activation) sur les signaux de sortie. Il existe beaucoup de fonctions d'activations comme : L'unité linéaire rectifiée (Rectified Linear Unit en anglais), qui est une fonction d'activation très utilisée, elle consiste à remplacer les nombres négatifs des images filtrées par des zéros.

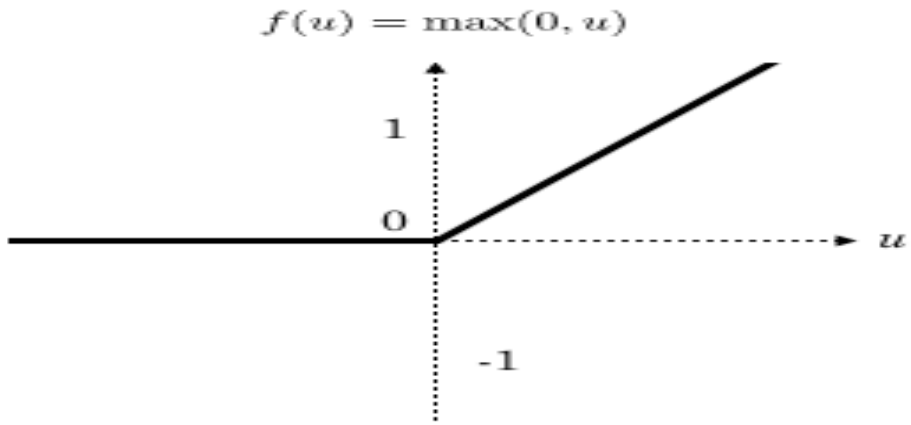


Figure II.13 : La fonction d'activation ReLU

2.3.6 La partie de la classification :

Le code CNN obtenu en sortie de la partie convolutive est fourni en entrée dans une deuxième partie de classification afin de combiner les caractéristiques du code CNN afin de classer l'image.

2.3.6.1 Couche entièrement connecté : (fully connected layer)

Cette couche est placée à la fin d'un réseau de neurone convolutif, elle est entièrement connectée à tous les neurones de sortie. Cette couche reçoit un vecteur en entrée, puis elle applique une combinaison linéaire et une fonction d'activation dont le but final est de classifier les images d'entrée. A la fin elle renvoi un vecteur de taille N ou N est correspondante au nombre de classe dans lequel chaque composante représente la probabilité pour l'entrée image d'appartenir à une classe.

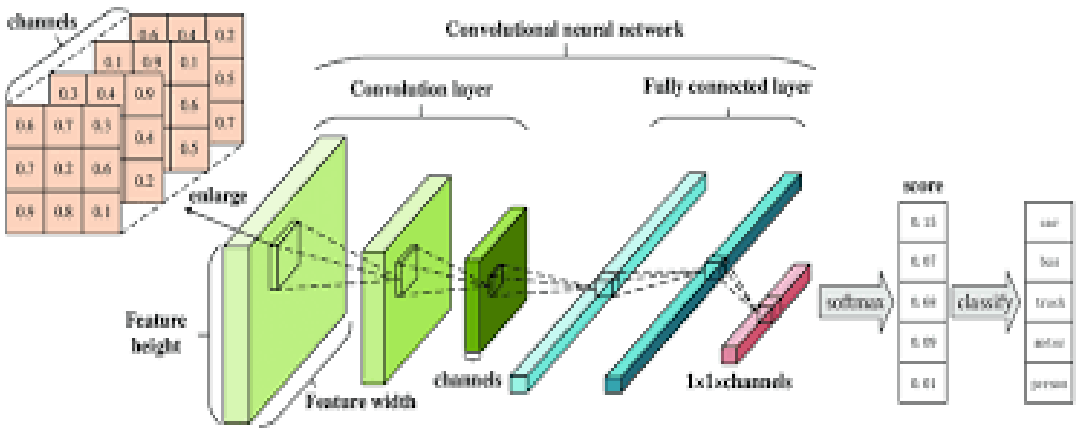


Figure II.14 : la couche fully connected et les fonctions d'activation

Les fonctions d'activation utilisée dans cette couche sont :

SoftMax : est utilisé pour la classification multiple dans le modèle de régression logistique. Elle est appliquée avec la formule mathématique suivante :

$$SoftMax\ (xi) = {Exp(xi)}/{\sum_{j=0} Exp(xj)}$$

Sigmoid : est utilisé pour la classification binaire dans le modèle de régression logistique. Elle est appliquée avec la formule mathématique suivante :

$$Sigmoid\ (x) = {1}/{1 + Exp^{-x}}$$

2.3.7 Exemples d'architectures du CNN :

Les réseaux de neurones convolutifs (CNN) se sont avérés très efficaces dans la classification d'images et pour d'autres types de données. Nous utilisons diverses architectures comme :

2.3.7.1 L'architecture GoogleNet : (inception V1)

Le réseau d'inception a était toujours l'un des principales découvertes dans le domaine de l'apprentissage profond et en particulier pour le CNN, jusqu'à présent il existe trois versions

de réseaux inception Version 1, 2,3. La première version (Inception V1) est entrée dans le domaine au sein de la compétition annuelle l'ILSVRC « ImageNet Large Scale Visual Recognition Challenge » en 2014 qui a été développé par une équipe de Google et suggéré sous le nom GoogleNet. **[22]**

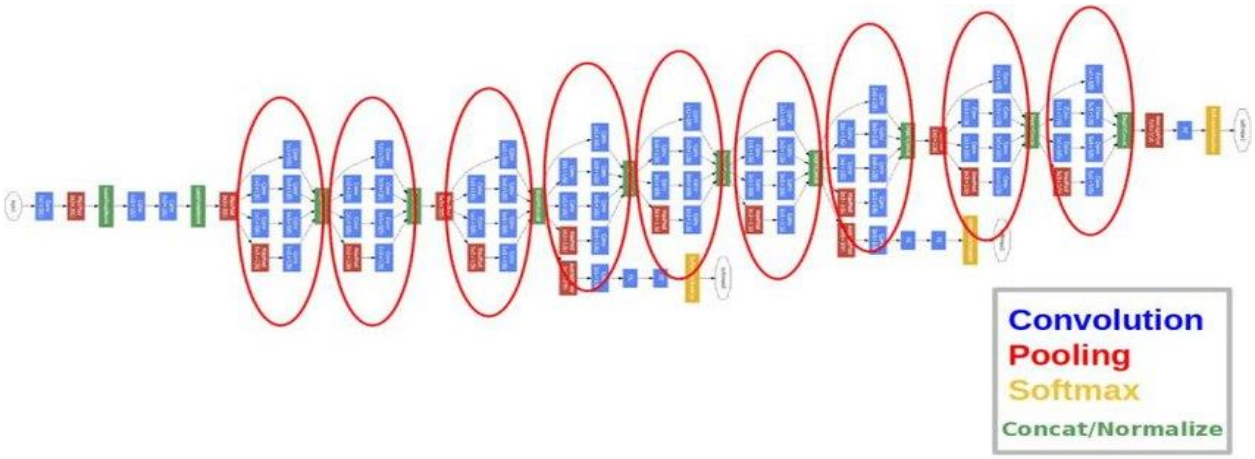


Figure 115 : les différentes couches de GoogleNet

L'architecture de GoogleNet comporte 22 couches profondes, avec 27 couches de mise en commun "maxpooling". Il y a 9 modules d'inception empilés linéairement au total dont leurs extrémités sont connectées à la couche de pooling moyenne globale. La figure suivante montre l'architecture complète de GoogleNet : **[22]**

2.3.7.2 L'architecture Xception :

C'est une version extrême d'inception ou (Extreme Inception) en anglais, qui a était développé par Google au sein de la compétition annuelle l'ILSVRC « ImageNet Large Scale Visual Recognition Challenge » en 2017.

Xception est une architecture de réseau neuronal convolutif entièrement basée sur une pile linéaire de couches convolutives séparables en profondeur avec des connexions résiduelles, elle se compose de blocs Depthwise Separable Convolution et Maxpooling, tous liés par des raccourcis.

Cette méthode repose sur deux points importants qui assurent son efficacité : **[23]**

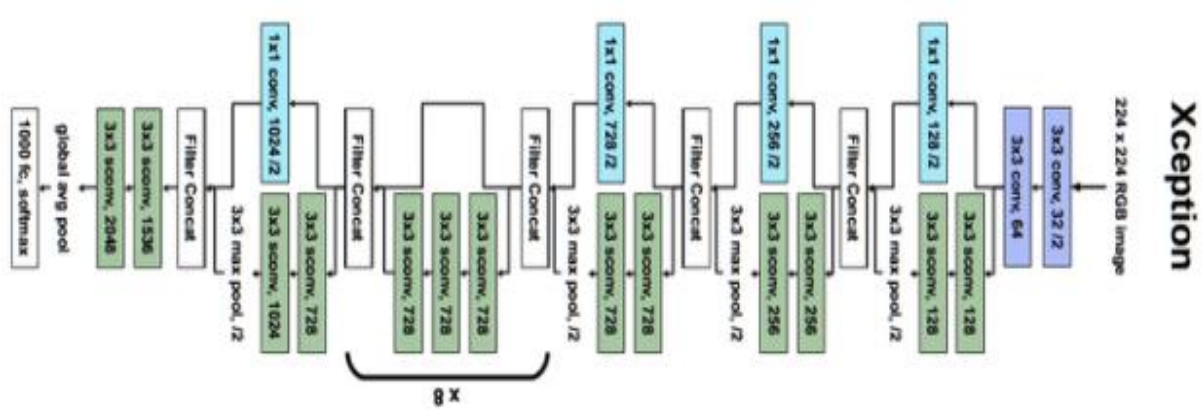


Figure II.16 : les couches de l'architecture Xception

- **Les convolutions séparables en profondeur** : sont des convolutions classiques qui sont censées être beaucoup plus efficaces en termes de temps de calcul, elles se divisent en deux étapes principales :
- **Convolution en profondeur** : (Depthwise Convolution) est une première étape dans laquelle au lieu d'appliquer une convolution de taille r x r x C, on applique une convolution de taille r x r x 1, on ne fait pas le calcul de convolution sur tous les canaux, mais seulement 1 par 1, on fait l'opération de convolution que pour 1 seul filtre de la convolution, pas pour N filtres d'eux. La spécificité de **XCeption** est que la Depthwise Convolution n'est pas suivie d'une Pointwise Convolution, mais l'ordre est inversé

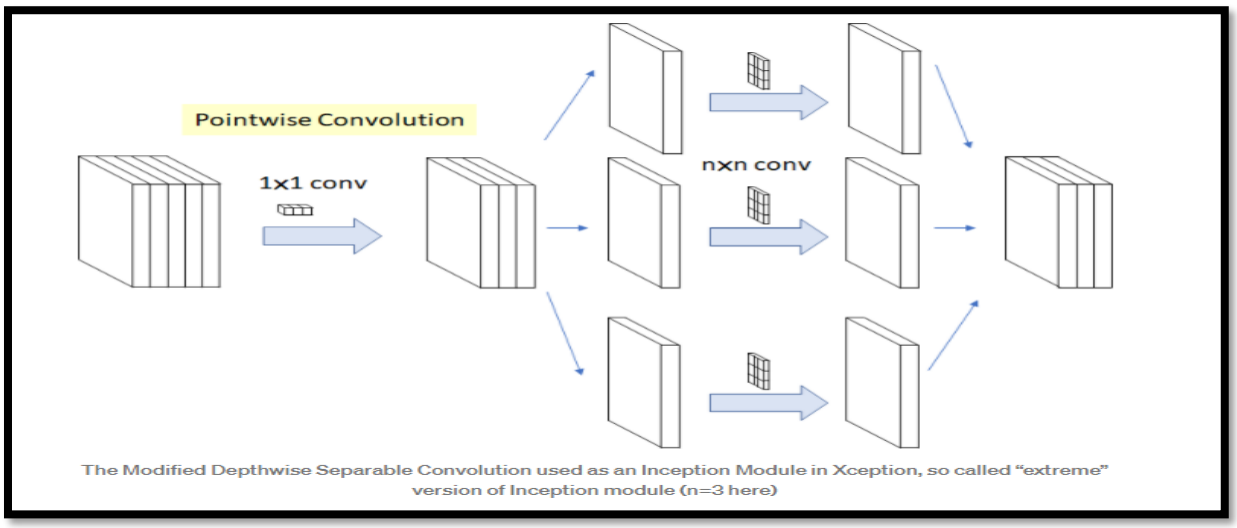


Figure II.17 : La structure de l'architecture Xception

2.3.7.3 L'architecture groupe de géométrie visuel « VGG » :

C'est une architecture de réseau de neurone basé sur le CNN, qui a été proposé et appliqué en 2014 dans le défi l'ILSVRC « ImageNet Large Scale Visual Recognition Challenge » par Karen Simonyan & Andrew Zisserman de l'Université d'Oxford, qui ont détaillé leur travail sur l'article « VERY DEEP CONVOLUTIONAL NETWORKS FOR LARGE-SCALE IMAGE RECOGNITION ». Le réseau a atteint une précision de 92,7 % dans les cinq premiers tests sur le jeu de données ImageNet. **[24]**

Le VGG16 est composé de 13 couches convolutionnelles, de 5 couches de max-pooling et de 3 couches entièrement connectées. Par conséquent, le nombre de couches ayant des paramètres réglables est de 16 (13 couches convolutionnelles et 3 couches entièrement connectées). C'est la raison pour laquelle le nom du modèle est VGG16.

Le nombre de filtres dans le premier bloc est de 64, puis ce nombre est doublé dans les blocs suivants jusqu'à atteindre 512. Ce modèle est complété par deux couches cachées entièrement connectées et une couche de sortie. **[24]**

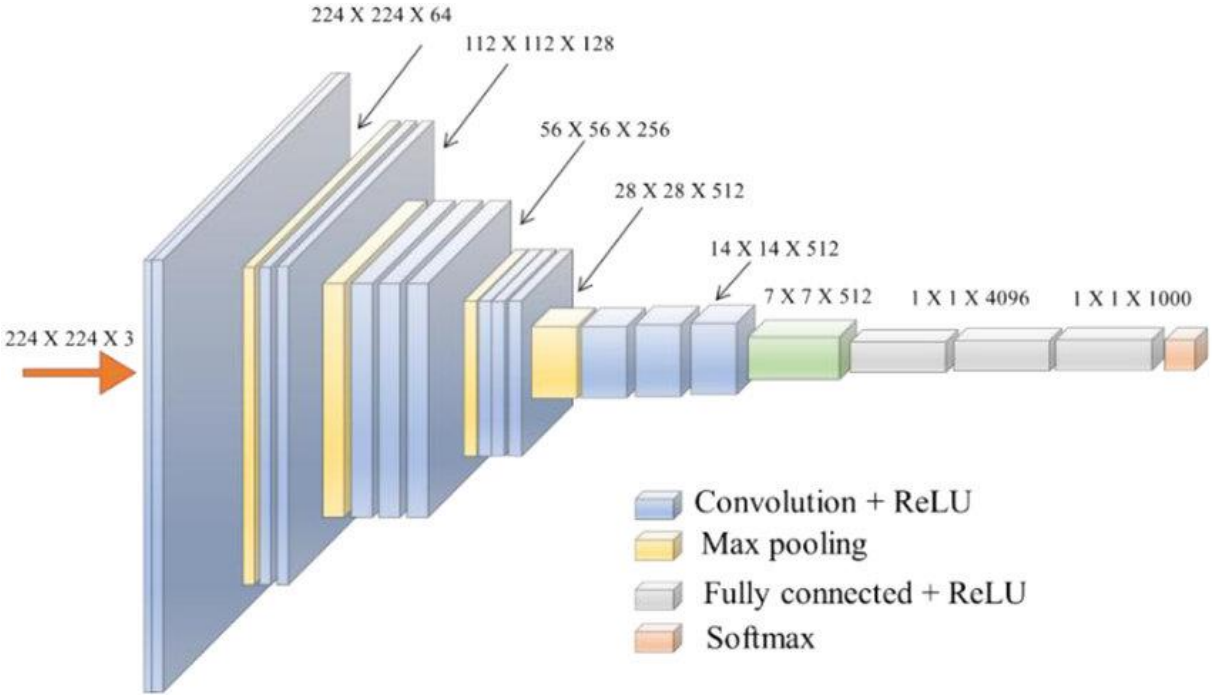


Figure II.18 : les couches de l'architecture Vgg16

L'entrée : Les dimensions d'entrée de l'architecture sont fixées à la taille de l'image, (244 × 244). Dans une étape de prétraitement, la valeur RVB moyenne est soustraite de chaque pixel d'une image.

Les couches convolutionnelles : Une fois le prétraitement terminé, les images sont transmises à une pile de couches convolutionnelles avec de petits filtres à champ réceptif de taille (3×3). Le pas de convolution est fixé à 1 pixel afin que la résolution spatiale soit préservée après la convolution.

Les couches Maxpooling : La mise en commun spatiale est effectuée par cinq couches de max-pooling, qui suivent plusieurs couches de convolution. Le max-pooling est effectué sur une fenêtre de (2 × 2) pixels, avec une taille de stride fixée à 2.

Les couches entièrement connectées : La configuration des couches entièrement connectées est toujours la même : les deux premières ont 4096 canaux chacune et la troisième a 1000 canaux,

Les couches cachées : Toutes les couches cachées de VGG sont suivies par la fonction d'activation ReLU (une innovation importante d'AlexNet qui a réduit le temps de formation).

3 Apprentissage par transfert :

3.1 Définition :

L'apprentissage par transfert est une méthode d'apprentissage automatique où un modèle développé pour une tâche est réutilisé comme un point de départ d'un autre modèle pour une seconde tâche. Il s'agit d'une approche populaire dans le domaine de l'apprentissage profond, où des modèles pré-entraînés sont utilisés comme point de départ pour des tâches de vision par ordinateur et de traitement du langage naturel, étant donné les vastes ressources en temps et en calcul nécessaires pour développer des modèles de réseaux neuronaux sur ces problèmes et les énormes sauts de compétence qu'ils permettent d'obtenir sur des problèmes connexes. **[25]**

3.2 Apprentissage par transfert sur des images :

Il est courant d'effectuer un apprentissage par transfert avec des problèmes de modélisation prédictive qui utilisent des données d'image en entrée. Il peut s'agir d'une tâche de prédiction qui prend des images ou des données vidéo en entrée.

Pour ces types de problèmes, il est courant d'utiliser un modèle d'apprentissage profond pré-entraîné pour une tâche de classification d'images importante et difficile, comme la compétition de classification des images ImageNet. Ces modèles peuvent être téléchargés et incorporés directement dans de nouveaux modèles qui attendent des données d'image en entrée. Comme ***Vgg16***, ***modèles d'inception*** par google, ***RestNet*** par microsoft.

Cette approche est efficaces car les images ont été entraînées sur un grand corpus de photographies et exigent du modèle qu'il fasse des prédictions sur un nombre relativement important de classes, ce qui nécessite que le modèle apprenne efficacement à extraire les caractéristiques des images afin d'obtenir de bonnes performances dans ce problème.**[25]**

3.3 Avantage de l'apprentissage par transfert :

L'apprentissage par transfert est une optimisation, un raccourci pour gagner du temps ou obtenir de meilleures performances. En général, il n'est pas évident qu'il y ait un avantage à utiliser l'apprentissage par transfert dans le domaine tant que le modèle n'a pas été développé et évalué. Dans leur chapitre sur l'apprentissage par transfert, Lisa Torrey et Jude Shavlik décrivent trois avantages possibles à rechercher lors de l'utilisation de l'apprentissage par transfert : **[26]**

Un meilleur départ : La compétence initiale (avant le raffinement du modèle) sur le modèle source est plus élevée qu'elle ne le serait autrement.

Pente plus élevé : Le taux d'amélioration de la compétence pendant la formation du modèle source est plus élevé qu'il ne le serait autrement.

Asymptote plus élevée : La compétence convergée du modèle formé est meilleure qu'elle ne le serait autrement.

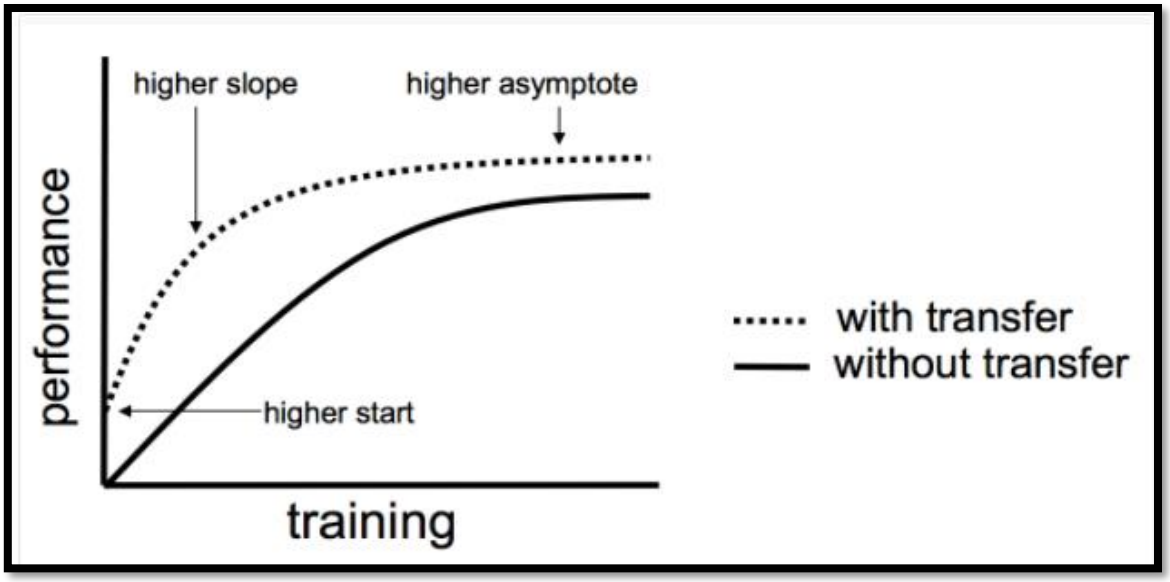


Figure II.19 : Les avantages de l'apprentissage par transfert **[26]**

4 Les étapes d'application d'un modèle d'apprentissage profond :

4.1 Le prétraitement des données:

Le prétraitement des données pour les réseaux de neurone est un processus par lequel nous rendons les données plus adaptées à l'entraînement des algorithmes de l'apprentissage profond qui sera implémentés avec l'utilisation de la bibliothèque scikit-learn, et pour cela il existe deux méthodes différentes pour assurer que les données sont prêtes à être utiliser et chaque technique comporte une formule mathématique :

4.1.1 La standardisation des données :

C'est l'une des techniques de « Feature Scaling », également appelée « **Z-Score normalisation** », elle est appliquée quand les *entrées de caractéristiques répondent* à des **distributions normales** « Distributions gaussiennes avec **une moyenne nulle et une variance unitaire** » avec des moyennes et des écart-types différents.

Par conséquent, cette transformation aura pour impact d'avoir toutes nos caractéristiques répondant à la même loi normale, elle peut être appliquée quand les caractéristiques ont des unités différentes par la formule mathématique suivante :

$$\mathbf{Z} = \frac{(X-m)}{e}$$

Tel que :

- *X* : c'est la valeur qu'on veut la standardiser.
- *m*: c'est la moyenne (***mean***) *des* observations pour cette feature.
- *e* : c'est l'écart-type (***Standard Deviation***) *des* observations pour cette feature.

4.1.2 La normalisation des données :

La normalisation est une étape de prétraitement cruciale, il est courant de normaliser les fonctionnalités avant de transmettre les données à un algorithme d'apprentissage automatique ou d'apprentissage profond. Cela aide à entraîner les algorithmes plus rapidement et à obtenir plus de performances. La normalisation est le processus par lequel vous représentez des données appartenant à une caractéristique particulière de telle sorte que sa moyenne soit nulle et que l'écart type soit égal à un.

La méthode « ***Min-Max Scaling*** » est la méthode la plus simple utilisé pour la normalisation des données, elle peut- être appliqué quand les données varient dans des échelles différentes. A l'issue de cette transformation, les caractéristiques seront comprises dans un intervalle fixe entre [0,1]. Le but d'avoir un tel intervalle restreint est de réduire l'espace de variation des valeurs d'une caractéristique « feature » et par conséquent réduire l'effet des valeurs aberrantes « *outliers* », cette transformation sera appliquée grâce à la formule suivante :

$$\mathbf{X} = \frac{\boldsymbol{X - Min(X)}}{\boldsymbol{Max(X) - Min(X)}}$$

Tel que :

- Min(X) : la plus petite valeur observée pour la caractéristique X.
- Max(X) : la plus grande valeur observée pour la caractéristique X.
- X : La valeur de la caractéristique qu'on veut la normaliser.

4.2 L'équilibre des données:

Un ensemble de données déséquilibrer c'est un ensemble dans on observe une large différence entre la classe minoritaire et la classe majoritaire, en outre le problème est que les modèles formés sur des ensembles de données non équilibrés ont souvent de mauvais résultats lorsqu'ils doivent généraliser ou prédire une classe ou classer des observations inconnues.

Afin d'avoir des données équilibrées, on peut utiliser des différentes stratégies pour la modification de l'ensemble de données avant l'entrainement de notre model prédictif, parmi ces stratégies on cite La technique de ré-échantillonnage.**[27]**

4.2.1 Le sous-échantillonnage « Undersampling » :

Le sous-échantillonnage vise à équilibrer la distribution des classes en éliminant de manière aléatoire les exemples de la classe majoritaire. Ceci est fait jusqu'à ce que les instances de classe majoritaire et minoritaire soient équilibrées.

Cette approche peut améliorer le temps d'exécution et les problèmes de stockage en réduisant le nombre d'échantillons de données. Mais d'un autre côté, elle peut écarter des informations potentiellement utiles qui pourraient être importantes pour la construction du modèle, et c'est pour cela cette méthode n'est pas trop fréquenter.

4.2.2 Le sur-échantillonnage « Oversampling » :

Le sur-échantillonnage augmente le nombre d'instances de la classe minoritaire en les répliquant de manière aléatoire afin de présenter une plus grande représentation de la classe minoritaire dans l'échantillon.

L'avantage du sur-échantillonnage par rapport au sous-échantillonnage est que cette méthode n'entraîne aucune perte d'information. Mais elle augmente la probabilité de su-rajustement puisqu'elle reproduit les événements de la classe minoritaire.

5 La transformation des données non images en format image :

5.1 La méthode Deepinsight :

5.1.1 Définition :

La méthode de ***DeepInsight*** consiste d'abord à transformer un échantillon non-image en une forme d'image bien organisée, puis à le fournir à l'architecture CNN à des fins de prédiction ou de classification.**[28]**

5.1.2 Principe de fonctionnement :

1. Un vecteur de caractéristique ***x*** comprenant des valeurs d'expression de gênes est transformé en une matrice de caractéristique ***M*** par une transformation ***T***.

2. L'emplacement des entités dans les coordonnées cartésiennes dépend de la similitude des entités.
3. Une fois que les emplacements de chaque caractéristique sont déterminés dans une matrice de caractéristiques, les valeurs d'expression ou les valeurs de caractéristique sont mappées, Donc une image unique pour chaque échantillon.
4. ***N*** échantillons de *d* caractéristiques fourniront *N* échantillons de ***m*** ×***n*** matrices de caractéristiques. Cette forme matricielle 2D aura toutes les caractéristiques ***d.***
5. Ensuite, cet ensemble de ***N*** matrices de caractéristiques est traité dans l'architecture CNN pour apprendre le modèle et fournir une prédiction ou une classification. **[28]**

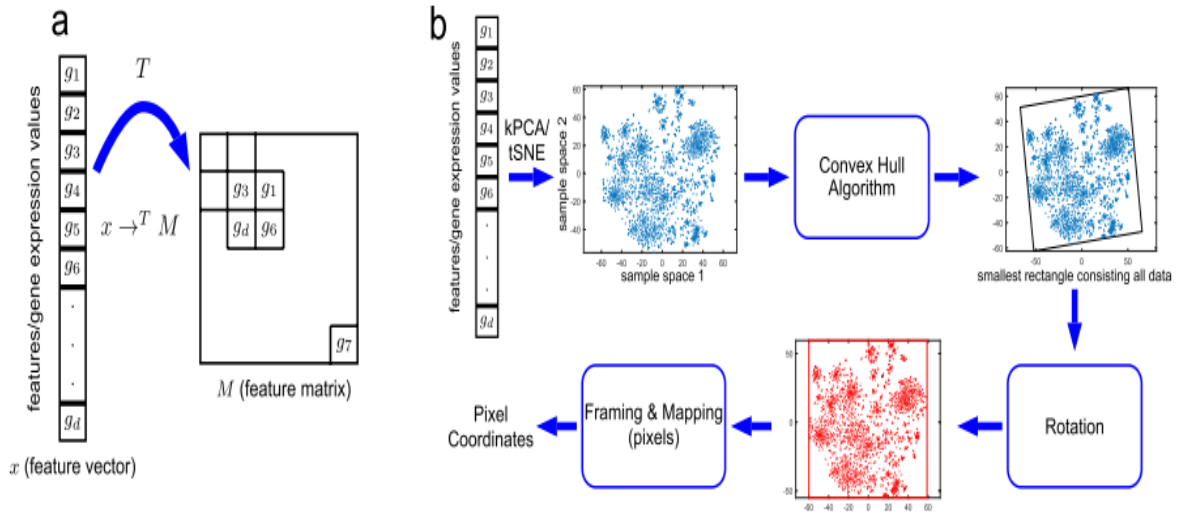


Figure II.20 : les étapes de la méthode DeepInsight

5.1.3 L'implémentation de DeepInsght :

Pour commencer la procédure de la transformation d'un échantillon non-image en sa forme image, On a besoin d'un ensemble d'apprentissage pour trouver l'emplacement des entités. Si l'ensemble d'apprentissage composé de ***n*** échantillons est défini comme $\chi = \{ x_1 , x_2 ,.. , x_n \}$ où un vecteur de caractéristiques a *d* caractéristiques , alors nous pouvons également définir un gène ou un ensemble de caractéristiques $G = \{ g_1 , g_2 , \ldots, g_d \}$, Nous avons utilisé cet ensemble de fonctionnalités *G* et appliqué une technique de mesure de similarité ou une technique de réduction de dimensionnalité comme le T-SNE ou l'analyse en composantes principales du noyau (kPCA) pour obtenir un plan 2D, Ce sont des techniques de réduction de dimensionnalité non linéaire, Les points dans ce plan cartésien sont les

caractéristiques ou les gènes. Ces points définissent uniquement l'emplacement des entités, pas l'entité elle-même ou les valeurs d'expression. Une fois l'emplacement des entités défini, l'algorithme de l'enveloppe convexe est utilisé pour trouver le plus petit rectangle contenant tous les points. Etant donné que l'image doit être encadrée sous une forme horizontale ou verticale pour l'architecture CNN, une rotation est effectuée. Ensuite, les coordonnées cartésiennes sont converties en pixels.

La conversion des coordonnées cartésiennes en images de pixels est effectuée en faisant la moyenne de certaines caractéristiques car la taille de l'image à une limitation en pixels. La trame de pixels sera donc constituée des positions des caractéristiques (ou gènes) pour un échantillon x_j (pour $j = 1, 2, \ldots, n$). Une fois l'emplacement est déterminé, l'étape suivante consiste à mapper les valeurs de caractéristiques (ou d'expression génique) à ces emplacements de pixels. Si plusieurs entités ont acquis le même emplacement dans le cadre de pixels, alors lors de la cartographie des entités, les entités respectives seront moyennées et placées au même emplacement. Par conséquent, si la résolution de l'image ou la taille de la grille est très petite (par rapport au nombre d'entités données), alors de nombreuses caractéristiques se chevauchent et la représentation de l'image peut ne pas être très précise. Une résolution appropriée doit être sélectionnée en fonction de la capacité matérielle et du nombre de fonctionnalités requises pour le traitement. Alternativement, une réduction de dimensionnalité peut être appliquée a priori. **[28]**

5.2 La technique de réduction de dimensionnalité T-SNE :

5.2.1 Définition :

T-SNE est *un algorithme* de réduction de dimensionnalité en anglais ***(t-distributed stochastic neighbor embeddin)***. Développé par *« **Laurens van der Maaten »** et « **Geoffrey Hinton »* en 2008, il consiste à analyser des *données* décrites dans des espaces à grande dimensionnalité pour les représenter dans des espaces à deux ou trois dimensions, rarement quatre dimensions.***[29]***

5.2.2 **Principe de fonctionnement :**

C'est est un algorithme non-linéaire d'extraction de caractéristiques qui sert à construire une nouvelle représentation des données de telle sorte que les données proches dans l'espace original aient une probabilité élevée d'avoir des représentations proches dans le nouvel

espace. A l'inverse, les données qui sont éloignées dans l'espace original, ont une probabilité faible d'avoir des représentations proches dans le nouvel espace. En pratique la similarité entre chaque paire de données, dans les deux espaces, est mesurée par le biais de calculs probabilistes basés sur des hypothèses de distribution. Et les nouvelles représentations se construisent de telle sorte à minimiser la différence entre les distributions de probabilités mesurées dans l'espace original et celles du nouvel espace. ***[29]***

5.2.3 L'intérêt de la réduction de dimensionnalité :

- La réduction du nombre des variables dans les données d'apprentissage.
- Une meilleure visualisation des données obtenues.
- La réduction des coûts de calcul et de stockage des informations.
- La réduction du poids des données, on économise l'espace mémoire.

5.3 La normalisation des images :

La couche unique de l'image a 256 nuances qui sont normalisées dans la plage de [0 ,1]. Par conséquent, les valeurs des caractéristiques doivent être normalisées avant d'appliquer la transformation d'image. Nous avons effectué deux types de normalisations :**[28]**

5.3.1 La première normalisation :

Chaque caractéristique est supposée indépendante et donc normalisée par son minimum et son maximum.

5.3.2 La deuxième normalisation :

La topologie des caractéristiques mutuelles est conservée dans une certaine mesure en la normalisant avec la maximale valeur de l'ensemble de la formation.

Ensuite, le DeepInsight évalue les performances de l'ensemble de validation sur les deux types de normalisation et accepte celle avec l'erreur de validation la plus faible.

6 Les mesures d'évaluations :

Un modèle performant c'est un modèle qui généralise, c'est-à-dire il a la capacité à faire des prédictions non seulement sur les données utilisées pour la construction, mais surtout sur de nouvelles données, Peu importe le projet il est nécessaire d'évaluer la performance de modèle pour mesurer les risques mais également pour faire la comparaison entre plusieurs algorithmes ou plusieurs versions de même algorithme.

Il existe plusieurs indicateurs permettant de mesurer la performance des modèles, et chacun a ses spécificités « pour les modèles de classification et pour les modèles de régression » :

6.1 La matrice de confusion:

La matrice de confusion aide à visualiser si le modèle est confus dans la discrimination entre les deux classes, il s'agit d'une matrice «2 x 2 », les étiquettes de deux lignes et deux colonnes sont positives et négatives pour refléter les deux étiquettes de la classe, pour refléter les deux étiquettes de classe, Les 4 éléments de la matrice représentent les 4 métriques qui comptent le nombre de prédictions correctes et incorrectes faites par le modèle :

- ***True Positive (TP) :*** la prédiction et la valeur réelle sont positives.
- ***True Negative (TN):*** la prédiction et la valeur réelle sont négatives.
- ***False Positive (FP) :*** la prédiction est positive alors que la valeur réelle est négative.
- ***False Negative (FN) :*** la prédiction est négative alors que la valeur réelle est négative.

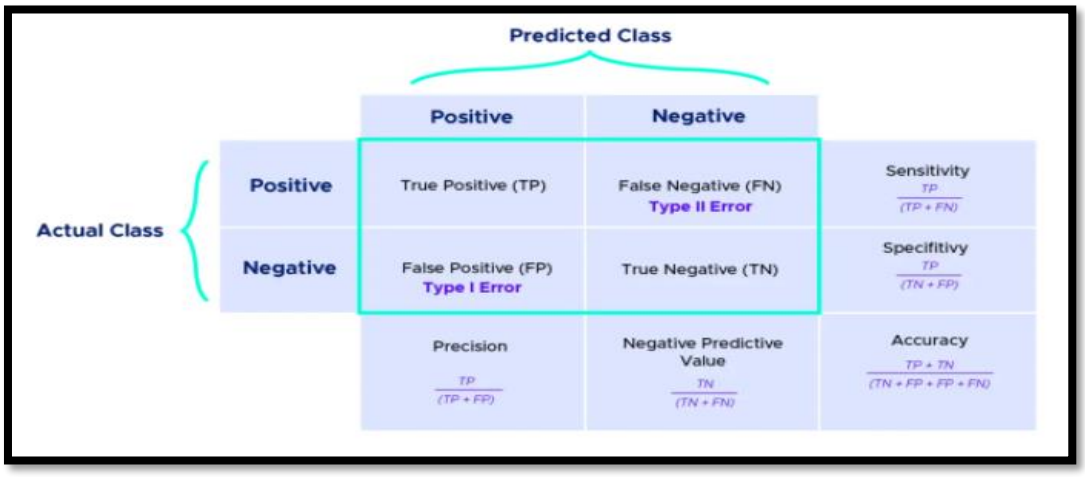


Figure II.21 : La matrice de confusion

6.2 La précision :

La précision de la classification est le nombre total de prédiction correctes devisé par le nombre total de prédictions effectué »es pour un ensemble de données, elle permet de répondre à la question suivante : Quelle proportion d'identifications positives était effectivement correcte ?

$$\text{Précision} = TP/ (TP + FP)$$

6.3 Le rappel :

Le rappel est une métrique qui quantifie le nombre de prédictions positives correctes faites à partir de toutes les prédictions positives qui auraient pu être faites Il permet de répondre à la question suivante : quelle proportion de résultats positifs réels a été identifiée correctement ?

$$\text{Rappel} = TP/ (TP + FN)$$

6.4 F mesure :

F- Mesure offre un moyen de combiner la précision et le rappel en une seule mesure qui capture les deux propriétés.

$$\text{Mesure F} = (2 * \text{Précision} * \text{Rappel}) / (\text{Précision} + \text{Rappel})$$

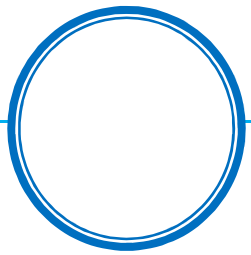
6.5 Accuracy :

Accuracy est une métrique qui décrit généralement les performances du modèle dans toutes les classes. Il est utile lorsque toutes les classes sont d'égale importance. Il est calculé comme le rapport entre le nombre de prédictions correctes et le nombre total de prédictions.

$$\text{Accuracy} = (TP + TN) / (TP + TN + FP + FN)$$

7 Conclusion:

Dans ce chapitre nous avons vu en général le domaine du Deep Learning et ces différentes architectures les plus fréquentes ainsi que ses utilités dans différents domaines, Egalement en a détaillé les mesures de l'évaluation des modèles.



CHAPITRE III

Conception de la méthode proposée

1 Introduction :

Les réseaux neuronaux convolutifs se sont avérés très adaptés à l'analyse de diverses données non structurées. Cette capacité a permis aux algorithmes d'apprentissage profond de jouer un rôle important dans la détection précoce du cancer par la classification des données génétique. Dans ce chapitre, nous allons présenter notre approche proposée pour la détection du cancer avec l'utilisation des données génétique obtenu à partir des bio-puces d'ADN, qui est basée sur un modèle convolutif d'apprentissage profond.

2 L'approche proposée :

Actuellement, la tâche de diagnostiquer les maladies repose sur les données d'expression génétique. L'une des techniques qui aide à cette tâche est l'utilisation des algorithmes de l'apprentissage profond. Ces algorithmes sont efficaces dans l'identification et la classification des gènes informatifs qui peuvent être ensuite utilisés pour la prédiction des classes des échantillons à tester.

Dans le cadre de l'identification du cancer, Nous avons examiné un algorithme d'apprentissage profond basé sur le réseau neuronal convolutionnel « CNN », pour la classification des données de puces ADN. La performance du model proposé a été évaluée sur quatrejeux de données ou chaque ensemble de données présente un type précis du cancer.

Dans la présente étude, nous avons commencé par donner un diagramme général qui présente les différentes méthodes utilisées sur les ensembles de données avant l'application du model d'apprentissage profond. Comme il est indiqué dans la Figure « 1 », la première étape est de faire un prétraitementsur les données afin de les rendre plus adapté à l'application du model convolutive « CNN », ensuiteon a utilisédes différentes méthodes après les avoir copiées en deux exemplaires (Dimension1 et Dimension 2) pour améliorer les performances de la classification génétique de notre modèle CNN de l'apprentissage profond.

3 Schéma de l'approche proposé :

La figure III.1 récapitule les différentes phases suivies dans la classification des données d'expressions génétiques issues des puces d'ADN concernant un type de cancer.

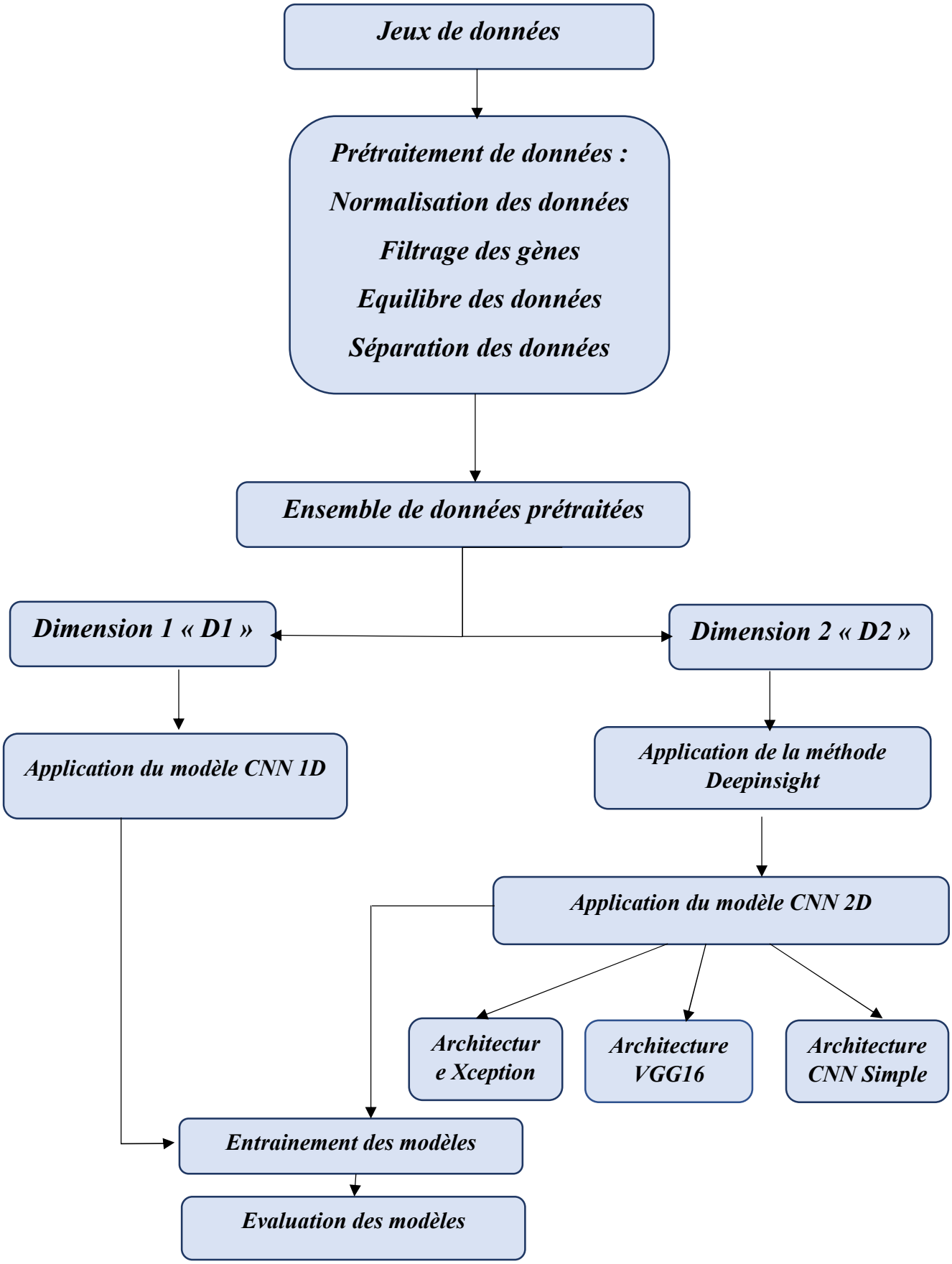


Figure III.1 : Schéma des étapes de l'approche proposé

4 Les Jeux de données :

4.1 Definition:

C'est une collection d'information ou un ensemble de données, généralement cet ensemble est collecté pour un but spécifique à partir des diverses sources, organisés et disposés dans un format prêt à être traité. Parfois elle peut être plus complexe lorsque les données sont dans un format non structuré tel que l'image ou l'audio, et elle doit être présentées de manières standard pour être adapté à l'analyse, et au traitement des données.

4.2 Description des jeux de données utilisées :

Dans cette étude, quatre jeux de données sur le cancer sont utilisés qui sont stockées sous format CSV, ces ensembles de données contiennent des profils d'expression génétique qui sont extraits à l'aide de la technologie des puces à ADN.

Tous les profils génétiques des patients atteints de la tumeur et des patients normaux ont été décrits comme des ensembles de données de classes différentes. Le fichier de données stocke les valeurs sous format numérique, organisées en lignes et en colonnes. Chaque ligne indique le numéro du patient et chaque colonne indique le numéro du gène dans l'ensemble de données génétique sur le cancer.

4.2.1 Colon cancer :

C'est un ensemble de données qui contient des informations génétiques relatives au cancer du côlon de 63 personnes, classées par : 34,92% échantillons malades et 63,49% échantillons non malades.

4.2.2 Leukemia cancer :

C'est un ensemble des informations génétiques de 72 personnes relatives au cancer de la leucémie, classées par deux classes : 63,51% sont malades et 33,78% classées comme non malades

4.2.3 Prostate cancer :

C'est un jeu de données qui contient des données génétiques relatives au cancer de la prostate de 136personnes, classées avec : 56,61% personnes malades et 43,38% classées comme non malades.

4.2.4 Tumeur du système nerveux central :

C'est un ensemble de données composé par des données génétiques de 60personnes relatives à la tumeur du système nerveux central, classées par deux classes : 35% sont malades et 65% classées comme non malades.

Le tableau suivant présente la description des ensembles de données utilisés dans cette étude, en termes de nombre de gènes et de nombre de patients(échantillons), ainsi que leur différente classe qui indique si le patient est malade ou non.

Dataset	Nombre de gènes	Nombre d'échantillons	Les classes
Colon cancer	2001	63	1 : malade -1 : non-malade
Leukemia cancer	7130	72	1 : malade -1 : non-malade
Prostate cancer	12601	136	Tumor : malade Normal : non-malade
CNS cancer	7130	60	1 : malade 0 : non-malade

Tableau III.1 : description des ensembles de données

5 Les étapes d'application de l'approche :

5.1 Le prétraitement des données:

Afin de rendre les données prêtes à être utilisés par le modèle d'apprentissage profond nous devons passée par les étapes suivantes :

5.2 La normalisation des données :

La normalisation est une exigence commune pour de nombreux estimateurs d'apprentissage profond implémentés par la bibliothèque scikit-learn, et pour cela nous avons utilisé la fonction ***Min-Max Scaling*** « définie dans le chapitre 2 » pour avoir des valeurs de

gènes varié dans un intervalle fixe entre [0,1]. Cette transformation sera appliquée par la formule ci-dessous :

$$G = \frac{g - Min(g)}{Max(g) - Min(g)}$$

Tel que :

- Min(g) : la plus petite valeur des gênes observée pour la caractéristique g.
- Max(g) : la plus grande valeur des gênes observée pour la caractéristique g.
- G : La valeur de gêne normalisé.

5.3 *Le filtrage des gènes:*

L'ensemble de caractéristiques de grande dimension avec relativement peu d'échantillonsconduit généralement à de mauvaises performances de classificateur pour l'apprentissage automatique,Pour résoudre ce problème de grande dimensionnalité de caractéristiques nous devons réduire ces dernières. Fondamentalement, il existe deux méthodes **l'extraction de caractéristiques** et **sélection de caractéristiques**.

Pour la sélection de caractéristiques il existe plusieurs techniques parmi eux on a ***Correlation-based Feature Selection (CFS)***qui est l'un des bien connustechniques pour classer la pertinence des caractéristiques en mesurant la corrélation entrecaractéristiques et classes et entre caractéristiques et autres caractéristiques.Étant donné le nombre de caractéristiques k et de classes C, le CFS a défini la pertinence des caractéristiques par la formule suivante :

$$Merit_s = \frac{kr_{kc}}{\sqrt{k + (k-1)r_{kk}}}$$

Où :

- **Merits :** est la pertinence du sous-ensemble de caractéristiques.
- **Rkc :** est le coefficient de corrélation linéaire moyen entre ces caractéristiques et classes.
- **rkk** : est le coefficient linéaire moyen coefficient de corrélation entre différentes caractéristiques.

5.4 La séparation des données :

Afin de mesurer la performance de notre modèle on a divisé notre jeu de données entre deux parties : une partie pour l'entrainement de données « Train Set » et l'autre partie pour le test « Test Set », cette division est faite aléatoirement par l'utilisation de la bibliothèque Sickit Learnqui est disponible sur python.**[30, 31]**

5.4.1 Le Train Set :

C'est les données d'entrainement qui sont utilisées pour construire et former un modèle d'apprentissage profond, ces données seront utilisées par ce modèle afin de faire une prédiction.

5.4.2 Le Test Set :

C'est les données inconnues qui sont utilisées pour tester le modèle d'apprentissage profond une fois l'algorithme est construit, on utilise ces données de test pour mesurer la performance du modèle et savoir si les prédictions sont juste ou non.

La figure suivante représente le pourcentage de notre séparation des jeux de données :

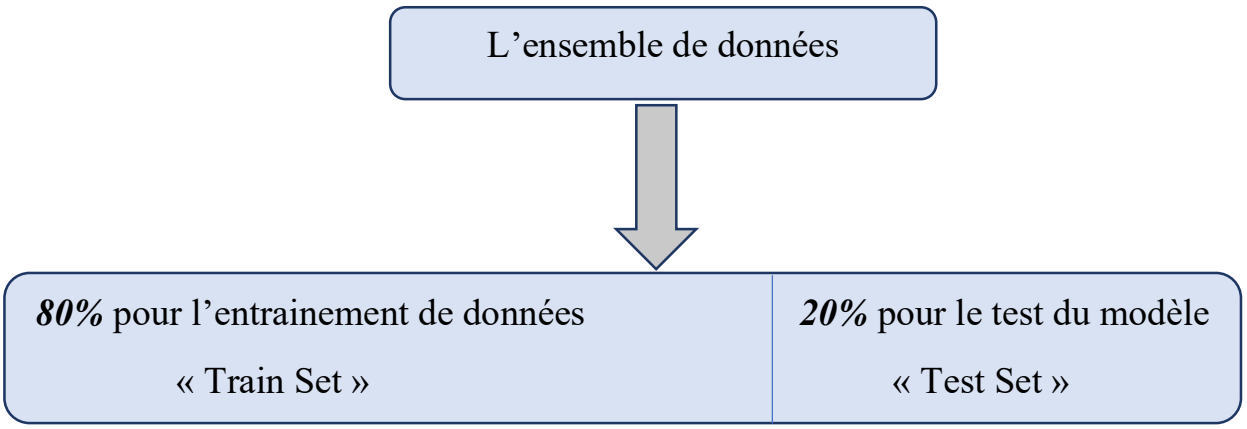


Figure III.2: la séparation des données

La taille des deux parties séparées peut se diffère d'un modèle a un autre, il existe certains modèles d'apprentissage profond qui nécessitent beaucoup de données pour l'entrainement, ainsi que d'autres modèles qui ont un besoin d'un ensemble de tests plus important pour optimiser les paramètres. En générale La taille du train Set peut être de 66 à 80% et la taille du test peut être de 20 à 33 %.

5.5 L'équilibre des données:

Afin d'avoir des ensembles de données équilibrées, on a utilisé la méthode de sur-échantillonnage « oversamling » pour avoir le même nombre de classes positive et négative dans l'ensemble d'entrainement.

Pour cela nous avons utilisé la fonction S-MOTE, qui est implémentée par la bibliothèque imbalanced learn. Cette méthode sélectionne d'abord une instance de la classe minoritaire d'une manière aléatoire, après elle trouve ses K voisins les plus proches. Une instance synthétique sera ensuite créée en choisissant au hasard l'un des K plus proche voisins. Ces instances seront ajoutées à l'ensemble de données original, le nouveau jeu de données est utilisé comme échantillon pour entraîner les modèles de classification.

Les résultats du prétraitement sur les quatre jeux de données sont présentés sur le tableau suivant :

	Normalisation des données	**Séparation des données « split »**	**L'équilibre des données par S-MOTE**
Colon cancer	Ensemble de données entre [1,0]	Train : 49 échantillons Test : 13 échantillons	Train : 64 [1 : 32 échantillons -1 : 32 échantillons]
Leukemia cancer	Ensemble de données entre [1,0]	Train : 57 échantillons Test : 15 échantillons	Train : 74 [1 : 37 échantillons -1 : 37 échantillons]
Prostate cancer	Ensemble de données entre [1,0]	Train : 95échantillons Test : 41 échantillons	Train : 108 [1 : 54 échantillons -1 : 54 échantillons]
CNS cancer	Ensemble de données entre [1,0]	Train : 49 échantillons Test : 13 échantillons	Train : 62 [1 : 31 échantillons -1 : 31 échantillons]

Tableau III.2 :prétraitement des données

6 L'implémentation de la méthode Deepinsight :

Cette approche sert à transformer les vecteurs des gènes en image pour chaque échantillon, et pour cela nous avons installé la bibliothèque « pyDeepInsight » qui fournit une version python de la procédure de transformation d'image de DeepInsight en utilisant la commande suivante :

```
[ ]  pip install git+git://github.com/alok-ai-lab/DeepInsight.git#egg=DeepInsight
```

Après l'installation de la bibliothèque pyDeepInsight on passe par les étapes suivantes :

6.1 La Normalisation:

La valeur minimale est ajustée pour chaque caractéristique, puis un maximum global est utilisé dans l'échelle logarithmique pour placer les valeurs des gènes 0 et 1.

6.2 L'installation de transformateur des images :

Initialiser le transformateur d'image, En utilisant « ImageTransformer » avec les paramètres suivante : feature_extractor = tsne, pixels = 100, random_state = 1701, n_job = -1.

6.3 L'entraînement du transformateur:

Entraîner le transformateur d'image sur les données d'entraînement, on obtient un tracé montrant les caractéristiques réduites (points bleus), convexe plein (rouge) et le rectangle de délimitation minimum (vert) avant la rotation.

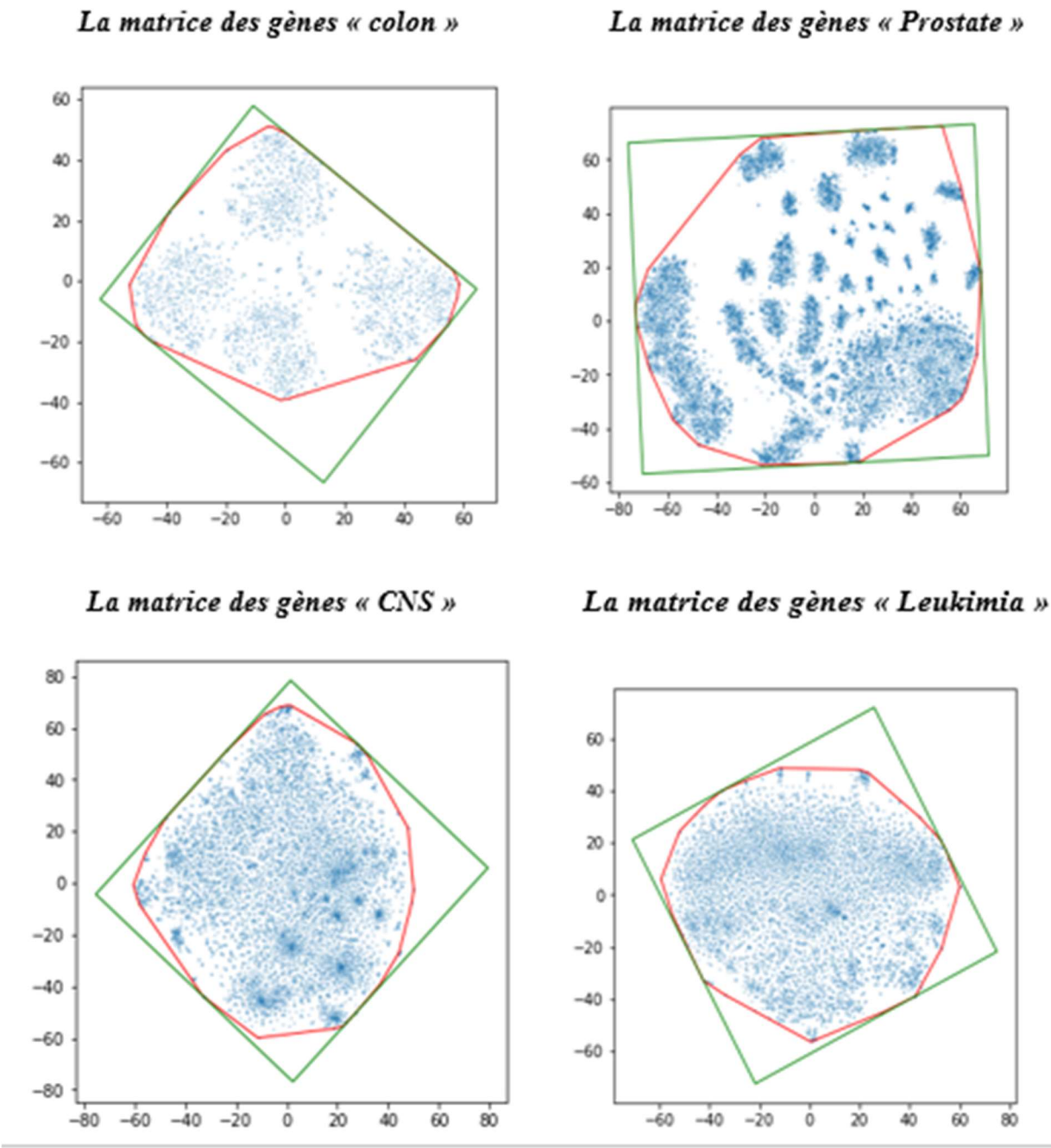


Figure III.3 : L'emplacement des gènes de l'ensemble d'entrainement

6.4 La matrice de densité:

Son objectif est de visualiser le chevauchement global des caractéristiques (des gènes).

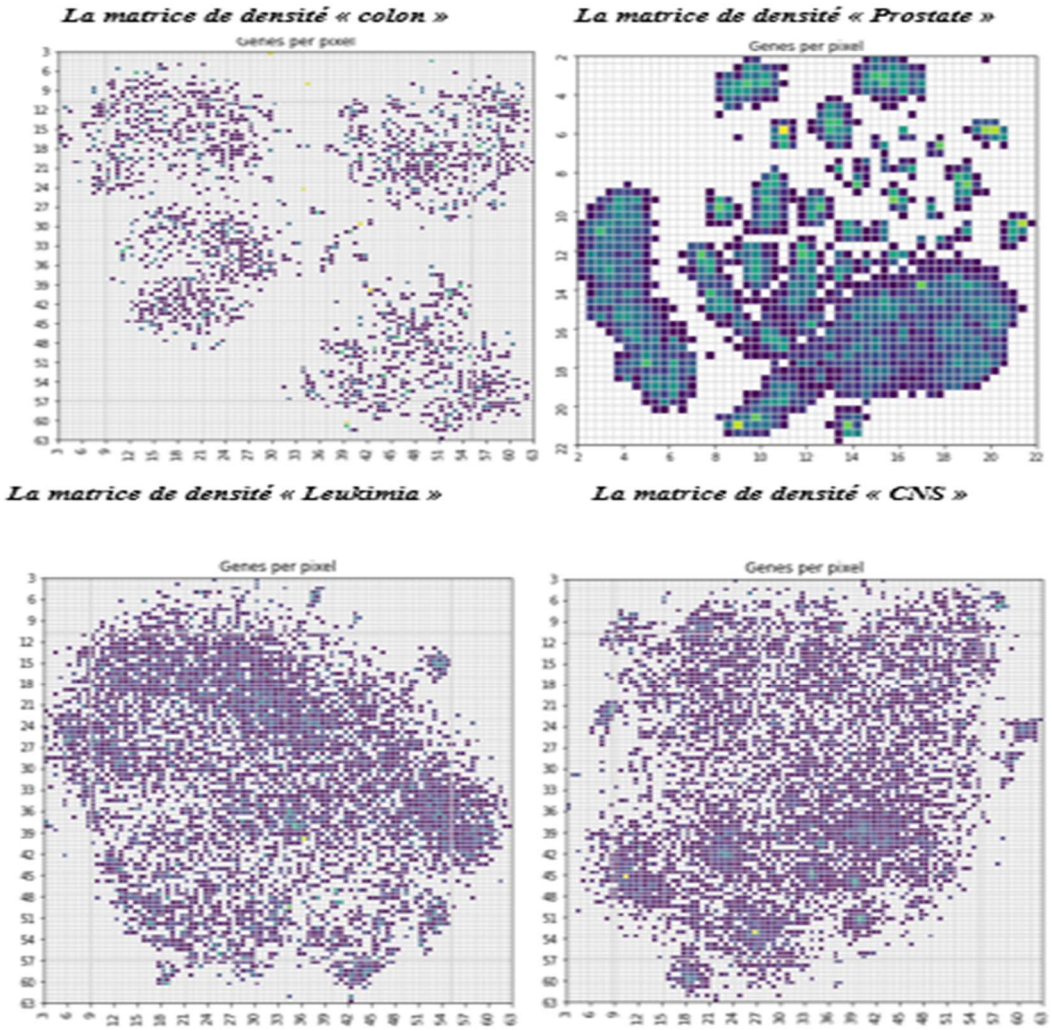
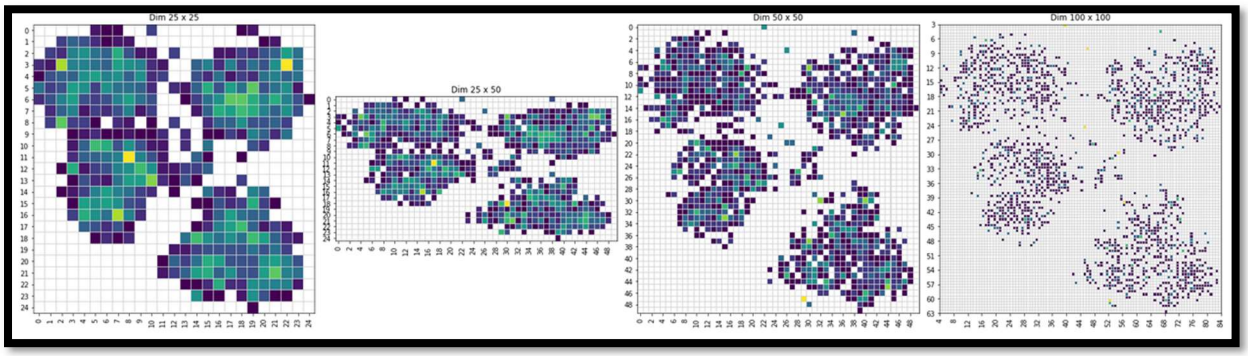


Figure III.4: la matrice de densité pour chaque ensemble de données

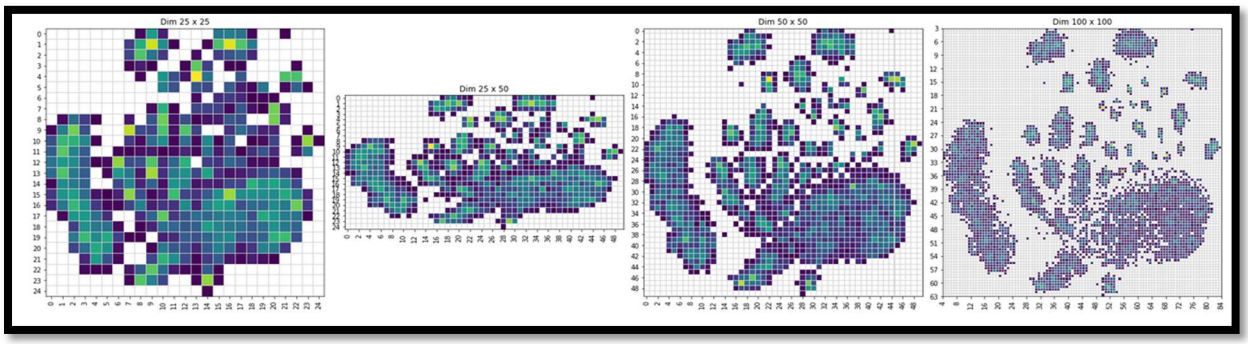
6.5 L'ajustement des pixels:

Mettre à jour la taille des pixels sans réapprentissage. La figure III.4 présente des exemples d'ajustements des pixels selon plusieurs dimensions (25x25, 25x50, 50x50 et 100x100).

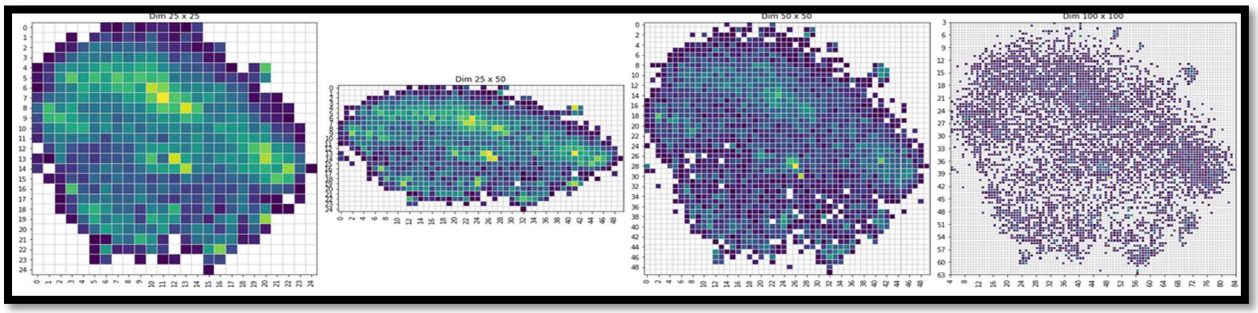
Ajustement des pixels « Colon »



Ajustement des pixel« Prostate »



Ajustement des pixels « Leukimia »



Ajustement des pixels « CNS »

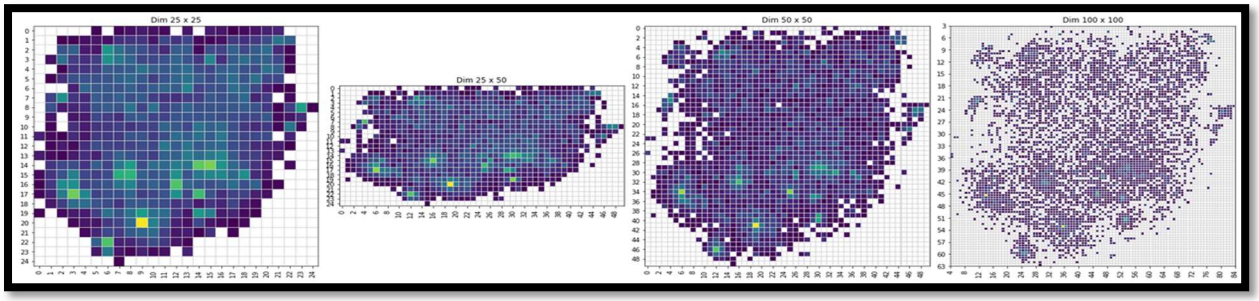


Figure III.5: ajustement des pixels de la matrice de densité

6.6 Transformation en image:

Le transformateur entraîné est utilisé pour transformer des échantillons de données d'entraînement après avoir les ajuster. Ces deux actions peuvent être effectué en même temps grâce à la fonction « fit_transform ( ) ».

6.7 Affichage des images d'entraînement :

Les images affichées sont de dimensions (100,100, 3).

Image d'entrainement « Colon »

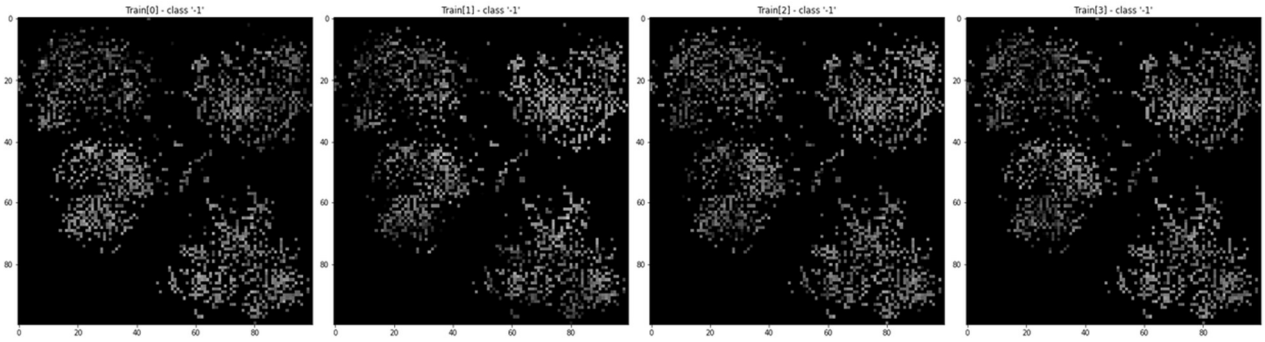


Image d'entrainement « Prostate »

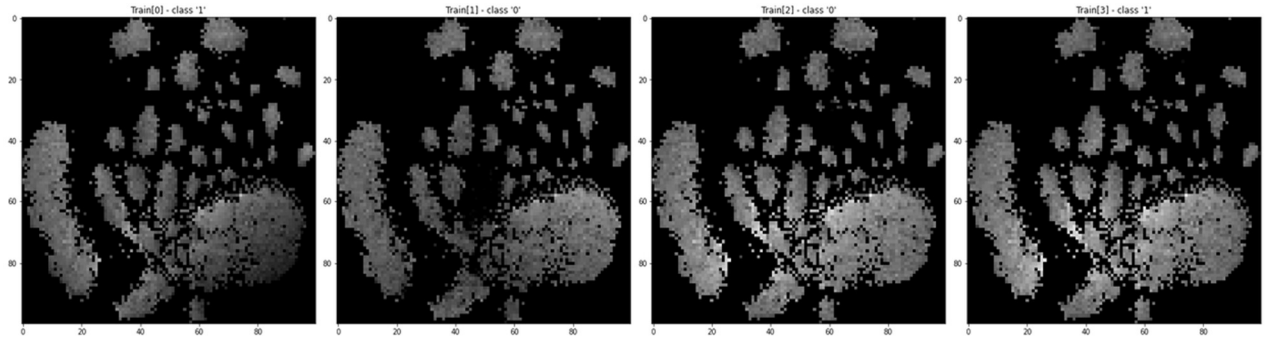


Image d'entrainement « Leukimia»

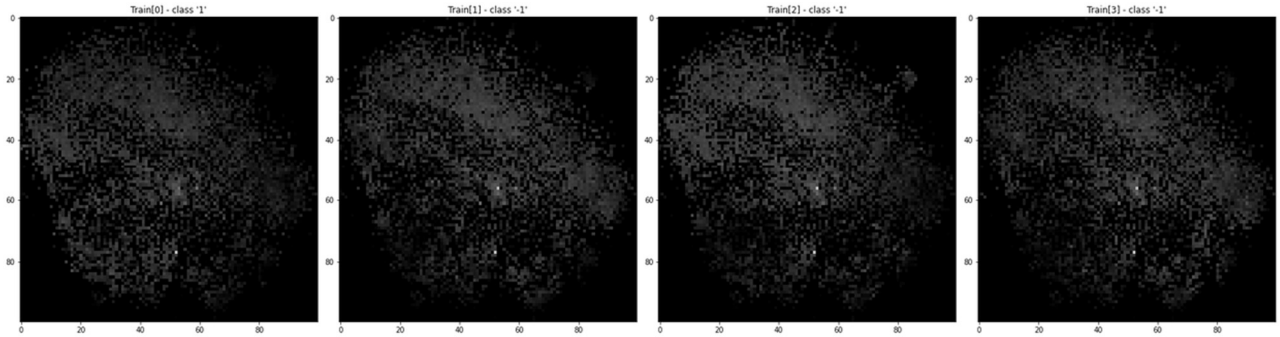


Image d'entrainement « CNS »

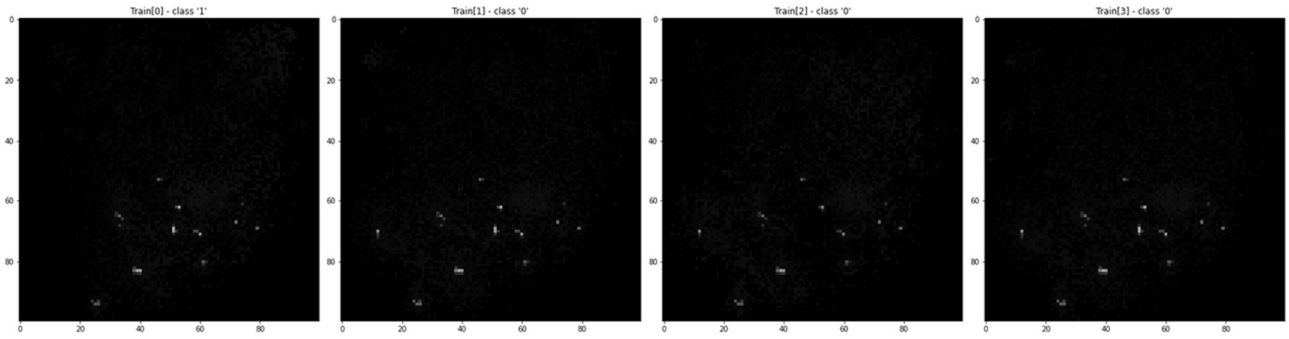


Figure III.6 : Les images d'entrainement pour chaque ensemble de données

6.8 La transformation des données de test :

Elle s'effectue de la même manière que la transformation des données d'apprentissage.

7 La Description des Modèles implémentées :

7.1 Description du modèle 1D:

Concernant cette architecture, nous avons implémenté un modèle CNN spécifique pour chaque jeu de données :

7.1.1 Colon :

Le modèle CNN configuré dans ce travail est composé de 14 couches en global, commençant par l'entrée vecteur de dimension [1,2000], suivie d'une couche covolutionnelle et une couche Maxpooling, ensuite quatre couches convolutionnelle et 4 couches Maxpooling (chaque couche convolutive est suivie d'une Maxpooling), Pour finir une couche entièrement connectée Fllaten est suivie d'une fonction d'activation **SoftMax**.

7.1.2 Prostate :

A propos ce modèle nous avons utilisé 16 couches en globale, commençant par une entrée de taille [1,12600], suivie de 12 couches entre la convolution et le maxpooling tel que chaque couche convolutionnelle et suivie d'une Maxpooling A la fin une couche entièrement connectée Fllaten est suivie d'une fonction d'activation **SoftMax**.

7.1.3 Leukimia :

Pour ce modèle nous avons utilisé 16 couches en globale, commençant par une entrée de taille [1,7129], suivie de 12 couches entre la convolution et le maxpooling tel que chaque couche convolutionnelle et suivie d'une Maxpooling A la fin une couche entièrement connectée Fllaten est suivie d'une fonction d'activation **SoftMax**.

7.1.4 CNS :

Pour ce modèle nous avons utilisé 13 couches en globale, commençant par une entrée de taille [1,7129], suivie de deux couches convolutionelle et une Maxpooling, ensuite quatre couches convolutionnelle entre chaqu'une une couche Maxpooling, A la fin une couche entièrement connectée Fllaten est suivie d'une fonction d'activation **SoftMax**.

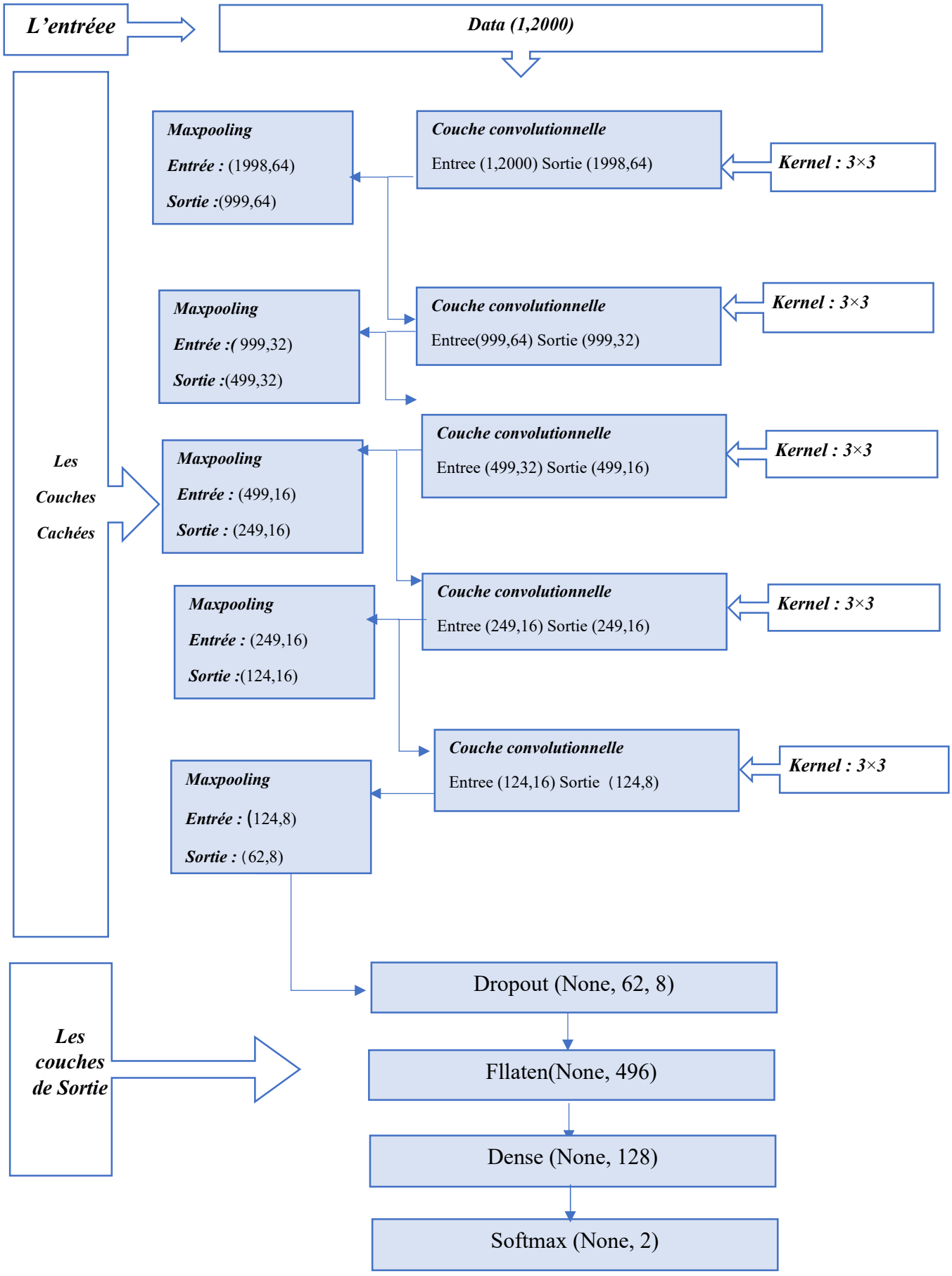


Figure III.7: Description du modèle 1D-colon

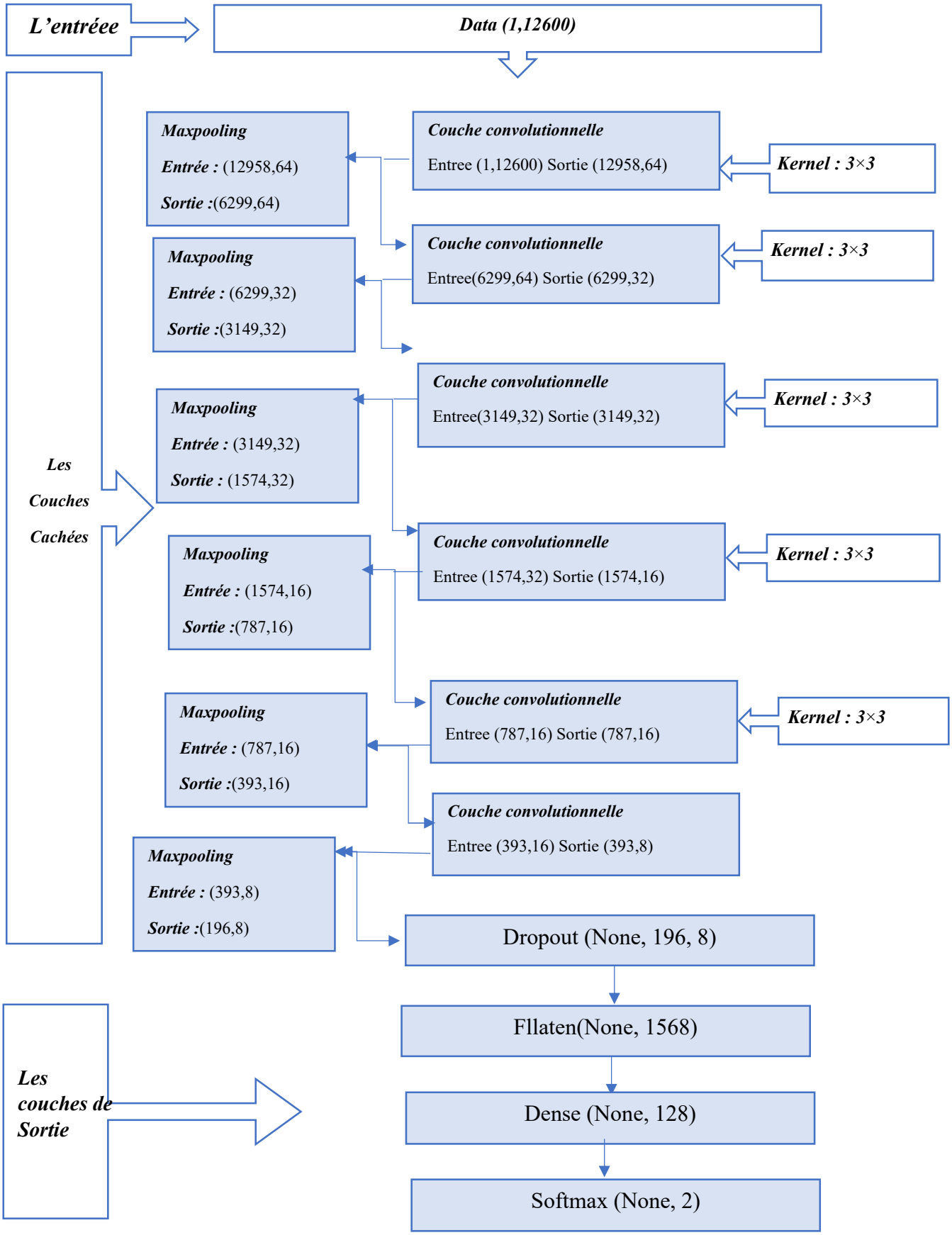


Figure III.8: Description du modèle 1D-prostate

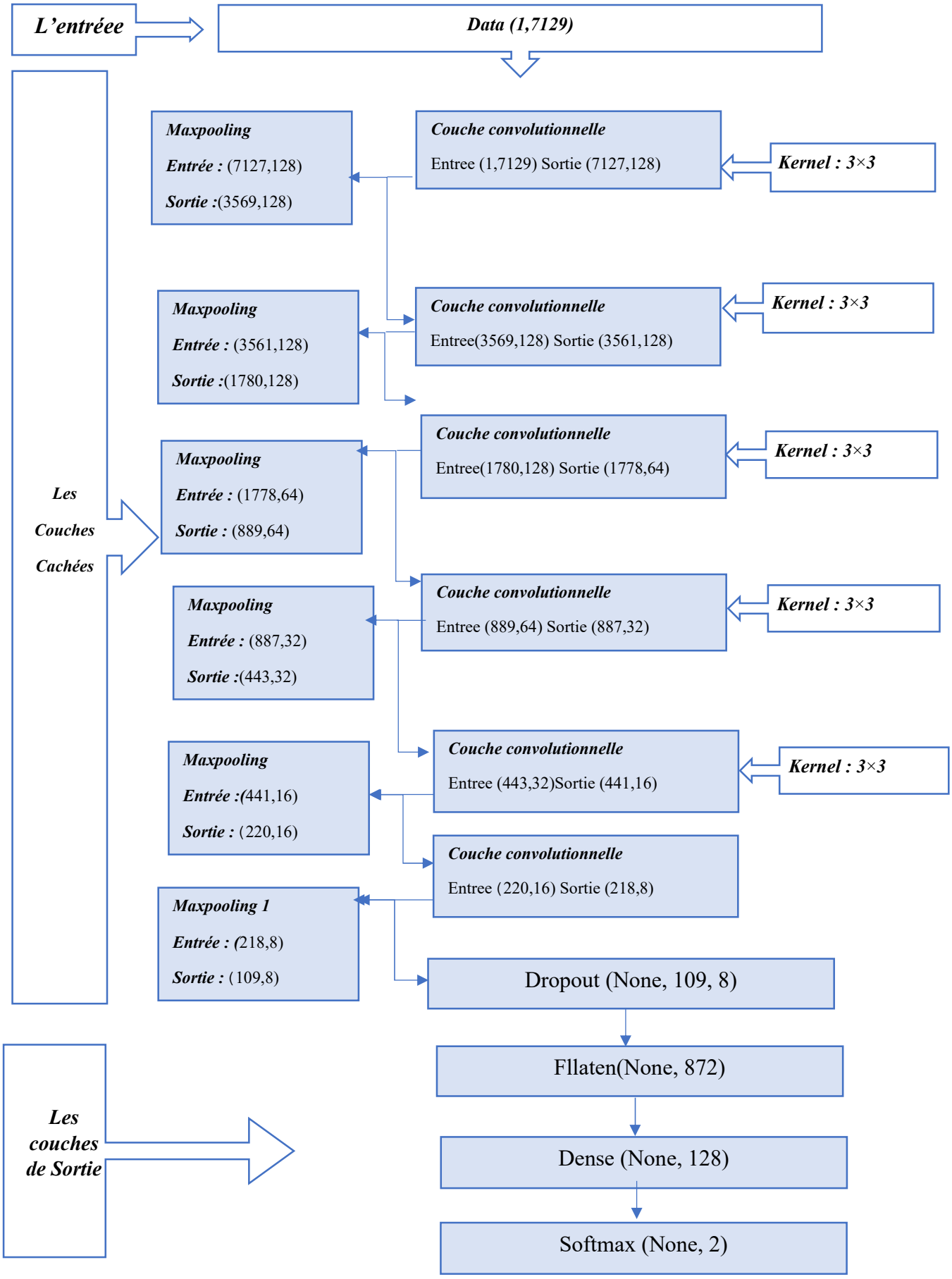


Figure III.9: Description du modèle 1D-Leukimia

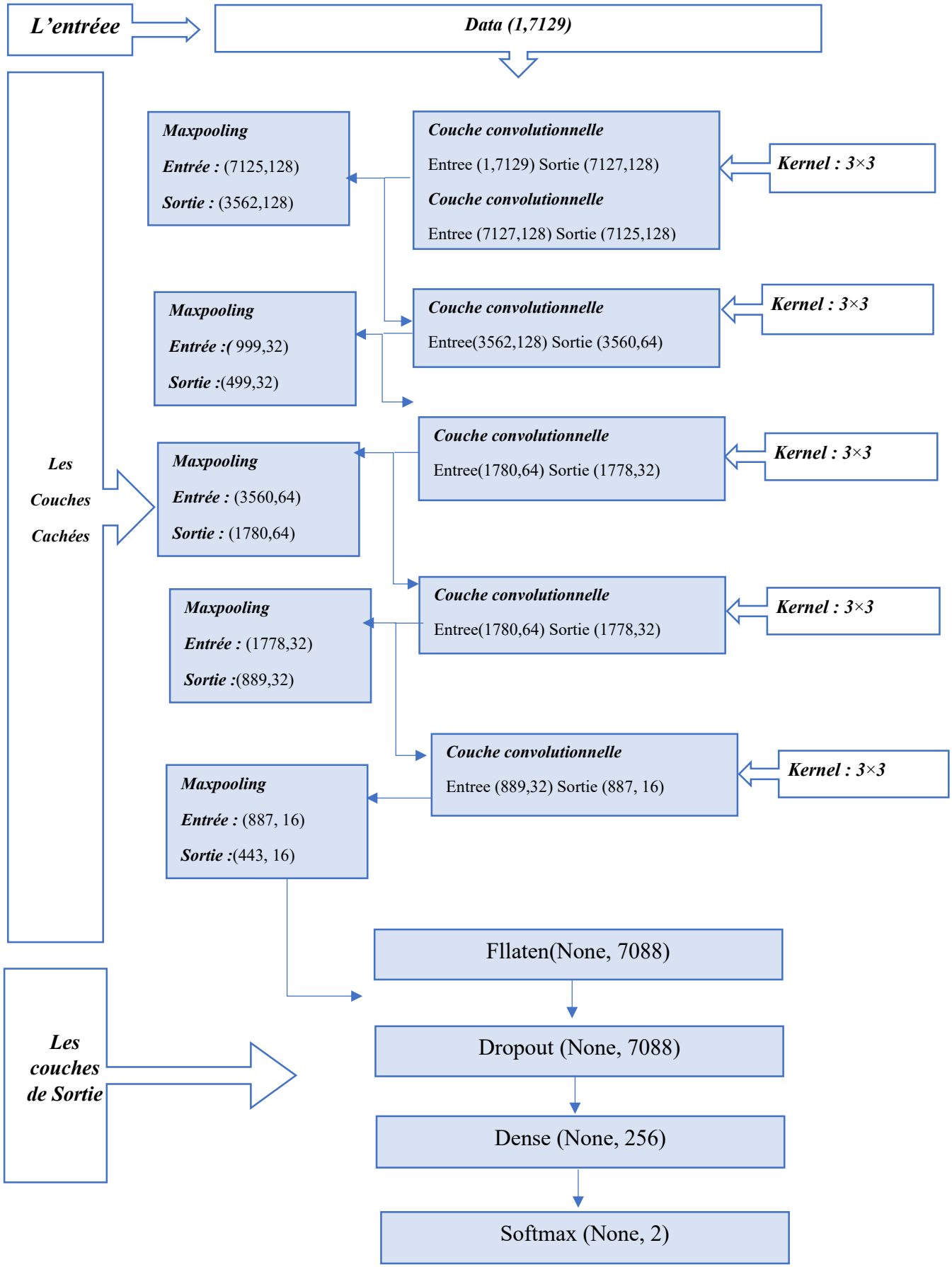


Figure III.10 : Description du modèle 1D-CNS

7.2 Description du modèle 2D:

Pour la réalisation de modèle CNN 2D, Nous avons passé par trois étapes principales, En commençant par la transformation de données en image grâce à la méthode de ***DeepInsight***, Puis l'implémentation des deux architectures ***VGG16*** et ***Xception***, également nous avons implémenter un modèle 2D propre, on finira par l'évaluation de chaque modèle.

7.2.1 Architecture VGG16 :

Pour l'implémentation de cette architecture nous avons utilisé l'apprentissage par transfert afin de conserver la partie d'extraction des caractéristiques qui est composé de quatre couches convolutive et deux MaxPooling (chaque 'une est placée après chaque deux couches de convolutions), ensuite neuf couches de convolution et trois MaxPooling (une couche après chaque 3 couches de convolutions) comme il est montré dans la figure suivante :



Figure III.11 : Les couches du modèle VGG16

Et pour la partie de classification, nous avons ajouté une couche entièrement connecté (Flatten), suivie de deux couches Dense, une Dropout (20%) et une fonction d'activation Sigmoid, comme il est décrit sur le schéma (figure III.14)

7.2.2 Architecture Xception :

Nous avons utilisé l'apprentissage par transfert pour conserver la partie d'extraction des caractéristiques de l'architecture Xception qui se compose de 36 couches de convolution devisé sur 14 blocks. Les données passent d'abord par le flux d'entrée, puis par le flux intermédiaire qui est répété huit fois, et enfin par le flux de sortie. Notez que toutes les

couches de convolution et de convolution séparable sont suivies d'une normalisation par lot toutes les couches de Convolution Séparable utilisent un multiplicateur de profondeur de 1.

Comme il est montré dans la figure suivante :

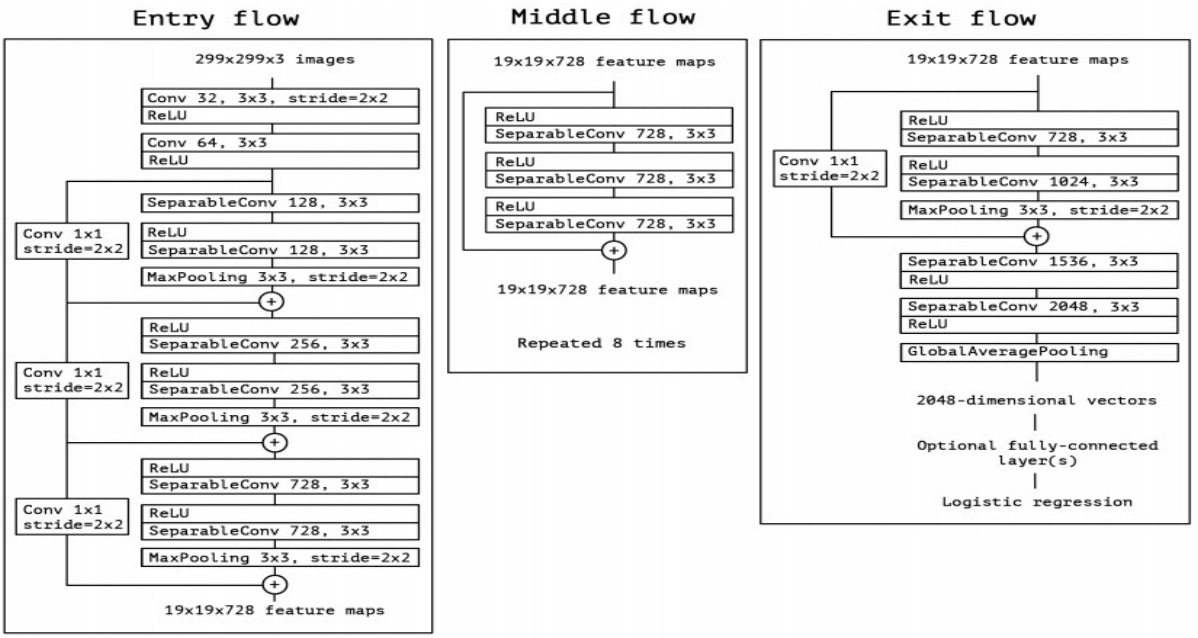


Figure III.12 : Les couches du modèle Xception

Et pour la partie de classification, nous avons ajouté une couche entièrement connectée (Flatten), suivie de deux couches Dense, une Dropout (20%) et une fonction d'activation Sigmoid, comme il est décrit sur le schéma (figure III.13)

***Figure* III.13** : Schéma du modèle 2D Xception

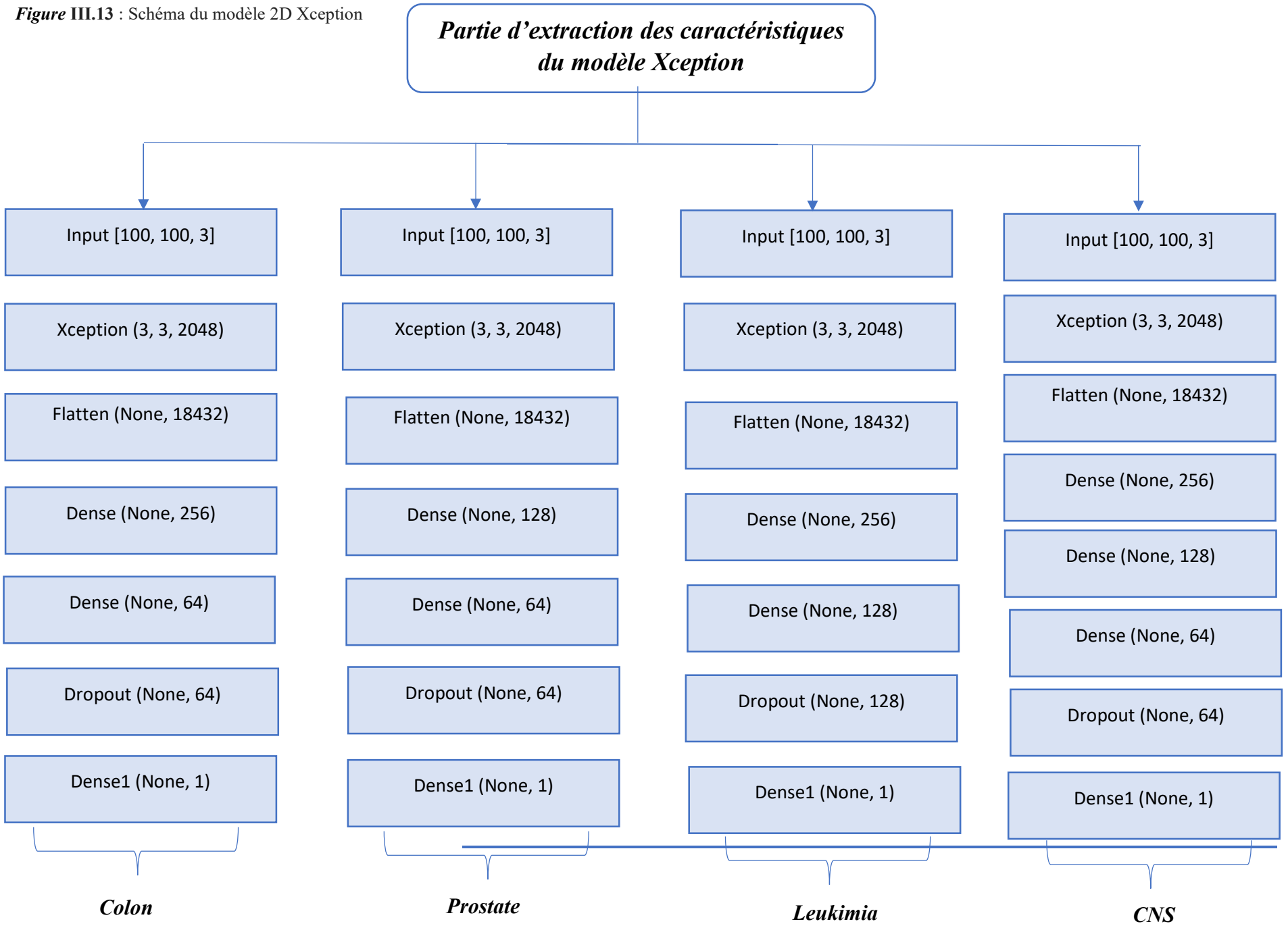
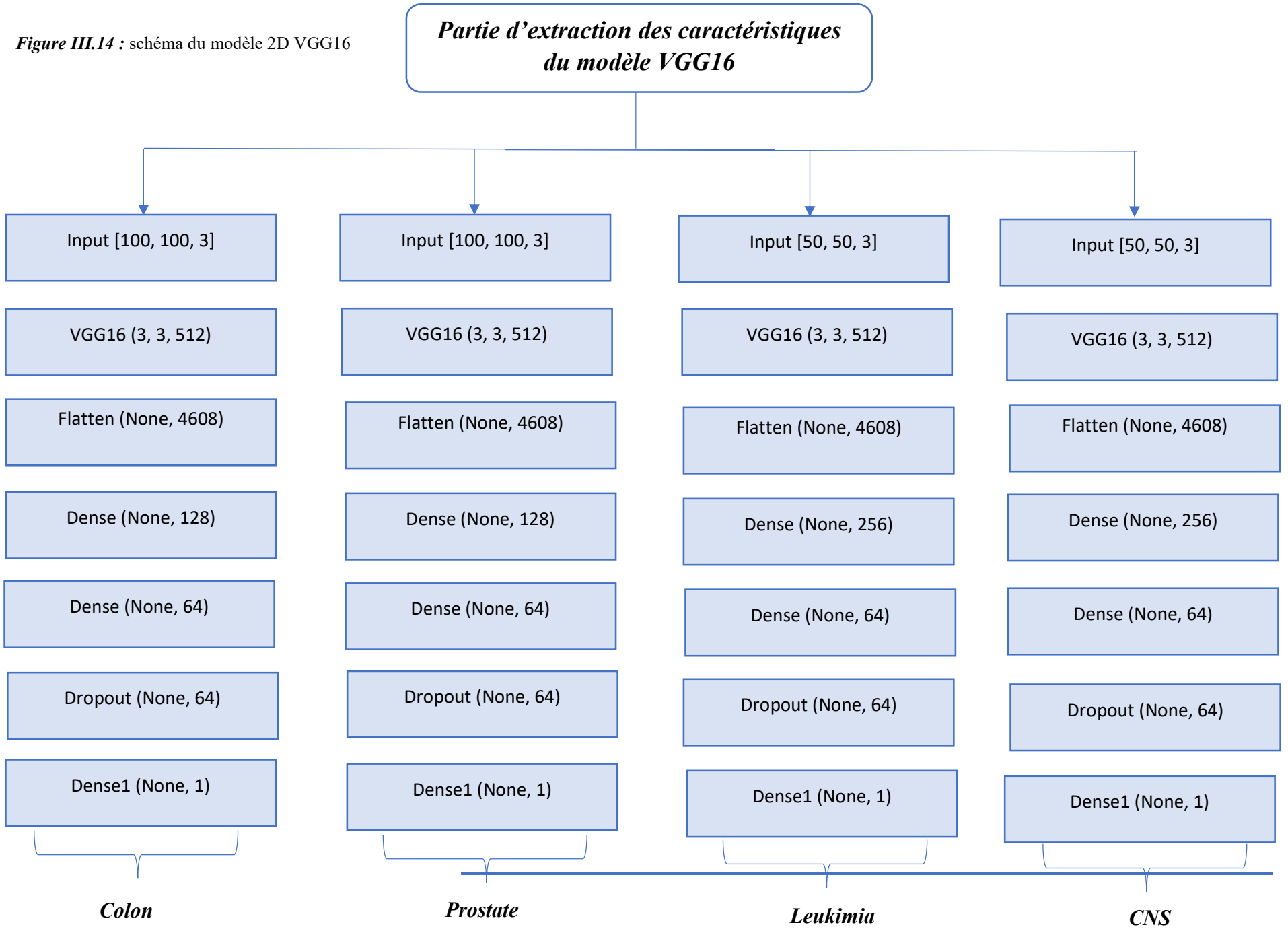


Figure III.14 : schéma du modèle 2D VGG16



7.2.3 Modèle CNN simple :

Concernant ce modèle, il s'agit de quatre couches de convolution entre chaqu'une une couche MaxPooling pour l'extraction des caractéristiques, Puis pour la partie de classification, une couche entièrement connecté (Flatten), suivie de deux couches Dense, une Dropout (20%) et une fonction d'activation Sigmoid, comme il est décrit sur le schéma (figure III.15)

Sachant que le même modèle a été implémenté pour les quatre ensembles de données

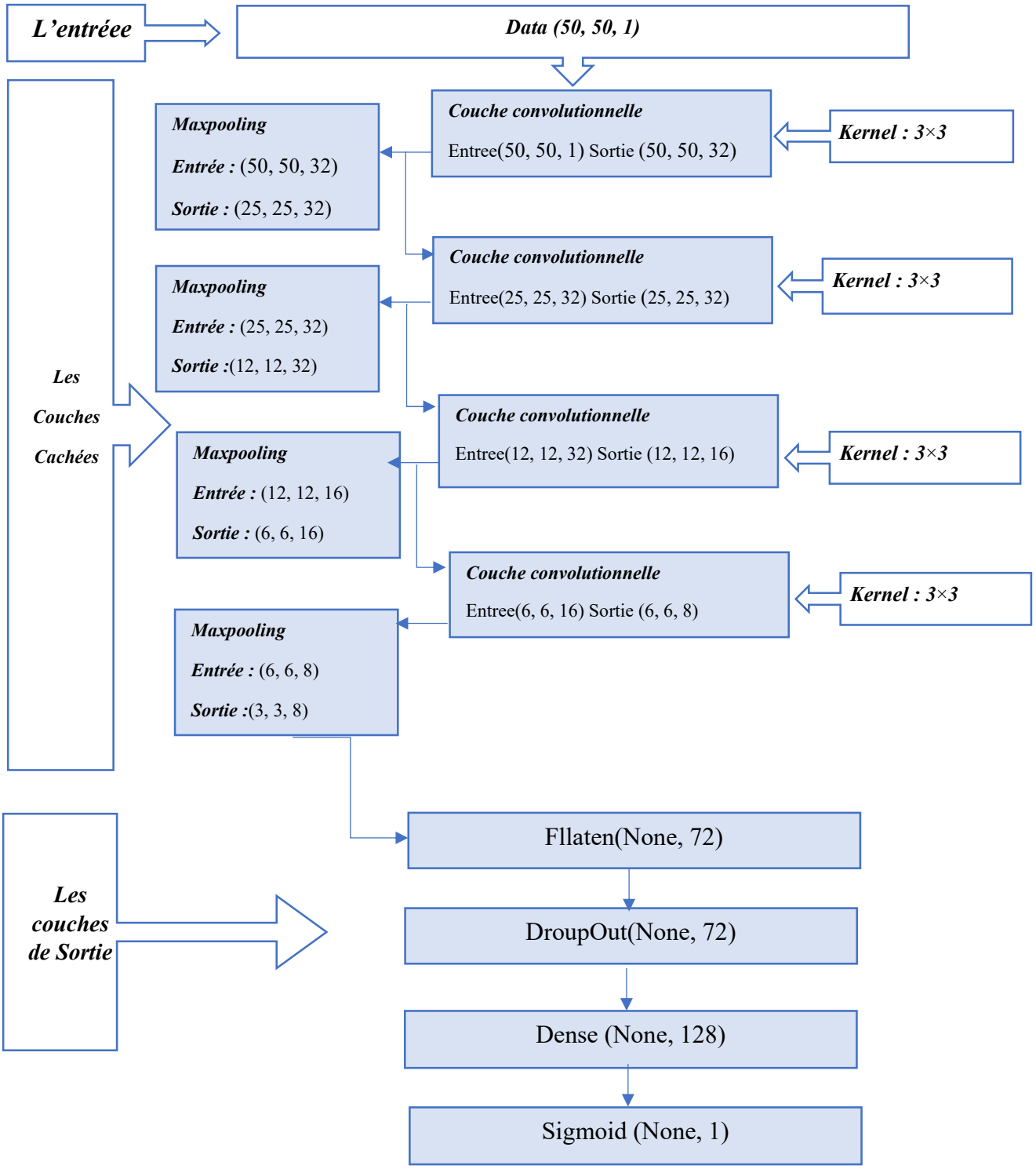
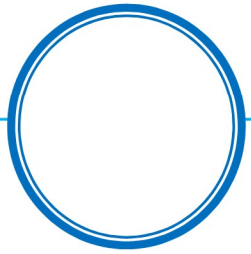


Figure III.15 : Description du modèle 2D simple

7.3 Conclusion:

Dans ce chapitre, nous avons détaillé notre approche proposé qui vise à créer un modelé capable d'améliorer la détection des gènes cancéreuse dans le cas du cancer de colon, prostate, sang, et de système nerveux central, en utilisant des différentes méthodes d'apprentissage profond et en sélectionnant les caractéristiques les plus adaptées.



CHAPITRE IV

Implémentation et résultats

1 Introduction :

Dans ce chapitre, nous allons présenter l'implémentation de notre approche proposée dans la partie précédente, d'abord on commence par la présentation des différents outils matériels et logiciels utilisés pour la réalisation de notre projet, ensuite on décrit les différentes étapes suivies pour l'implémentation de l'approche proposée, et on termine par présenter et discuter les résultats obtenus.

2 Outils et environnement :

2.1 Matériel utilisé :

Notre travail a été effectué avec l'utilisation d'un ordinateur portable qui a les caractéristiques suivantes : ***DELL Latitude E7250*** avec une capacité de mémoire de 8 Go, et un processeur Intel(R) Core(TM) i7-5600U, 5ème génération, vitesse 2,60 GHz, avec Windows 10 (type de système 64 bits).

2.2 Language de programmation:

2.2.1 Python :

Python est un langage de programmation open source, interprété, orienté objet, et puissant de haut niveau, qui a été développé par Guido van Rossum au début des années 1990, à l'Université d'Amsterdam au pays bas et nommé par rapport à la série télévisé « Monthy Python's Flying Circus », sa première apparition au monde était en février1991 avec une version numérotée Python 0.9.0.

Ce langage est connu par sa syntaxe simple et ses riches bibliothèques qui le rendent facile à apprendre et à utiliser pour différente objectifs et dans différents domaines comme (application console, application bureau, application web, application mobile, apprentissage automatique, apprentissage profond...).

2.2.2 Caractéristiques :

- ***Open-source :***

Python est un logiciel open source disponible publiquement, tout le monde peut utiliser son code source qui ne coûte rien.

- ***Langage de haut niveau :***

C'est un langage qui demande pratiquement une très peu connaissance au fonctionnement de la machine, car il est très proche au langage humain.

- ***Multiplateforme :***

Les langages de haut niveau sont multiplateforme, ce qui signifie qu'ils sont capables de fonctionner sur toutes les principales plates-formes matérielles et logicielles. Python peut être utilisé sur Linux, Windows, MacOs, Android, IOS.

- ***Orienté objet :***

Python est un langage de programmation orienté objet complet, avec des fonctionnalités telles que les classes, l'héritage, les objets et la surcharge « overload ».

- ***Interprété :***

Un script python n'a pas besoin d'être compilé pour être exécuté, il prend le code source en entrée, puis exécute chaque instruction immédiatement. Il n'est pas nécessaire de compiler ou de lier.

- ***Interactif :***

Python possède une console interactive où vous obtenez une invite Python (ligne de commande) et interagissez directement avec l'interpréteur pour écrire et tester vos programmes. Ceci est utile pour la programmation mathématique.

- ***Facile à apprendre :***

Langage (de script/extension) populaire, syntaxe claire et facile, pas de déclarations de types, gestion automatique de la mémoire, types de données et opérations de haut niveau, conçu pour être lu (syntaxe plus proche de l'anglais) et écrit rapidement (Code plus court comparé à C, C++ et Java).

2.3 Environnement d'implémentation :

2.3.1 Google colaboratory :

Le travail a été réalisé avec l'utilisation du Google Colaboratory, ou "Colab" en abrégé, ce sont des ordinateurs portables Jupyter hébergés par Google. Ils permettent d'écrire et

d'exécuter un code Python via un navigateur. Il offre un accès gratuit aux GPU et TPU, ne nécessite aucune configuration et permet de partager le code de manière transparente.

C'est un outil qui offre pendant 12 heures par jours les outils suivants :

GPU : 1xTesla K80, ayant 2496 cœurs CUDA, calcul 3,7, 12 Go (11,439 Go utilisables).

CPU : 1x cœur unique hyper fileté, c'est-à-dire (1 cœur, 2 fils) Processeurs Xeon @ 2,3 GHz (NonTurbo Boost), 45 Mo de cache.

RAM : 12 Gb disponibles.

Disque : 320 Go disponibles.

2.4 Les modules utilisés :

Le langage Python est connu par la simplicité de sa syntaxe et les bibliothèques riches qui le rendent facile à apprendre et à utiliser à des domaines différents, Pour réaliser notre expérience, nous avons utilisé les modules suivants :

2.4.1 Tensorflow :

C'est une interface accessible et hautement productive pour résoudre les problèmes d'apprentissage automatique, en mettant l'accent sur l'apprentissage profond moderne. Elle fournit des abstractions et des blocs de construction essentiels pour développer et expédier des solutions d'apprentissage automatique avec une vitesse d'itération élevée. Elle nous a permet d'implémenter et d'exécuter les deux architectures utilisées (***Xception*** *et* ***VGG16***) d'une manière facile et simple grâce à son écosystème complet et flexible d'outils.

2.4.2 Keras :

Keras est une API d'apprentissage profond écrite en Python, fonctionnant au-dessus de la plateforme d'apprentissage automatique « TensorFlow ». Il fournit des blocs de construction très puissants et abstraits pour construire des réseaux d'apprentissage profond.

2.4.3 Scikit-learn :

Scikit-learn est un module Python intégrant un large ensemble d'algorithmes d'apprentissage automatique pour des problèmes supervisés et non supervisés de moyenne étendu. Ce module vise à mettre l'apprentissage automatique à la portée des non-spécialistes en utilisant un langage général de haut niveau. Grace à la facilité d'utilisation, les performances, la documentation et la cohérence de l'API

2.4.4 Numpy :

C’est un paquetage fondamental pour le calcul scientifique en Python. Il s'agit d'une bibliothèque qui fournit un objet tableau multidimensionnel, ainsi qu'un ensemble de fonctions permettant d'effectuer des opérations rapides sur les tableaux, notamment des opérations mathématiques, logiques, de manipulation de formes, de tri, de sélection, et bien plus encore.

2.4.5 Pandas :

C’est une bibliothèque open source, pour le langage programmation Python permettant la manipulation et l'analyse des données. Elle propose en particulier des structures de données et des opérations de manipulation de tableaux numériques et de séries temporelles. Qui nous a permet d’analyser, de lire et de manipuler les jeux d’entraînement.

2.4.6 IO :

Ce module fournit les principales fonctionnalités de Python pour gérer divers types d’entrée et de sortie.

2.4.7 imbalanced-learn :

Imblearn est un paquetage python offrant un certain nombre de techniques de ré-échantillonnage couramment utilisées dans les ensembles de données présentant un fort déséquilibre entre les classes.

2.4.8 Matplotlib :

Matplotlib est un paquetage graphique 2D pour Python, destinée à tracer et visualiser des données sous formes graphiques.

3 Résultats et discussion :

3.1 Structure des modèles utilisés :

Avant de compiler nos modèles utilisés, nous présentons la structure du réseau configuré (résumé) pour chaque modèle CNN 1D.

Couches (Types)	*Sorties*	*Paramètres*
Conv1d_1 (Conv1D)	(None, 1998, 16)	64
Conv1d_2 (Conv1D)	(None, 1996, 32)	1568
Max_pooling1d_1	(None, 998, 32)	0
Conv1d_3 (Conv1D)	(None, 996, 64)	6208
Max_pooling1d_2	(None, 498, 64)	0
Conv1d_4 (Conv1D)	(None, 498, 32)	6176
Max_pooling1d_3	(None, 248, 32)	0
Conv1d_5 (Conv1D)	(None, 246, 128)	12416
Max_pooling1d_4	(None, 123, 128)	0
Conv1d_6 (Conv1D)	(None, 121, 128)	49280
Max_pooling1D_5	(None, 60, 128)	0
Dropout (Dropout)	(None, 60, 128)	0
Flatten3 (Flatten)	(None, 7680)	0
Dense (Dense)	(None, 1000)	7681000
Dense (Dense)	(None, 2)	2002

Tableau IV.1 : colon dataset

Total params: 7,758,714
Trainable params: 7,758,714
Non-trainable params: 0

Couches (Types)	***Sorties***	***Paramètres***
Conv1d_1 (Conv1D)	(None, 12598, 16)	64
Conv1d_2 (Conv1D)	(None, 12596, 32)	1568
Max_pooling1d_1	(None, 6298, 32)	0
Conv1d_3 (Conv1D)	(None, 6296, 64)	6208
Max_pooling1d_2	(None, 3148, 64)	0
Conv1d_4 (Conv1D)	(None, 3146, 64)	12352
Max_pooling1d_3	(None, 1573, 64)	0
Conv1d_5 (Conv1D)	(None, 1571, 128)	24704
Max_pooling1d_4	(None, 785, 128)	0
Dropout (Dropout)	(None, 785, 128)	0
Flatten3 (Flatten)	(None, 100480)	0
Dense (Dense)	(None, 1000)	100481000
Dense (Dense)	(None, 2)	2002

Tableau IV.2 : Prostate dataset

Total params: 100,527,898
Trainable params: 100,527,898
Non-trainable params: 0

Couches (Types)	***Sorties***	***Paramètres***
Conv1d_1 (Conv1D)	(None, 7127, 16)	64
Conv1d_2 (Conv1D)	(None, 7125, 16)	784
Max_pooling1d_1	(None, 3562, 16)	0
Conv1d_3 (Conv1D)	(None, 3560, 32)	1568
Max_pooling1d_2	(None, 1780, 32)	0
Conv1d_4 (Conv1D)	(None, 1778, 64)	6208
Max_pooling1d_3	(None, 889, 64)	0
Conv1d_5 (Conv1D)	(None, 887, 128)	24704
Max_pooling1d_4	(None, 443, 128)	0
Dropout (Dropout)	(None, 443, 128)	0
Flatten3 (Flatten)	(None, 56704)	0
Dense (Dense)	(None, 1000)	56705000
Dense (Dense)	(None, 2)	2002

Tableau IV.2: Leukimia dataset

Total params: 56,740,330
Trainable params: 56,740,330
Non-trainable params: 0

Couches (Types)	***Sorties***	***Paramètres***
Conv1d_1 (Conv1D)	(None, 7127, 16)	64
Conv1d_2 (Conv1D)	(None, 7125, 32)	1568
Max_pooling1d_1	(None, 3562, 32)	0
Conv1d_3 (Conv1D)	(None, 3560, 64)	6208
Max_pooling1d_2	(None, 1780, 64)	0
Conv1d_4 (Conv1D)	(None, 1778, 32)	6176
Max_pooling1d_3	(None, 889, 32)	0
Conv1d_5 (Conv1D)	(None, 887, 128)	12416
Max_pooling1d_4	(None, 443, 128)	0
Dropout (Dropout)	(None, 443, 128)	0
Flatten3 (Flatten)	(None, 56704)	0
Dense (Dense)	(None, 1000)	56705000
Dense (Dense)	(None, 2)	2002

Tableau IV.3: CNS dataset

Total params: 56,733,434
Trainable params: 56,733,434
Non-trainable params: 0

Pour l'exécution de nos modèles, nous avons utilisée trois paramètres essentiels : optimiseur, perte et métriques.

- ***Adam*** : est utilisé comme optimiseur car il s'agit généralement d'un bon optimiseur à utiliser dans de nombreux cas. L'optimiseur contrôle et ajuste le taux d'apprentissage tout au long de la formation.

- ***binary_crossentropy*** : est utilisé comme fonction de coût car c'est le choix le plus courant pour la classification. Plus le score est faible, plus le modèle est performant.

- ***La métrique de précision*** utilisée pour mieux interpréter et voir le score de précision sur la validation et la formation.

Pour La formation du modèle, la fonction fit () est utilisée avec les paramètres suivants :

- Données d'entrainement (X_train) avec les données cibles (y_train), qui sont obtenues en prenant 80% de données initiales.
- Données de validation (X_test) avec leurs données cibles (y_test) constituées de 20% de données restante.

L'entrainement du modèle ce fait en utilisant plusieurs nombre d'époques car il est courant d'entraîner le modèle plusieurs fois.

.

3.2 Résultats obtenus :

3.2.1 Colon dataset :

3.2.1.1 Modèle CNN 1D :

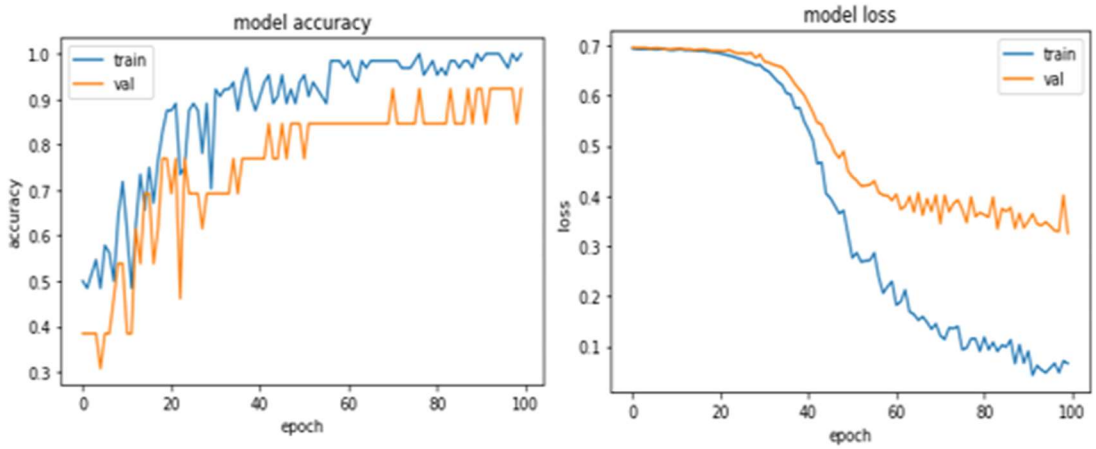


Figure IV.1 : Graph colon 1D

3.2.1.2 Discussion des graphes :

Comme nous pouvons le constater clairement, la précision de l'apprentissage sur les données de validation augmente avec le nombre d'époques, jusqu'elle atteindre une valeur maximal égale à 92,31%, ceci reflète qu'à chaque époque le modèle apprenne plus d'informations et la même chose pour l'ensemble de la validation. Le taux d'erreur d'apprentissage et sa validation est diminuée avec le nombre d'époques jusqu'à atteindre une valeur proche de 0.4.

On remarque un écart entre la précision sur les données d'entrainement et la précision sur les données de validation, qui nécessite un contrôle ou optimisation des hyper paramètres du modèle et introduction de plus de techniques de régularisation pour améliorer le modèle et réduire cet écart entre les deux taux de classification.

3.2.2 Modèles CNN 2D :

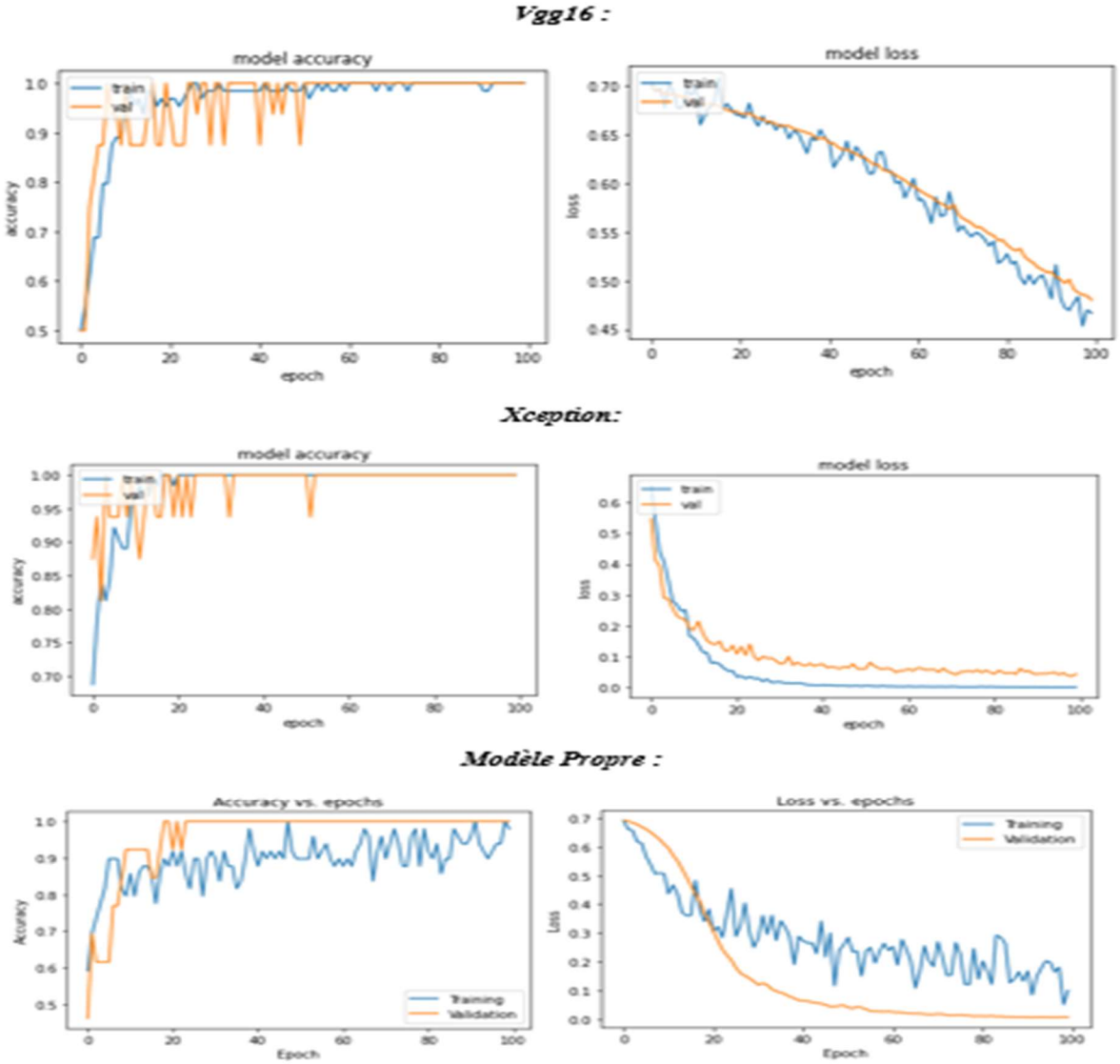


Figure IV.2 : Graph colon 2D

3.2.2.1 Discussion des graphes :

D'après la Figure IV.2 on observe que :

Le modèle ***Vgg16*** atteint une précision égale à 100%, qui s'augmente avec le nombre d'époque jusqu'elle devient stable à l'époque 51, ainsi que l'erreur d'apprentissage est diminuée jusqu'à la valeur 10,78%.

Le modèle ***Xception*** atteint une précision égale à 100%, qui s'augmente avec le nombre d'époque jusqu'elle devient stable à l'époque 25, ainsi que l'erreur d'apprentissage est diminuée jusqu'à la valeur0,0433.

Le modèle ***Propre*** atteint une précision égale à 98%, qui s'augmente avec le nombre d'époque, ainsi que l'erreur d'apprentissage est diminuée jusqu'à la valeur 0.

3.3 Prostate dataset :

3.3.1 Modèle CNN 1D :

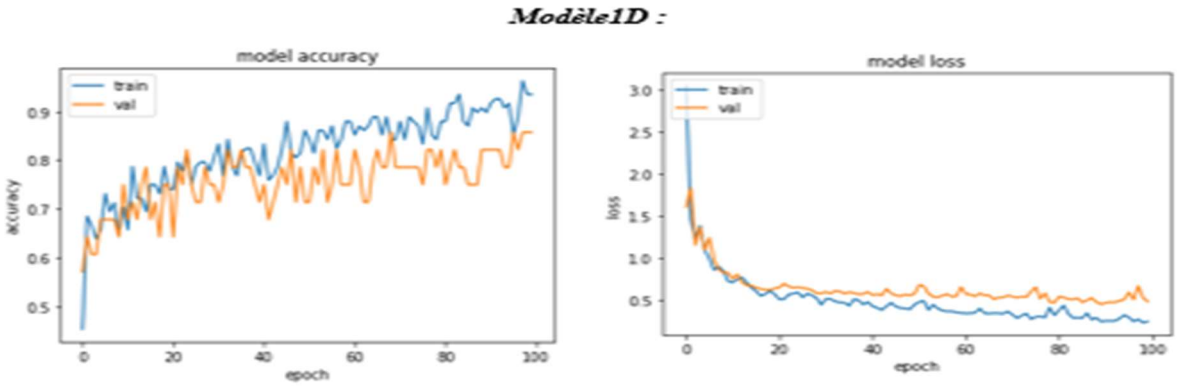


Figure IV.3 : Graph prostate 1D

3.3.1.1 Discussion des graphes :

Comme nous pouvons le constater, la précision de l'apprentissage augmente avec le nombre d'époque, jusqu'elle atteindre une valeur maximal égale à 85,31%, ceci reflète qu'à chaque époque le modèle apprenne plus d'information et la même chose pour l'ensemble de la validation. Le taux d'erreur d'apprentissage et sa validation est diminuée avec le nombre d'époque jusqu'à atteindre une valeur égale à 0.22.

3.3.2 Modèle 2D :

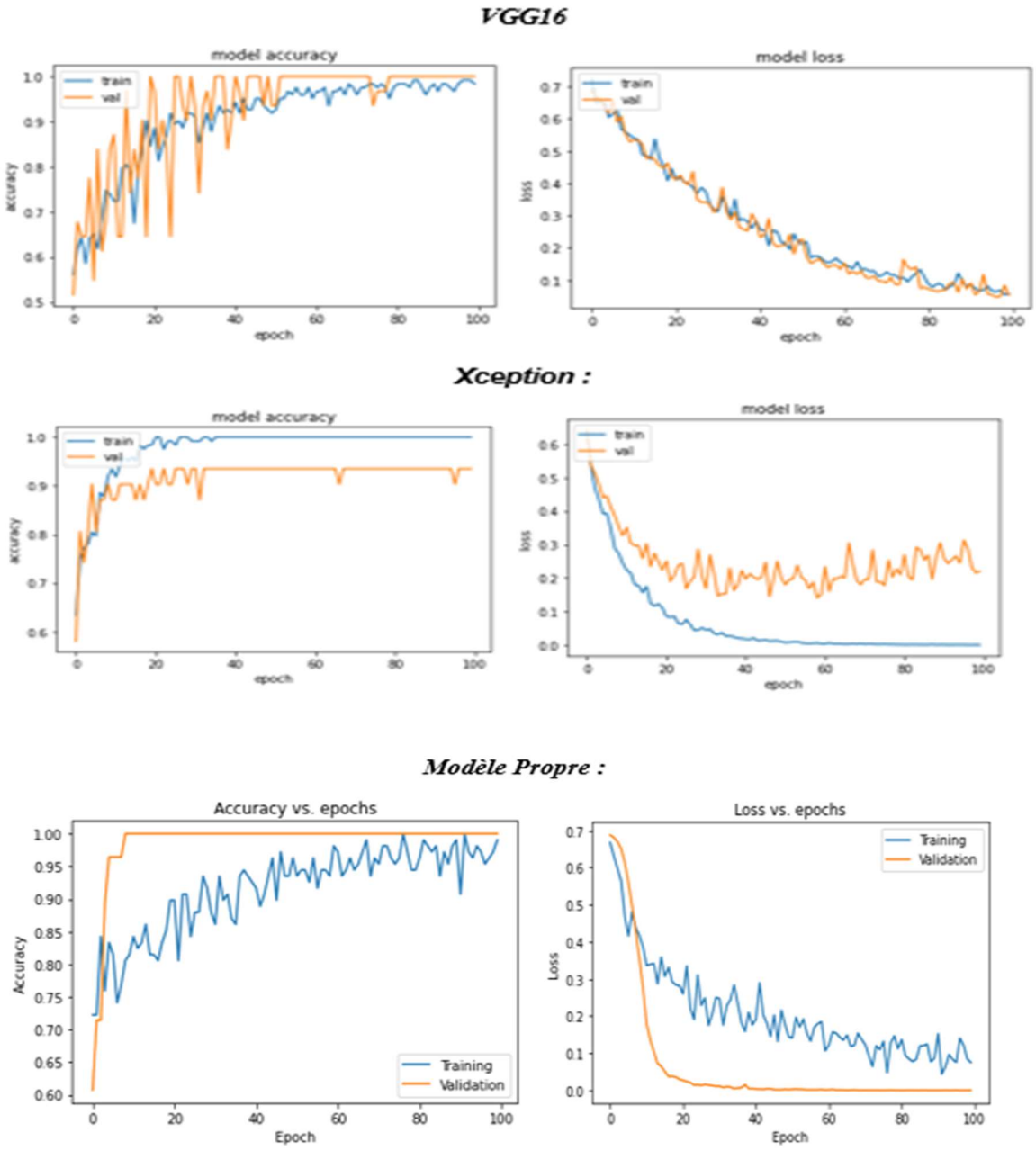


Figure IV.4 : Graph prostate 2D

3.3.2.1 Discussion des graphes :

D'après la Figure IV.4 on observe que : Le modèle ***VGG16*** atteint une précision égale à 98%, qui s'augmente d'une manière perturbé avec le nombre d'époques, ainsi que l'erreur d'apprentissage est diminuée jusqu'à la valeur 0 .542.

Le modèle ***Xception*** atteint une précision égale à 93%, qui s'augmente avec le nombre d'époque jusqu'elle devient stable à l'époque 36, ainsi que l'erreur d'apprentissage est diminuée jusqu'à la valeur 0.433%.

Le modèle ***Propre*** atteint une précision égale à 98%, qui s'augmente avec le nombre d'époque, ainsi que l'erreur d'apprentissage est diminuée jusqu'à la valeur 0,06.

3.4 Leukimia dataset :

3.4.1 Modèle CNN 1D :

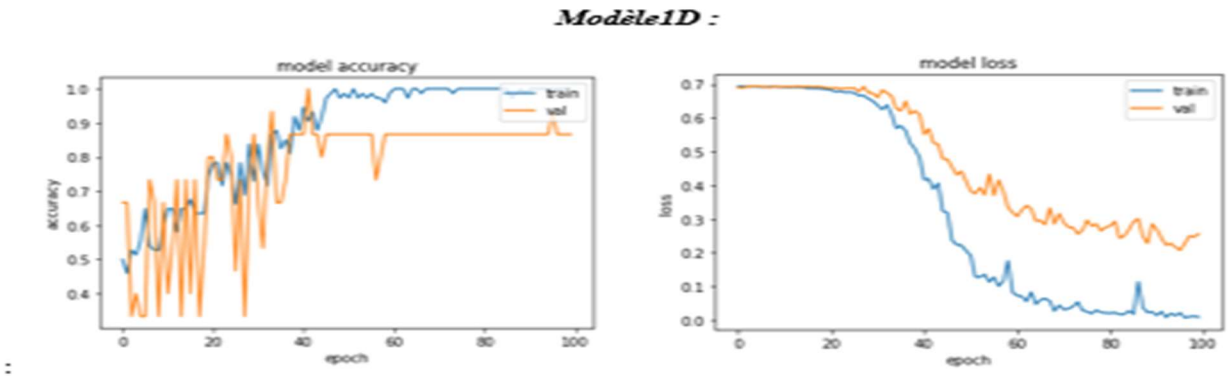


Figure IV.5 : Graph Leukimia 1D

3.4.1.1 Discussion des graphes :

Comme nous pouvons le constater clairement, la précision de l'apprentissage augmente avec le nombre d'époque, jusqu'elle atteindre une valeur maximal égale à 86,66%, ceci reflète qu'à chaque époque le modèle apprenne plus d'information et la même chose pour l'ensemble de la validation. Le taux d'erreur d'apprentissage et sa validation est diminuée avec le nombre d'époque jusqu'à atteindre une valeur égale à 0.24.

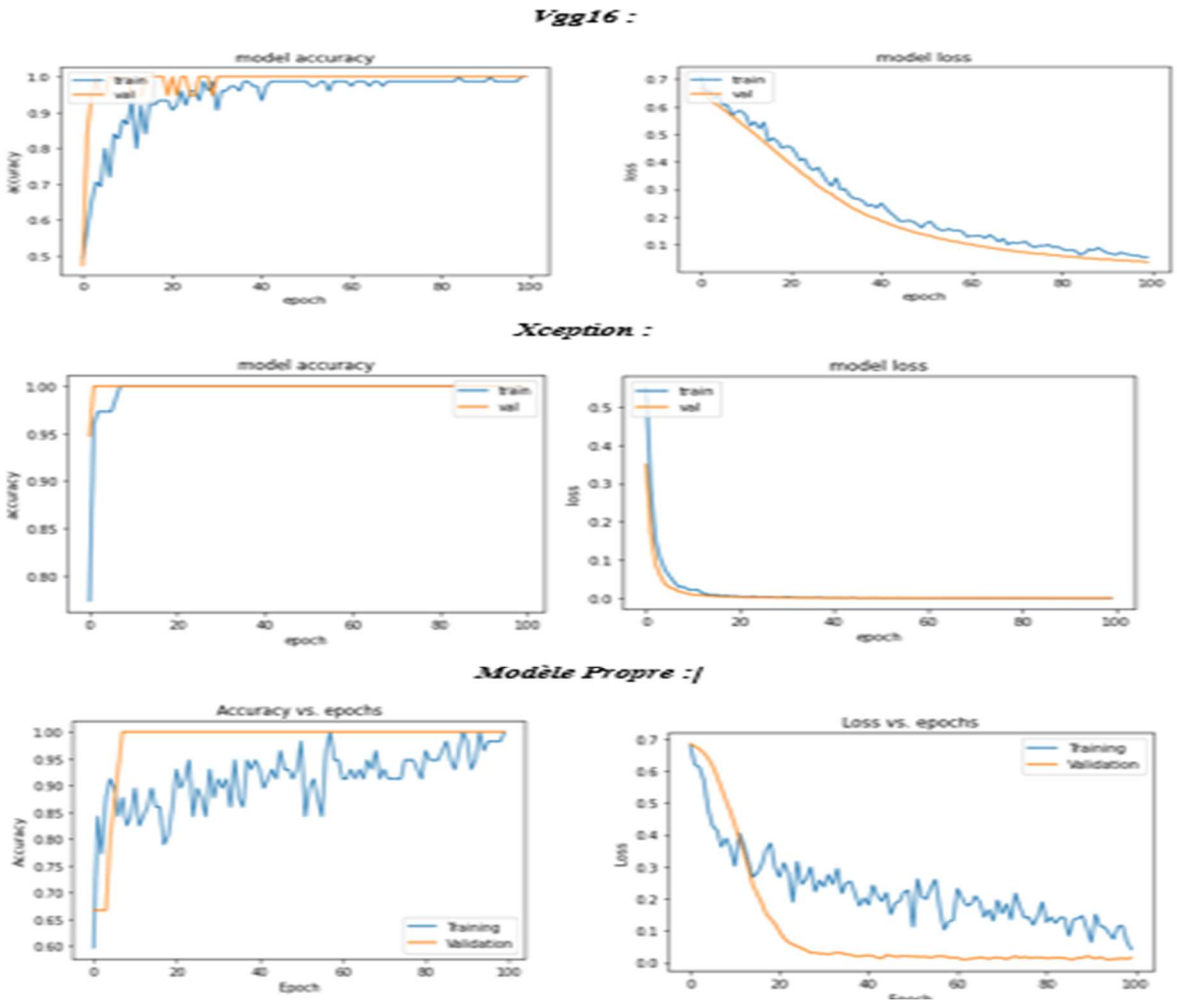


Figure IV.6 : Graph Leukimia 2D

3.4.1.2 Discussion des graphes :

D'après la Figure II.6 on observe que : Le modèle ***Vgg16*** atteint une précision égale à 100%, qui s'augmente avec le nombre d'époque jusqu'elle devient stable à l'époque 31, ainsi que l'erreur d'apprentissage est diminuée jusqu'à la valeur 0.336.

Le modèle ***Xception*** atteint une précision égale à 100%, qui s'augmente avec le nombre d'époque jusqu'elle devient stable à l'époque 2, ainsi que l'erreur d'apprentissage est diminuée jusqu'à la valeur 0.

Le modèle ***Propre*** atteint une précision égale à 100%, qui s'augmente avec le nombre d'époque, ainsi que l'erreur d'apprentissage est diminuée jusqu'à la valeur 0.0415

3.5 CNS Dataset :

3.5.1 Modèle CNN 1D :

Modèle1D :

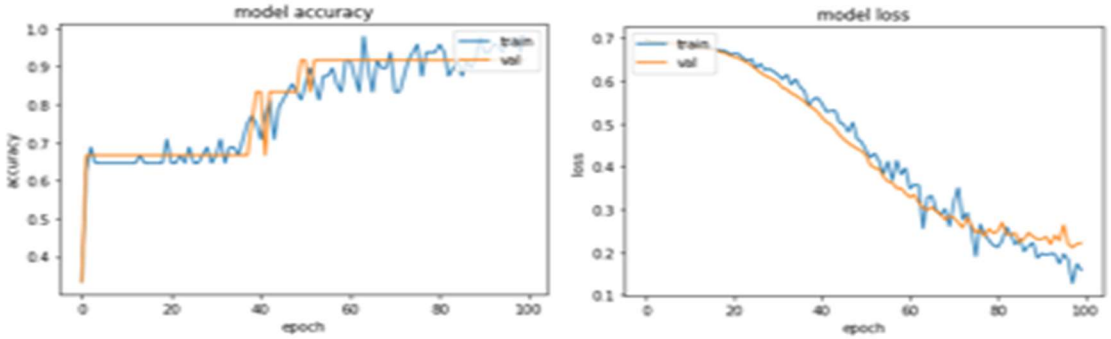


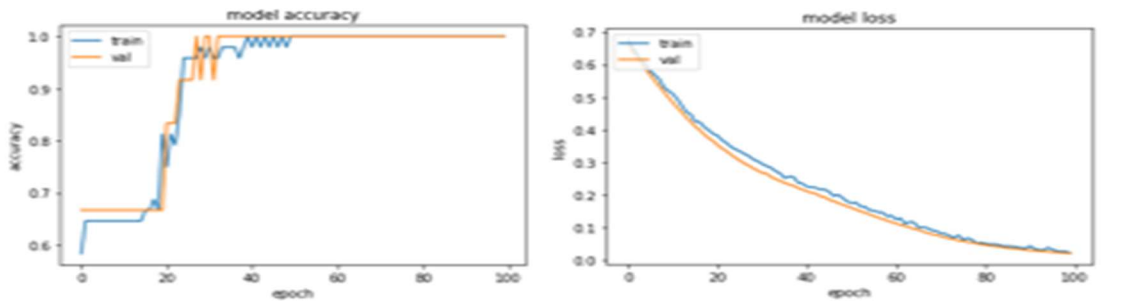
Figure IV.7 : Graph CNS 1D

3.5.1.1 Discussion des graphes :

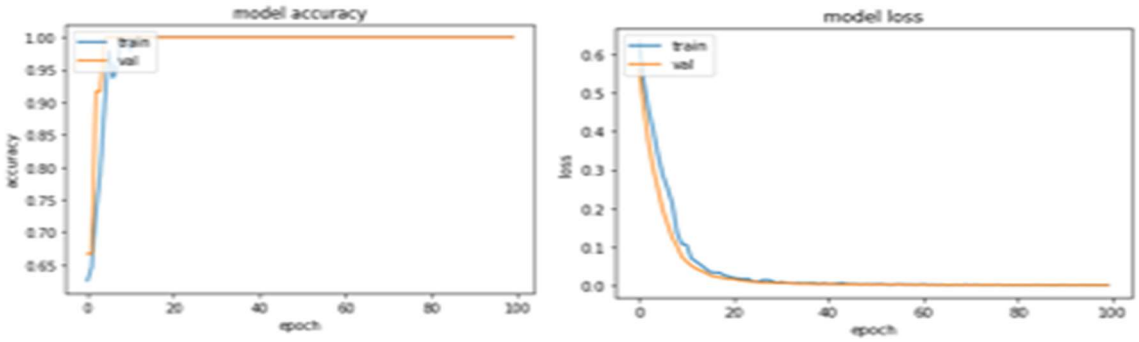
Comme nous pouvons le constater clairement, la précision de l'apprentissage augmente avec le nombre d'époque, jusqu'elle atteindre une valeur maximal égale à 91,67%, ceci reflète qu'à chaque époque le modèle apprenne plus d'information et la même chose pour l'ensemble de la validation. Le taux d'erreur d'apprentissage et sa validation est diminuée avec le nombre d'époque jusqu'à atteindre une valeur égale à 0 .22.

3.5.2 Modèle CNN 2D :

Vgg16 :



Xception :



Modèle propre :

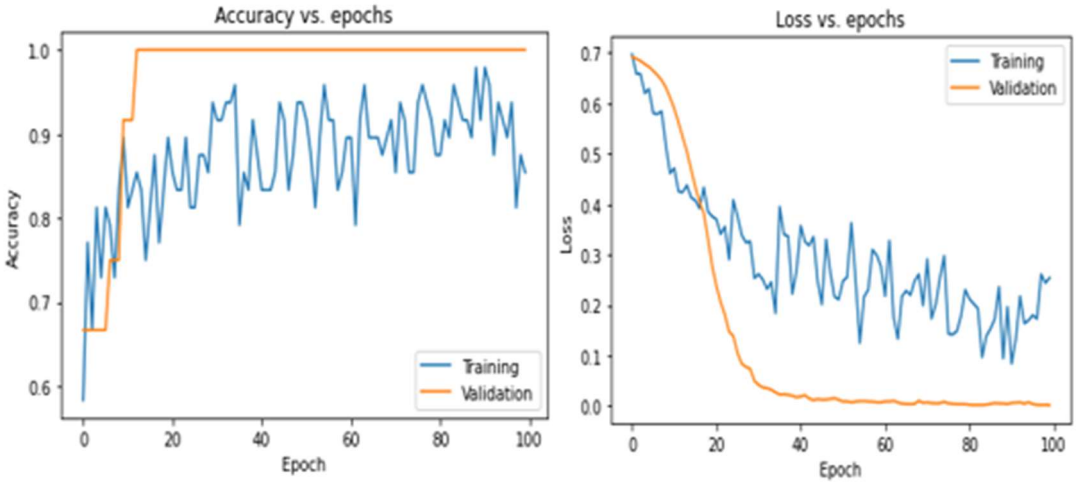


Figure IV.8 : Graph CNS 2D

3.5.2.1 Discussion des graphes :

D'après la Figure IV.8 on observe que : Le modèle ***Vgg16*** atteint une précision égale à 100%, qui s'augmente avec le nombre d'époque jusqu'elle devient stable à l'époque 50, ainsi que l'erreur d'apprentissage est diminuée jusqu'à la valeur 0.198.

Le modèle ***Xception*** atteint une précision égale à 100%, qui s'augmente avec le nombre d'époque jusqu'elle devient stable à l'époque 5, ainsi que l'erreur d'apprentissage est diminuée jusqu'à la valeur 0.

Le modèle ***Propre*** atteint une précision égale à 98%, qui s'augmente avec le nombre d'époque, ainsi que l'erreur d'apprentissage est diminuée jusqu'à la valeur 0.25.

4 Comparaison avec les classicateurs classiques :

4.1 Machine à vecteurs de support (SVM) :

Les machines à vecteurs de support ou séparateurs à vaste marge (en anglais Support Vector Machine, SVM) sont un ensemble de techniques d'apprentissage supervisé destinées à résoudre des problèmes de classification. les SVM sont une généralisation des classifiées linéaires. Les SVM ont été développés dans les années 1990 à partir des considérations théoriques de Vladimir Vapnik sur le développement d'une théorie statistique de l'apprentissage : la Théorie de Vapnik Chervonenkis. Les SVM ont été appliqués à de très nombreux domaines (bio-informatique, recherche d'information, vision par ordinateur, finance...). Selon les données, la performance des machines à vecteurs de support est de même ordre, ou même supérieure, à celle d'un réseau de neurones ou d'un modèle de mixture gaussienne. Hyperplan qui est le lieu des points x satisfaisant w*x + b = 0. En orientant l'hyperplan, la règle de décision correspond à observer de quel côté de l'hyperplan se trouve l'exemple x. On voit que le vecteur w définit la pente de l'hyperplan (w est perpendiculaire à l'hyperplan). Le terme b quant à lui permet de translater l'hyperplan parallèlement à lui-même.

4.2 Random forest:

Random forest est un algorithme d'apprentissage automatique supervisé qui est construit à partir d'algorithmes d'arbre de décision, développé par ***Leo Breiman*** pour des raisons de classification ou de régression, Il se compose d'un grand nombre d'arbres de décision individuels qui fonctionnent comme un ensemble, Chaque arbre individuel dans la forêt aléatoire crache une prédiction de classe et la classe avec le plus de votes devient la prédiction finale de modèle, Il fonctionne en quatre étapes :

1. Sélectionnez des échantillons aléatoires à partir d'un ensemble de données donné.
2. Construisez un arbre de décision pour chaque échantillon et obtenez un résultat de prédiction à partir de chaque arbre de décision.
3. Effectuez un vote pour chaque résultat prévu.

Les résultats des classifieurs classiques sont des résultats des expériences récentes qui ont était testé sur les même ensembles de donnée. Ils sont mentionnés dans l'article (On the classification techniques in datamining for miccroarray data classification) **[32]**

Modèle / Dataset	RF % (100 tree)	SVM %	VGG16 %	XCEPTION %	2D simple %	1D simple %
Colon	**75.00**	**80.27**	**100**	**100**	**100**	**92,31**
CNS	**41.67**	**65.00**	**100**	**100**	**99**	**91,67**
Leukemia	**80.00**	**72.50**	**100**	**100**	**100**	**86,66**
Prostate	**68.89**	**62.57**	**98**	**93**	**100**	**85,31**
Moyenne(accuracy)	**66,39**	**70,085**	**99,5**	**98,25**	**99,25**	**88,9875**

Tableau IV.4: Comparaison de l'accuracy entre les modèles

4.3 Discussion:

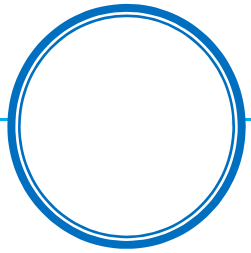
D'après le tableau précédent, nous remarquons que la performance du modèle VGG16 obtenue sur nos bases de données est de 99,5 %, dont il a une meilleure performance par rapport à la performance du modèle Xception qui est de 98,25% sur les mêmes bases, bien que le nombre de paramètres traité soient similaires. La performance du modèle CNN simple obtenue sur nos bases 98 % relativement faible par rapport aux Vgg16 et Xception. Mais ça reste un résultat honorable qui stimule la recherche et le développement dans ce domaine.

Pour le modèle 1D simple, nous remarquons que sa performance sur les bases de données est 88,9875%, qui restent faible par rapport à la performance des modèles de dimensions 2D. Ce qui indique que les meilleurs résultats sont obtenus lors de l'application de la méthode DeepInsight avec une extraction des caractéristique précise grâce à l'apprentissage par transfert qui permet d'utiliser des modèles déjà entrainer avec des poids ajustés.

D'après les résultats des classifieurs classiques qui sont mentionné dans le tableau précédent, on remarque que la performance des algorithmes d'apprentissage automatique (SVM et RandomForest) obtenue sur les mêmes bases de données (la moyenne sur les 4 baes est 70,085 et 66,39 respectivement). A partir de ces résultats on constate que les modèles d'apprentissage profond appliqués sur les ensembles de données utilisés sont plus adaptés à la classification des données issues des puces d'ADN.

5 Conclusion :

Dans ce chapitre, nous avons décrit l'environnement du travail, les différents outils et les bibliothèques utilisés pour l'implémentation de notre approche, ensuite nous avons présenté les graphs des résultats avec des commentaires, ainsi qu'une comparaison entre les différents modèles appliqués et les modèles classiques de l'apprentissage automatique.



Conclusion Générale

Conclusion générales :

Dans cette thèse, nous avons présenté une approche qui sert à détecter la maladie du cancer de plusieurs types (colon, prostate, sang, et le système nerveux) en utilisant la technologie des puces à ADN en s'appuyant sur l'apprentissage profond. Pour bien comprendre la problématique nous avons présenté la technologie des puces à ADN en détaillons leur structure et leur mode de fonctionnement, ainsi que leur apport dans le domaine de la cancérologie.

Après avoir compris le problème, nous sommes passés à la réalisation d'une approche qui se base principalement sur les réseaux de neurones convolutif CNN en utilisant l'apprentissage par transfert, pour l'implémentation des modèles à deux dimensions en choisissant les architectures VGG16 et Xception, ces dernières ont été appliqué sur des images obtenues par la méthode Deepinsight. Egalement on a construit un modèle d'une seule dimension qui se compose des couches de convolutions et de MaxPooling. Pour évaluer ces modèles nous avons choisi la précision et le taux d'erreur comme des mesures d'évaluation.

Les résultats obtenus nous ont permis de conclure que : les modèles convolutionnels CNN sont plus performants dans la classification des données des puces à ADN, tout ça pour faire une classification des échantillons dans deux catégories (malade / non_malade), Contrairement aux classificateurs classiques de l'apprentissage automatique qui ont donné des résultats moins efficaces et non performants.

Le Thème qui nous a été attribué est très instructif sur le plan pédagogique et très intéressant, sur le plan technologique. Il nous a permis d' :

- Accroitre nos connaissances dans le domaine de l'apprentissage.
- Initier sur différentes notions d'implémentation d'un réseau de neurones convolutifs, ainsi l'utilisation de nouvelles méthodes comme la méthode DeepInsight.
- Améliorer nos compétences sur l'utilisation de l'environnement de travail GoogleColab.

Comme perceptive il est à noter, que nous avons utilisé ces modèles sur des petites bases de données des expressions génétiques (de colon, prostate, CNS et leucémie). Il sera souhaitable dans l'avenir de les appliquer sur des grandes datasets qui contiennent un nombre d'échantillons important. Egalement il est désirable d'utiliser la méthode Deepinsight avec d'autres architectures de l'apprentissage profond comme les auto-encodeurs.

Bibliographies

[1] E. B. Huerta, “L7XJK - pré-traitement de données de biopuces et la sélection de de données de biopuces et la sélection de gènes Edmundo Bonilla Huerta To cite this version :,” 2010.

[2] T. R. GOLUB et al, "Molecular Classification of Cancer: Class Discovery and Class Prediction by Gene Expression Monitoring”, SCIENCE, 1999,Vol 286, Issue 5439.

[3] S. H. Shah, M. J. Iqbal, I. Ahmad, S. Khan, and J. J. P. C. Rodrigues, “Optimized gene selection and classification of cancer from microarray gene expression data using deep learning,” *Neural Comput. Appl.*, vol. 8, 2020, doi: 10.1007/s00521-020-05367-8.

[5] D. Q. Zeebaree, H. Haron, and A. M. Abdulazeez, “Gene Selection and Classification of Microarray Data Using Convolutional Neural Network,” *ICOASE 2018 - Int. Conf. Adv. Sci. Eng.*, no. December 2018, pp. 145–150, 2018, doi: 10.1109/ICOASE.2018.8548836.

[6] J.-P. Vert, “Les applications industrielles de la bioinformatique,” *Ann. des Mines - Réalités Ind.*, vol. Février 20, no. 1, p. 17, 2013, doi: 10.3917/rindu.131.0017.

[7] “🔎 Bio-informatique - Définition et Explications.” https://www.techno-science.net/glossaire-definition/Bio-informatique.html#ref_1 (accessed May 29, 2021).

[8] “Puce à ADN — Wikipédia.” https://fr.wikipedia.org/wiki/Puce_à_ADN (accessed May 29, 2021).

[9] U. Blaise and P. Clermont, “Ecole Doctorale des Sciences pour l ’ Ingénieur Thèse Docteur d ’ Université Conception et analyse des biopuces à ADN en environnements parallèles et distribués,” 2014.

[11] “Applications de puce ADN.” https://www.news-medical.net/life-sciences/Microarray-Aplications-(French).aspx (accessed May 29, 2021).

[15] J. P. Feugeas, “Apport des puces à ADN en cancérologie,” *Bio Trib. Mag.*, vol. 39, no. 1, pp. 14–18, Oct. 2011, doi: 10.1007/s11834-011-0053-9.

[19] "Réseaux de neurones récurrents - Data Analytics Post." https://dataanalyticspost.com/Lexique/reseaux-de-neurones-recurrents/ (accessed Sep. 06, 2021).

[22] "Implementation of GoogLeNet on Keras | by Khuyen Le | MLearning.ai | Medium." https://medium.com/mlearning-ai/implementation-of-googlenet-on-keras-d9873aeed83c (accessed Sep. 07, 2021).

[23] F. Chollet, "Xception: Deep Learning with Depthwise Separable Convolutions."

[24] "VGG Neural Networks: The Next Step After AlexNet | by Jerry Wei | Towards Data Science." https://towardsdatascience.com/vgg-neural-networks-the-next-step-after-alexnet-3f91fa9ffe2c (accessed Sep. 07, 2021).

[25] "A Gentle Introduction to Transfer Learning for Deep Learning." https://machinelearningmastery.com/transfer-learning-for-deep-learning/ (accessed Sep. 07, 2021).

[26] C. C. Aggarwal, "Educational and software resources for data classification," *Data Classif. Algorithms Appl.*, pp. 657–665, 2014, doi: 10.1201/b17320.

[27] B. Krawczyk, "Learning from imbalanced data: open challenges and future directions," *Prog. Artif. Intell.*, vol. 5, no. 4, pp. 221–232, Nov. 2016, doi: 10.1007/S13748-016-0094-0.

[28] A. Sharma, E. Vans, D. Shigemizu, K. A. Boroevich, and T. Tsunoda, "DeepInsight: A methodology to transform a non-image data to an image for convolution neural network architecture," *Sci. Rep.*, vol. 9, no. 1, pp. 1–7, Dec. 2019, doi: 10.1038/s41598-019-47765-6.

[29] "T-SNE - Data Analytics Post." https://dataanalyticspost.com/Lexique/t-sne/?fbclid=IwAR1NYNS3S0mQ3EsQZS-s68yuPVuXY1blyVCg4W92AA3-KkBNPMONVzXtjMc (accessed Aug. 01, 2021).

[30] "Why Do We Split Datasets?. Why it is important to create a… | by Seyma Tas | Medium." https://seymatas.medium.com/why-do-we-split-datasets-55c46964fd84 (accessed Aug. 01, 2021).

[31] "How to split a Dataset into Train and Test Sets using Python | Towards Data Science."

https://towardsdatascience.com/how-to-split-a-dataset-into-training-and-testing-sets-b146b1649830 (accessed Aug. 01, 2021).

[32] Husna Aydadenta and Adiwijaya "On the classification techniques in data mining for microarray data classification," *Journal of Physics: Conference Series PAPER*,2018 Phys.: Conf. Ser. 971 012004

Webographie :

[4] "Cancer." https://www.who.int/fr/news-room/fact-sheets/detail/cancer (accessed Sep. 05, 2021).

[10] "Puce adn microarray methode analyse genomique fonctionnelle vegetale Enseignement et recherche Biochimie - Universite Angers Emmanuel Jaspard." http://biochimej.univangers.fr/Page2/COURS/9ModulGenFoncVeg/5MethEtudGenFonc/3PucesADN/1PucesADN.htm (accessed May 29, 2021).

[12] "Définition | Cancer | Futura Santé." https://www.futura sciences.com/sante/definitions/medecine-cancer-108/ (accessed May 29, 2021).

[13] "Algérie : Prévalence du cancer en Algérie. - Business France." https://www.businessfrance.fr/algerie-prevalence-du-cancer-en-algerie (accessed Sep. 06, 2021).

[14] "GÉNIE GÉNÉTIQUE, Utilisation des puces à ADN - Encyclopædia Universalis." https://www.universalis.fr/encyclopedie/genie-genetique/6-utilisation-des-puces-a-adn/ (accessed May 29, 2021).

[16] "Définition | Deep Learning - Apprentissage profond | Futura Tech." https://www.futura-sciences.com/tech/definitions/intelligence-artificielle-deep-learning-17262/ (accessed Sep. 06, 2021).

[17] "Convolutional neural network | Deep Learning | DataScientest." https://datascientest.com/convolutional-neural-network#maxpooling (accessed Jun. 02, 2021).

[18] "Les auto-encodeurs expliqués en détail : JAFWIN." https://jafwin.com/2020/07/04/les-auto-encodeurs-expliques-en-detail/ (accessed Sep. 06, 2021).

[19] "Réseaux de neurones récurrents - Data Analytics Post." https://dataanalyticspost.com/Lexique/reseaux-de-neurones-recurrents/ (accessed Sep. 06, 2021).

[20] "Réseaux de neurones récursifs (RvNN) et réseaux de neurones récurrents (RNN)."

https://ichi.pro/fr/reseaux-de-neurones-recursifs-rvnn-et-reseaux-de-neurones-recurrents-rnn-52736299609345 (accessed Sep. 06, 2021).

[21] "Convolutional neural network | Deep Learning | DataScientest." https://datascientest.com/convolutional-neural-network (accessed Sep. 07, 2021).

[22] https://mobiskill.fr/blog/conseils-emploi-tech/apprentissage-supervise-vs-apprentissage-non-supervise/