Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

UNIVERSITÉ DE MOULOUD MAMMERIE DE TIZI OUZOU

Faculté : Génie électrique et informatique
Département : Informatique

Mémoire de fin d'études

En vue de l'obtention du diplôme de Master en Informatique

Spécialité : Système informatique

THÈME

Un modèle d'apprentissage automatique pour la prédiction des maladies, cas d'étude : cancer du sein

Présenté par :
Mesbahi Kamel, Aitouakli Thanina

Dirigé par : M.
Chebouba Lokmane

∗ Membres du jury :

— Mr.Arezki HAMMACHE	Président du Jury
— Mr.Fayçal Redha SAIDANI	Examinateur
— Mr. Lokmane CHEBOUBA	Promoteur

Date de soutenance : 22/09/2020

Dédicaces

Je dédie ce travail à :

Mes parents, **frères et soeurs**, pour m'avoir offert une éducation digne, un soutien sans faille et surtout, pour m'avoir appris que l'on peut cultiver un sens moral bien au-delà de sa condition.
Je marche sur chacune de leurs traces pour espérer un jour, être digne de leurs parcours.

Thanina

Dédicaces

Avec l'expression de ma reconnaissance,
je dédie ce modeste travail :

À la mémoire de mon père.
À ma famille, qui m'a doté d'une éducation digne, son amour a fait de moi ce que je suis aujourd'hui.

À mes amis et mes collègues qui m'ont chaleureusement supporté et encouragé tout au long de mon parcours, et à qui je souhaite plus de succès.
À tous ceux que j'aime et ceux qui m'aiment.

Kamel

Remerciement

Nous tenons à exprimer toute notre gratitude à notre encadreur **Dr. Chebouba Lokmane** pour ses conseils, ses directives et son aide.

Nos remerciements s'adressent aux membres du **jury** qui ont accepté d'évaluer notre travail. Nous tenons à exprimer nos sincères remerciements à tout le corps professoral et administratif de l'université Mouloud Mammeri de Tizi-Ouzou.

Résumé

L'avancement parallèle de la médecine et de l'informatique a pu prodiguer des techniques révolutionnaires dans la localisation et traitement de pathologies graves toutefois, le cancer reste à ce jour un diagnostic difficile malgré la précision des appareils d'imagerie médicale. La solution qui se présente de plus en plus est de mettre en oeuvre une autre alternative informatique qui est l'intelligence artificielle au service de la détection de cancers.
Notre travail consiste à étudier en profondeur les méthodes les plus performantes de l'apprentissage automatique afin de les appliquer dans un domaine aussi délicat que la medecine. Parmi ces méthodes, on cite les réseaux de neurones complétement connectés et les réseaux de neurones convolutifs, il s'agit d'une classification supervisée : les différentes versions de notre modèle prennent en entrée une base de données étiquetée d'images (histologiques) de cancers suspectés du sein et founrnie une prédiction relative aux quatres catégories de diagnostic à savoir : normal, bénin, in situ, invasif.
Nos résultats les plus importants se portent sur la précision obtenue pour la classification de cancers du sein en appliquant les techniques visées.
La portée et la validité de nos travaux résident dans le calcul automatique de son taux de réussite mais également à l'application éventuelle de notre modèle sur le terrain par des spécialistes.

Mots clés : apprentissage automatique (machine learning), apprentissage supervisé, classification, réseaux de neurones, réseaux de neurones convolutifs, entraînement, tensorflow, Keras.

Abstract

The parallel advancement of medicine and computer science has been able to provide revolutionary techniques in the localization and treatment of serious pathologies, however, cancer remains to this day a difficult diagnosis despite the precision of medical imaging devices. One of the solutions is to implement another computing alternative which is the artificial intelligence at the service of the detection of cancers.
Our job is to study in depth the most powerful methods of machine learning in order to apply them in a field as delicate as medicine. Among these methods, we cite fully connected neural networks and convolutional neural networks, this is a supervised classification : the different versions of our model take as input a labeled database witch is histological images of suspected breast cancers and provide a prediction relating to the four diagnostic categories : benign, invasive, in situ, normal.
Our most important results is the accuracy obtained for the classification of breast cancers by applying the selected techniques.
The scope and validity of our work is in the automatic calculation of it's accuracy but also in the direct application of our model by specialists.

Keywords : machine learning, supervised learning, classification, neural networks, convolutional neural networks, training, tensorflow, Keras.

Liste des abréviations

IA	Artificial Intelligence
ADN	Acide Désoxyribonucléique
ML	Machine Learning
DNN	Dense/Deep Neural Network
CNN	Convolutional neural network
RMSprop	Root Mean Square Propagation
ADAM	Adaptive Moment Estimation
SGD	Stochastic gradient descent
PET-scan	Positron emission tomography Scanner

Glossaire

Vaisseaux lymphatiques :
Les vaisseaux lymphatiques font circuler la lymphe, un liquide biologique comparable au sang, contenant des globules blancs, mais dépourvu de globules rouges.

Ultrasons :
Vibration similaire au son mais de fréquence beaucoup plus grande. On utilise les ultrasons comme outil de diagnostic.

Produit radioactif :
Se dit d'un corps ayant la propriété d'émettre des particules ou des rayonnements.

Loie de Moore :
Cofondateur de Intel, Gordon Moore avait affirmé dès 1965 que le nombre de transistors par circuit de même taille allait doubler, à prix constants, tous les ans. Il rectifia par la suite en portant à dix-huit mois le rythme de doublement.

Cytoplasme :
Partie fondamentale de la cellule qui contient le noyau.

Nodules :
Formation anormale, généralement arrondie et de petite taille, cancéreuse ou non, dans un organe ou à sa surface.

Fonction convexe :
En mathématiques, une fonction est convexe si celle-ci comporte un minimum global.

Lésion :
Terme médical qualifiant l'altération d'une cellule, tissu (ensemble de cellules) ou d'un organe vivant.

Immunohistochimie :
L'immunohistochimie est une technique de localisation de protéines dans les cellules d'un tissu, permettant notamment de détecter les cancers.

Table des matières

Introduction générale

Selon les statistiques récentes (datant de 2019), la hausse annuelle du taux de cancers du sein en Algerie est de 8 % équivalent à 25.000 cas par an. Les inquiétudes se portent sur la tranche d'âge touchée qui est de 20 à 30 ans, or il est prouvé que le cancer du sein atteint beaucoup plus les femmes âgées[1].

Context de travail

Le problème que nous avons choisi d'étudier a été organisé dans le cadre de la $15^{\text{ème}}$ conférence internationale sur l'analyse et la reconnaissance d'images (ICIAR 2018), elle résulte d'une coopération entre l'université de Porto, l'institut des systèmes et de l'ingénierie informatique, technologie et sciences (INESC TEC) et l'institut de recherche et d'innovation en santé (i3S), Portugal. L'équipe dans son ensemble possède une grande expérience en apprentissage automatique et en vision par ordinateur, une expertise médicale et une expérience dans l'organisation des défis précédents[2].

L'intelligence artificielle est un domaine qui nous facine, l'idée de créer une intelligence supérieure à la nôtre est tant paradoxale qu'inspirante de plus, plusieurs méthodologies de l'apprentissage automatique se fondent sur le fonctionnement de nos neurones, nos yeux...etc ; mettre en oeuvre un modèle ML inspiré de la biologie afin de résoudre un problème de pathologie biologique est captivant.

Problématique

Plusieurs problématiques sont à mettre en évidence dans le cadre de notre travail.
Premièrement, du côté du cancer du sein, la difficulté de diagnostic d'un cancer présente une fatalité, les spécialistes n'arrivent pas toujours à distinguer un cancer dans des résultats d'analyse, nous tenterons de résoudre cette problématique par l'IA.
Ensuite, il faudra trouver un modèle ML qui répondra aux contraintes imposées par notre thème : la médecine étant un domaine critique qui relève de la vie d'un humain, nous devons effectuer une étude méticuleuse des méthodes à utiliser afin d'obtenir une solution plus que fiable.
En outre, entraîner une intelligence artificielle à reconnaître une image requière des capacités de calcul de l'équivalent d'un serveur, il est nescessaire de trouver une alternative afin d'exploiter une puissance de calcul externe. Pour finir, les données mises à disposition par la compétition sont extrêmement volumineuses, il faudra déterminer des solutions afin de réduire les images sans perte d'informations pertinantes.

Méthodologie

Nous nous sommes orienté vers une méthodologie qui part de l'ancien au récent et du général au particulier.
les anciennes méthodes utilisées en intelligence artificielle ne sont plus d'actualité mais représentent une base solide afin d'initier notre étude, cela représente un axe général afin d'approfondir les outils mathématiques qui les régissent. Notre travail est basé sur des articles officiels et livres qui décrivent les plus grandes avancées de l'apprentissage automatique, les plus importants sont :

— *"Dive into deep learning" de [Alexander J. Smola]*, celui-ci nous a permis de comprendre les réseaux de neurones profonds.

— Différents articles de *[Yan Le cun]* sur les réseaux de neurones inspirés du cortex visuel.

— *"Machine learning yearning" de [Andrew Ng]* qui nous a aidé à comprendre et appliquer des techniques afin d'améliorer nos résultats.

Quant au cancer du sein, nous nous sommes documenté sur des cours et livres en médecine/biologie afin de comprendre cette maladie. Nous avons également questionné des spécialistes pour identifier la difficulté dans laquelle une IA peut apporter une aide considérable.

Contribution

Notre contribution dans le cadre de l'apprentissage supervisé se situe à plusieurs niveaux :

1. Réalisation d'un état l'art sur l'apprentissage automatique, différents réseaux de neurones et le cancer.
2. Construction de cinq modèles pour la classification de cancers du sein.
3. Entraînement et évaluation de ces modèles après les avoir implémentés.

Organisation du mémoire

Ce mémoire est articulé en trois parties englobant cinq chapitres en plus de l'introduction générale et de la conclusion générale.

La première partie comporte le premier chapitre où nous présentons le cancer du sein, ses différents types et caractéristiques sur image mais aussi les objectifs ayant déterminé le lancement de la compétition. Par la suite, nous présentons l'apprentissage automatique dans les trois chapitres suivants tout en détaillant les mathématiques qui actionnent la notion de réseaux de neurones et leurs applications dans la classification d'images. La troisième partie compte le cinquième chapitre où nous décrivons les modèles proposés, l'environnement, les outils utilisés ainsi que les résultats obtenus dans le cadre de la classification histologique d'un cancer du sein.

Chapitre 1

Le cancer du sein

introduction

Le cancer est une maladie des plus redoutable de l'histoire de l'humanité, elle ne cesse de mettre en épreuve la capacité de la science à la combattre puisqu'elle est dotée d'une particularité unique : celle-ci vient de nous-mêmes.
Au sein de ce chapitre, nous définirons le cancer en général et celui du sein en particulier sous ses différents types. Pour conclure, nous étudierons les étapes qui découlent de son diagnostic afin d'en déduire les motivations ayant déterminé l'introduction de l'IA au cœur ce domaine spécifique de la médecine.

✔Dans la perspective de comprendre fondamentalement le cancer, il est nécessaire de redéfinir la notion de cellule.

1.1 La cellule humaine

Une cellule constitue la partie la plus élémentaire de l'être. Comme illustré dans la figure ci-après, elle se présente sous forme de membrane enfermant un noyau où se loge L'ADN : c'est le manuel de production de protéines.

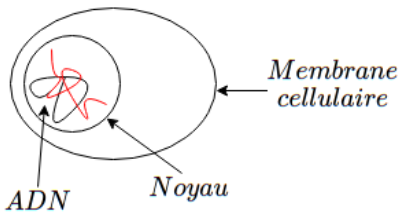


FIGURE 1.1 – Cellule humaine

1.2 La division cellulaire

Afin de se reproduire, une cellule se subdivise pour engendrer deux autres cellules filles. Cette opération est effectuée pour remplacer 100 milliards de cellules par jour chez l'homme. La figure suivante illustre ce processus [3].

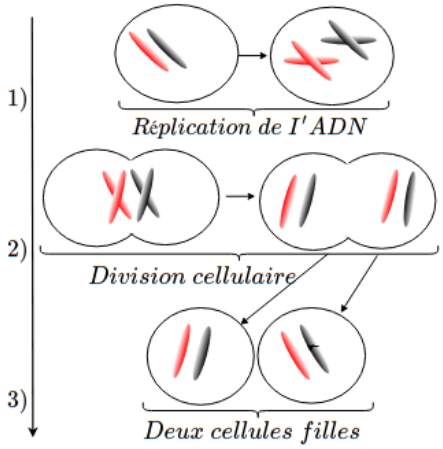


FIGURE 1.2 – Division cellulaire

✔ Au moment de la division, l'ADN est recopié à l'identique dans les deux autres cellules résultantes : c'est ici qu'un potentiel cancer peut se développer.

1.3 La division de cellules cancéreuses

Dans certains cas de reproduction, l'ADN se recopie faussement vers les cellules filles, c'est la mutation. Quand une cellule mute, elle s'autodétruit seulement, quand la molécule d'ADN est suffisamment altérée, la cellule perd sa capacité à s'éliminer et continue ainsi de proliférer en créant d'autres mutations formant un agglomérat de cellules qui grandit sans limite : c'est la tumeur[3]. La figure suivante illustre ce phénomène.

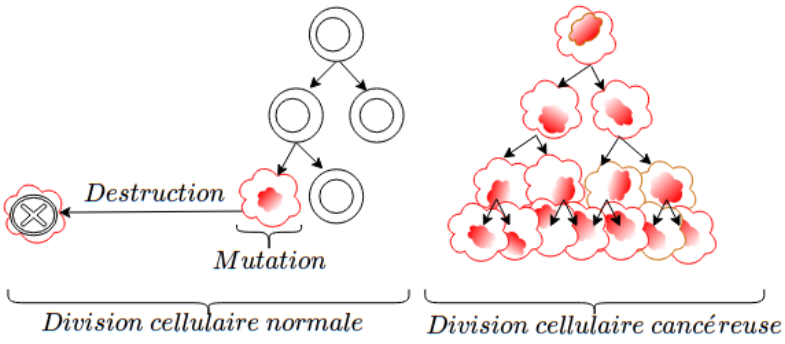


FIGURE 1.3 – Division cellulaire dans le cas d'un cancer

1.4 La tumeur bénigne et maligne

Quand une tumeur comportant des mutations sans danger pour la santé cesse de grandir, on parle de tumeur bénigne. Dans le cas contraire, c'est une tumeur maligne (un cancer).

1.5 La Métastase

L'organisme ayant conscience de la naissance de nouvelles cellules crée des vaisseaux sanguins afin de leurs apporter des nutriments. Les cellules cancéreuses empruntent ces vaisseaux pour migrer vers d'autres parties du corps : c'est la métastase. La membrane basale sert de moyen d'ancrage aux cellules, elle intervient comme filtre pour leurs nutrition, si celle-ci est perforée par les cellules cancéreuses, on parle de tumeur maligne [3] [6].

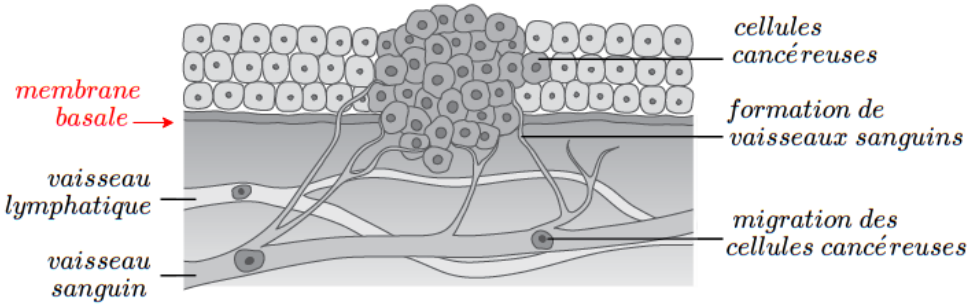


FIGURE 1.4 – Migration d'un cancer

1.6 Le cancer du sein

Le cancer du sein est le premier type de cancer connu de l'histoire. Il est également l'un des plus fréquents et invasifs chez la femme. La science le départage en deux catégories distinctes : le cancer invasif et non invasif, ils enferment tous deux d'autres sous catégories en fonction de la région du sein où il est localisé c'est pour quoi, avant de les classifier, nous définirons brièvement l'anatomie d'un sein.

Remarque : moins de 1% de cancers de sein sont décelés chez l'homme [4].

1.6.1 Anatomie du sein

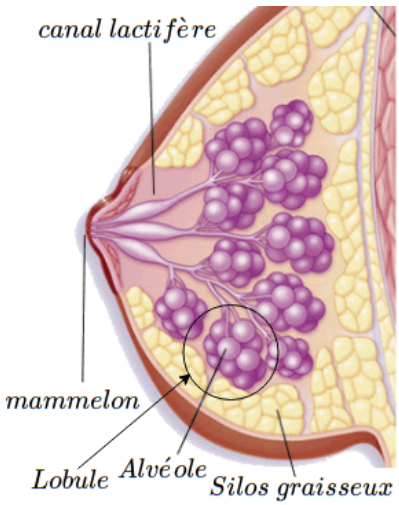
La fonction biologique du sein est de produire du lait afin de nourrir un nouveau-né.

La structure du sein est illustrée dans la figure ci-contre [5].

— Silo graisseux : principalement de la graisse.

— Lobules : les lobules sont un groupe d'alvéoles responsables de la production de lait.

— Canal lactifère : responsable du transport de lait jusqu'au mamelon.



1.6.2 Les cansers du sein non invasifs (in situ)

Le cancer du sein "in situ" s'inscrit dans la catégorie où la division cellulaire mutée ne dépasse pas sa membrane basale (pas de métastase).
✓Cette classe de cancer du sein est peuplée par le carcinome canalaire.

carcinome canalaire

Le carcinome canalaire est la forme de cancer du sein la plus fréquente. Il se forme dans les canaux de lactation. Il est dans la plupart du temps non invasif ce pour quoi il est classifié dans la catégorie « In situ ».

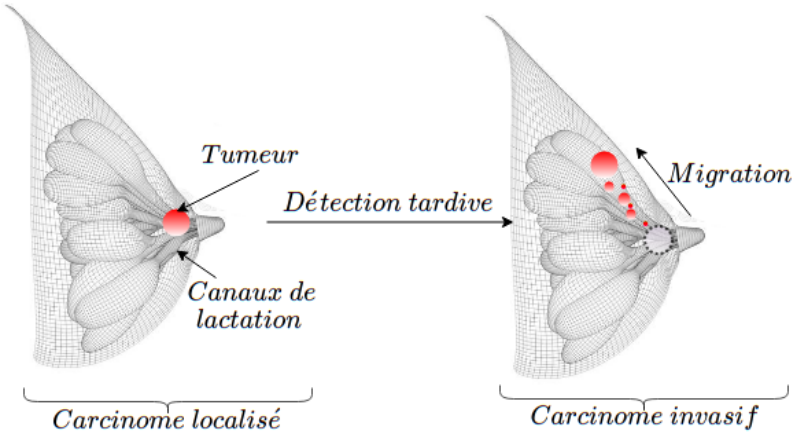


FIGURE 1.5 – Schéma simple d'un carcinome canalaire

1.6.3 Les cancers du sein invasifs (infiltrants)

On sous-entend par "invasif", le cancer qui se délocalise et migre vers les tissus voisins pour infiltrer le reste du corps. Il est à ce jour le type de cancer le plus redouté. Cette catégorie est peuplée par trois pathologies distinctes du sein : carcinome lobulare, carcinome inflammatoire, maladie de Paget.

Le carcinome lobulaire

Le carcinome lobulaire se développe dans les lobules, il traverse les parois pour migrer en direction des tissus voisins[8].

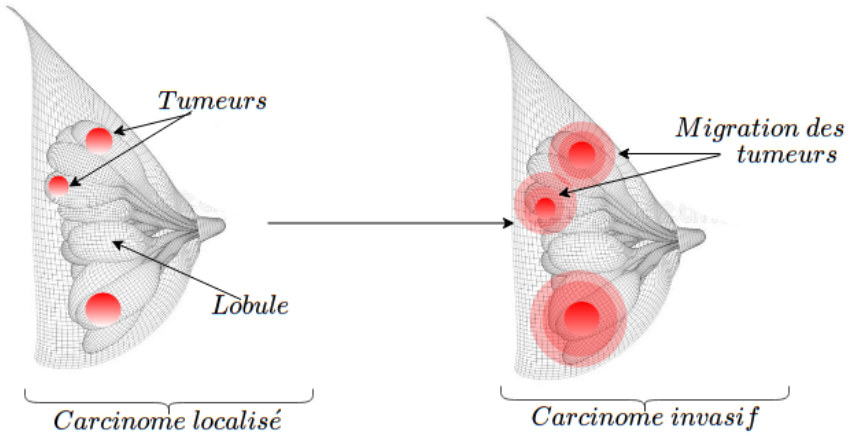


FIGURE 1.6 – Schéma simple d'un carcinome lobulaire

Le carcinome inflammatoire

Le carcinome inflammatoire est une forme rare mais ses symptômes sont les plus visibles.
Les tissus cancéreux se forment initialement dans les canaux mammaires puis migrent dans le reste du corps via les canaux lymphatiques de la peau. Scénario illustré dans la figure-1.7 [9].

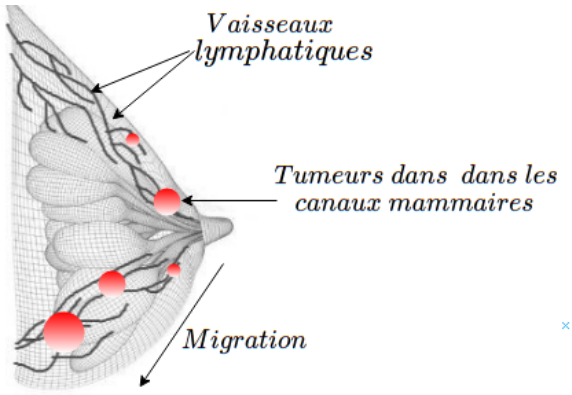


FIGURE 1.7 – Schéma simple d'un carcinome inflammatoire

La maladie de paget

La maladie de Paget est aussi un type rare de cancer du sein. Il apparaît sous la forme d'une éruption cutanée (plaie qui ne guérit pas) ou d'autres changements sur la peau du mamelon[10].

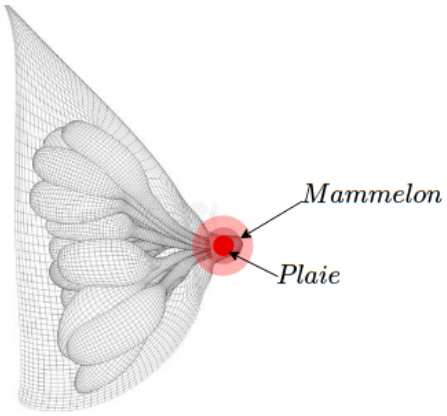


FIGURE 1.8 – Schéma simple de la maladie de paget

1.7 Classification générale de cancers

Afin d'éliminer toute ambiguïté relative aux termes utilisés pour désigner un cancer, les résultats médicaux d'une quelconque pathologie suspectée cancéreuse sont classifiés comme suit :

— **normal** : pas de cancer ni de tumeur, l'altération détectée est une autre pathologie.

— **bénin** : tumeur sans danger pour l'organisme et sans migration.

— **in situ** : tumeur sans migration comportant un danger pour l'organisme (risque de métastase future).

— **invasif** : tumeur avec migration présentant un danger pour l'organisme (métastase).

1.8 Le diagnostic

Le diagnostic est une tâche particulièrement délicate pour le corps médical, nombreux sont les cancers qui demeurent à ce jour difficile à détecter toutefois, la pluralité des diagnostics suivent inévitablement les étapes suivantes [11].

1.8.1 Bilan sanguin

Un bilan sanguin est demandé par le médecin si celui-ci détecte une grosseur (nodules) anormale au niveau du sein ou que le/la patiente présente des symptômes inquiétants.

1.8.2 Imagerie médicale

Si une anomalie est présente sur un bilan sanguin, une imagerie médicale est sollicitée selon l'altération détectée. Parmi les techniques d'imagerie on cite la radiographie, échographie, IRM et Pet-scan.

Radiographie

La radiographie est analysée à l'œil nu par le médecin, celui-ci ne peut pas détecter des tumeurs de taille très petite de plus, la présence de gaz dans le corps introduit du bruit dans l'image résultante.

Échographie

Cette méthode se base sur l'utilisation d'ultrasons. Un traitement informatique traduit le résutat en image permettant ainsi d'obtenir une représentation indirecte de la région.
Inconvénients : Certains types de tumeurs ne réfléchissent pas ces ultrasons, l'échographie ne permet donc pas de les détecter.
Une échographie est très indirecte et floue, il est souvent ardu pour un médecin de distinguer une anomalie.

L'IRM

L'IRM est une technique d'exploration qui se base sur la détection des réactions de différentes parties du corps exposées à un champ magnétique.

Inconvéninants : Tous les hôpitaux et centres d'imagerie ne possèdent pas l'équipement spécialisé en IRM du sein.

Le pet-scan

Le PET-scan permet de visualiser le fonctionnement des organes. Il consiste à injecter du glucose (un sucre faiblement radioactif) puis à analyser l'image obtenue par un scanner. Les cellules cancéreuses sont identifiables car elles ont une activité plus importante et consomment plus de glucose.

Inconvénients : certaines cellules tumorales ne consomment pas ce glucose, elles ne sont donc pas détectables par pet-scan.

1.8.3 La biopsie

Après avoir constaté une tumeur ou quelconque anomalie conséquente en imagerie, un prélèvement du tissu suspecté est effectué pour le faire analyser par un anatomopathologiste [11].

1.9 Anatomopathologie

L'anatomopathologie est une spécialité méconnue du public, son importance est pourtant cruciale, c'est une étude qui s'intéresse à la lésion microscopique et macroscopique des tumeurs en particulier et des organes en général. Dans le cas des tumeurs, l'anatomopathologiste est responsable de les étudier et d'en déduire un pronostic afin de définir une stratégie thérapeutique pour le patient[13].

1.9.1 Etude microscopique

L'étude microscopique est l'activité principale d'un anatomopathologiste. Elle est effectuée par méthode optique (directement dans le microscope électronique), ou par microscope photonique (histologie). Cette étape sert à déterminer la nature tumorale d'une lésion pour aboutir éventuellement à la dénomination d'une tumeur.

1.9.2 Histologie

L'histologie est une branche de la médecine et de la biologie qui étudie les tissus biologiques. Elle participe à l'exploration de pathologies comme les cancers et leurs effets.

1.9.3 Coloration

Les tissus biologiques comportent naturellement très peu de contraste, différentes méthodes de coloration sont utilisées afin de mettre en évidence les composants microscopiques pour faire de la classification histologique [15].

Remarque : la coloration des tissus est à l'origine du rose violacé présent sur les histologies utilisées en IA.

➥Le shcéma suivant résume la globalité d'un diagnostic

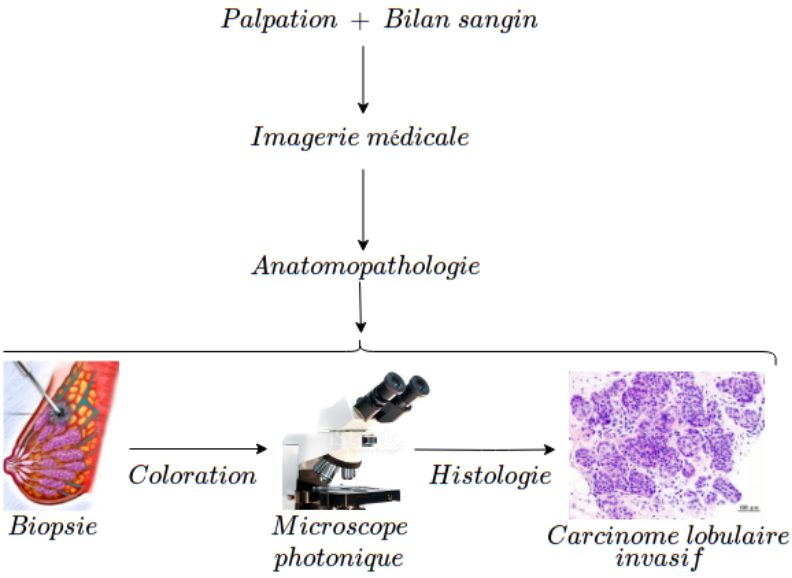


FIGURE 1.9 – Étapes générales de diagnostic

1.10 Obstacles de détection par imagerie médicale

Les obstacles ayant motivé l'introduction de l'anatomopathologie et l'histologie sont les suivants [11] :

1. L'imagerie faillit souvent à son rôle de détection.
2. Un médecin ne peut pas détecter une tumeur de moins de 1 cm de diamètre (1 milliard de cellules cancéreuses). Une histologie permet d'étudier une seule cellule.
3. L'imagerie médicale ne peut pas détecter les tumeurs de moins de $1mm$ de diamètre (10 millions de cellules cancéreuses).

1.11 Caractéristiques histologiques des tumeurs

Il existe une multitude de paramètres à prendre en compte pour dénommer la nature d'une histologie, les critères de bénignité et malignité n'ont pas de valeur générale car aucun d'entre eux n'est à ce jour formel. un diagnostic est donc très équivoque et relatif [14]. Nous citerons dans ce qui suit les signes de malignité les plus répandus.

tumeurs bénignes

1. Noyau cellulaire normal.
2. Division cellulaire faible.
3. Différencié : la cellule cancéreuse ressemble aux cellules qui lui ont donné naissance.
4. Pas de récidive : une ablation chirurgicale d'une tumeur ne conduit pas à une prochaine récidive.

tumeurs malignes

1. Noyau anormalement grand et foncé.
2. Cellules de différentes tailles avec une division cellulaire élevée.
3. Envahissement des tissus voisins (périphérie mal limitée, membrane basale perforée).
4. Irrégularité de contour de cellule.
5. Augmentation du rapport noyau/cytoplasme.

1.12 Exemples histologiques

Afin de comprendre mieux les images histologiques, nous avons choisi le carcinome canalaire pour exemple [19].
L'hyperplasie canalaire simple se caractérise par une prolifération intracanalaire de cellules (la membrane basale reste intacte).
L'hyperplasie canalaire potentiellement cancéreuse (atypique) est une hausse du nombre de cellules anormales qui se développent dans les canaux mammaires.

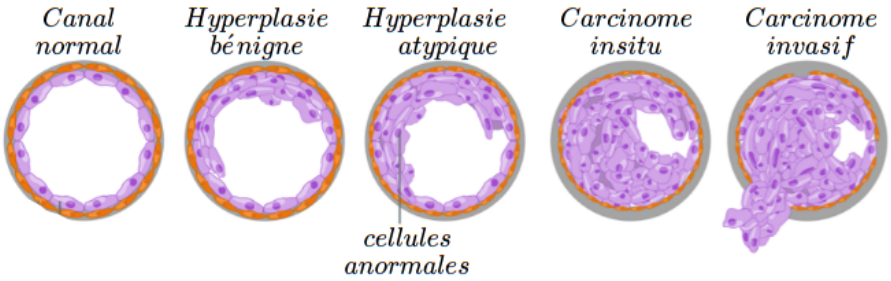


FIGURE 1.10 – Développement d'un cancer dans un canal lactifère

✔ La figure suivante illustre une hyperplasie bénigne, Le risque de progression vers un carcinome ou une récidive locale après ablation chirurgicale est nul ou très faible (de 0 à 4%) [16].

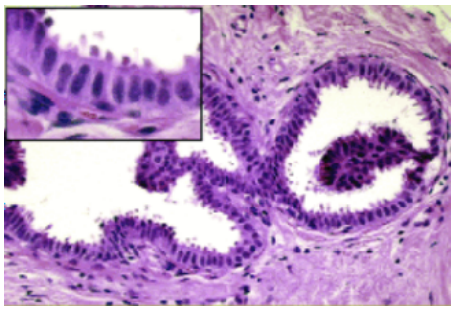


FIGURE 1.11 – Hyperplasie bénigne (histologie)

✔La figure suivante représente une prolifération totale de cellules anormales dans un canal, c'est une tumeur de consistance dure [17].

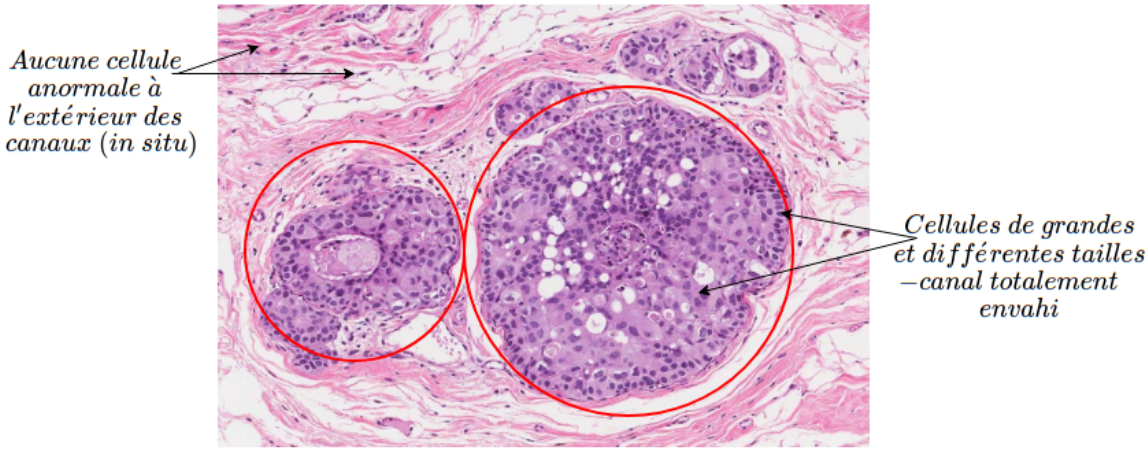


FIGURE 1.12 – carcinome canalaire in situ

La figure suivante représente une prolifération totale à l'extérieur d'un canal, c'est un carcinome invasif, les membranes basales sont totalement perforées [18].

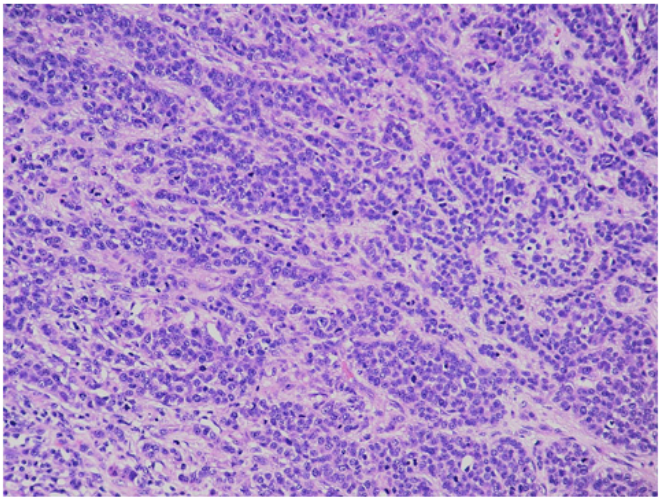


FIGURE 1.13 – carcinome canalaire invasif

1.13 Scepticisme de diagnostic autour d'un cancer

Une tumeur maligne peut se présenter comme une tumeur bénigne car les critères de malignité cités plus haut peuvent être absents ou trompeurs. Plus rarement, une tumeur bénigne peut se présenter comme étant une tumeur maligne mais souvent, une quelconque maladie ou mal formation sans importance peut se présenter comme une tumeur/cancer.

Exemples :

— Dystrophie : la dystrophie est une modification de cellules quand celles-ci n'ont pas un apport adéquat de nutriments comme le sang. Elles se modifient en conséquence et prennent un aspect tumoral (kystes mammaires ou nodules).

— Il existe des malformations naturelles de tissus qui laissent à penser que ce sont des tumeurs alors qu'elles ne le sont pas.

— Des cellules cancéreuses bénignes peuvent sécréter des substances telles que le calcium : le patient peut alors décéder d'une hypercalcémie.

Remarque : il existe néanmoins des tumeurs kystiques ou des dystrophies précancéreuses, ce pour quoi, la frontière entre malignité et bénignité est extrêmement étroite [20].

1.14 Objectifs et motivations de l'intelligence artificielle

1. Réduire le scepticisme afin d'orienter au mieux le spécialiste.
2. Les thérapies utilisées pour traiter le cancer (chimiothérapie, radiothérapie, ablation chirurgicale) sont très lourdes et peuvent porter atteinte au pronostic vital du patient c'est pour quoi, un diagnostic doit être extrêmement sûr et précis.
3. Le principal outil d'un anatomopathologiste est l'œil : l'intelligence artificielle peut imiter son comportement.
4. La grande quantité de données et la complexité des histologies rendent cette tâche fastidieuse et non triviale.
5. Dans certains diagnostics délicats, un anatomopathologiste se fie uniquement à ses expériences, il fait également appel à l'expérience de ses collègues pour des échanges de dossiers : il faut donc exploiter cette notion de déjà-vu. Un ordinateur peut accumuler une mémoire à très grande échelle par rapport à un humain.
6. Disponibilité d'une « tumorothèque » : une grande base de données recensée par les laboratoires, universités et autres afin d'avoir un maximum de références.
7. Un diagnostic précoce augmente le succès du traitement (gain de temps).

Conclusion

Dans ce chapitre, nous avons ciblé les éléments essentiels qui nous ont permis de nous familiariser avec le cancer du sein afin d'identifier les méandres rencontrés par les spécialistes qu'une machine peut éventuellement réduire. Le plus important réside donc dans la difficulté d'un diagnostic et l'importance d'avoir choisi des histologies pour base d'apprentissage.

Dans le chapitre suivant, nous aborderons une première approche de ce dit apprentissage : une vue d'ensemble afin d'effectuer un premier pas vers une solution.

Chapitre 2

Méthodes classiques de l'apprentissage automatique (ML)

Introduction

L'intelligence artificielle est dépeinte par l'ensemble des moyens techniques et théoriques réuni afin qu'une machine puisse apprendre d'elle-même. Plus antérieures que populairement présumées, les premières prémisses dudit domaine furent en 1950 suites à la publication d'Alan Turing " *Computing machinery and Intelligence*". À cette époque, les études amoncelées par le mathématicien demeurent hypothétiques faute de moyens physiques et économiques [21].
A partir de 2007, le domaine s'est vu connaître une expansion considérable grâce à la réalisabilité de la loi de Moore, à l'explosion de données au moyen d'internet et à l'investissement économique en informatique. L'intelligence artificielle prend alors de plus en plus de d'ampleur pour se faire une place en robotique, traitement d'images, jeux et autres.

Dans ce chapitre, nous aborderons les différents paradigmes de l'intelligence artificielle, ses méthodes classiques et prémodernes tels que le perceptron.

2.1 Taxonomie de machine learning

Le Machine Learning est le terme le plus utilisé pour désigner l'intelligence artificielle pourtant, ces deux notions ne sont pas équivalentes mais imbriquées.
La figure suivante illustre les différents sous-ensembles qui constituent l'IA [22].

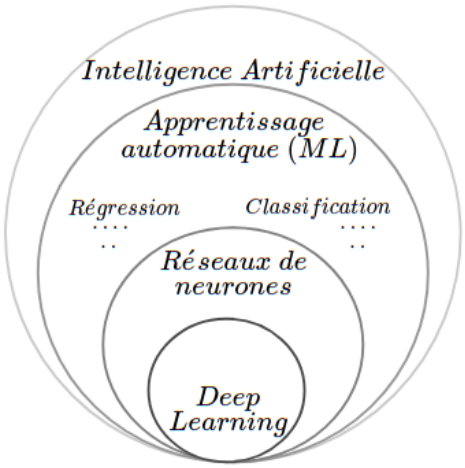


FIGURE 2.1 – Les sous ensembles de l'IA

La plupart des problèmes catégorisés ML sont résolus suivant trois étapes :

1. Définition des entrées.
2. Définition d'un modèle.
3. Agencement des sorties par le modèle.

Conformément à ce procédé, le ML est subdivisé selon la nature des données en entrée puis selon la nature des données en sortie.

2.1.1 Machine learning selon les entrées

Apprentissage supervisé

Dans cette catégorie, les données en entrée sont étiquetées (c'est-à-dire que la réponse à la tâche est connue pour ces données), il s'agit d'un apprentissage supervisé. Le but est de comparer la prédiction à la valeur effective afin de réduire au maximum l'erreur de prédiction.

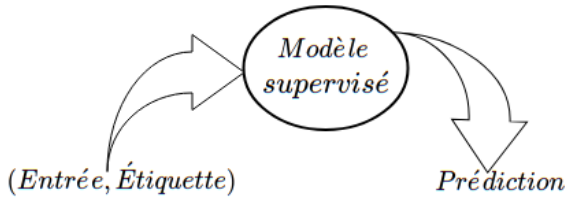


FIGURE 2.2 – Apprentissage supervisé

Apprentissage non supervisé

Dans l'apprentissage non supervisé, les données d'entrée ne sont pas étiquetées. Le programme raisonne sur les caractéristiques relatives à la structure des données pour créer des groupes, on parle de "Clustering" [23].

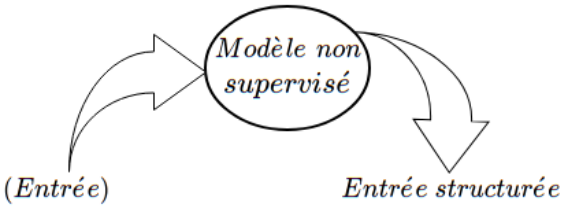


FIGURE 2.3 – Apprentissage non supervisé

Apprentissage par renforcement

Ce paradigme de l'apprentissage se distingue par l'absence de données initiales par conséquent, il est constitué d'un agent, un environnement puis un état et une action.
L'agent se trouvant dans un environnement avec un $état_i$ effectue une action, en fonction de celle-ci, l'environnement décide de le récompenser ou pas. Le but du modèle est de maximiser le nombre de récompenses.

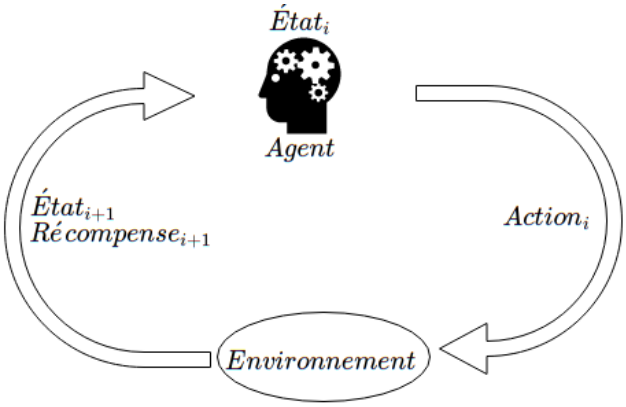


FIGURE 2.4 – Apprentissage par renforcement

2.1.2 Machine learning selon le type de résultat

La régression

La régression permet de fournir à un modèle des données pour en déduire un résultat à valeur continue. Une courbe d'ajustement est calculée afin d'approximer au mieux les données préexistantes, cette courbe permettra de prédire de nouvelles entrées au futur.

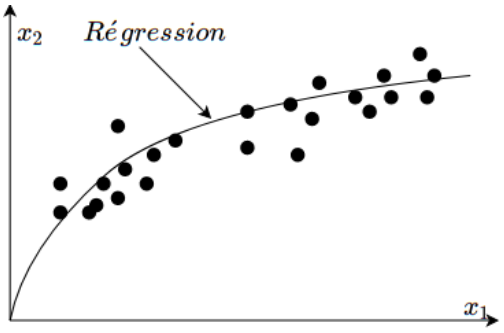


FIGURE 2.5 – La régression par courbe

La classification

Contrairement à la régression, la classification permet d'obtenir des résultats à valeurs discrètes. Celles-ci peuvent représenter des classes ou des clusters.
La figure 2.9 exemplifie la différence entre la classification et le clustering [24].

Remarque : la taxonomie du ML varie selon plusieurs critères : paramétrique, non paramétrique..etc ; nous avons effectué une répartition générale du ML conformément aux recherches de méthodes effectuées pour trouver un modèle adéquat.

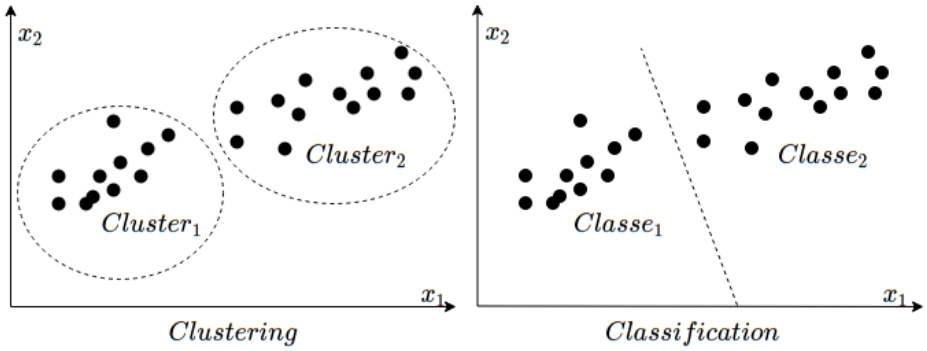


FIGURE 2.6 – La classification vs clustering

➜ La prédiction de maladies tels que le cancer du sein revient à créer un modèle disposant d'un jeu d'entraînement étiqueté par les histologies préexistantes afin de classifier les différents types de cancers détectés : c'est la classification supervisée.
La figure suivante résume la systématique générale du ML.

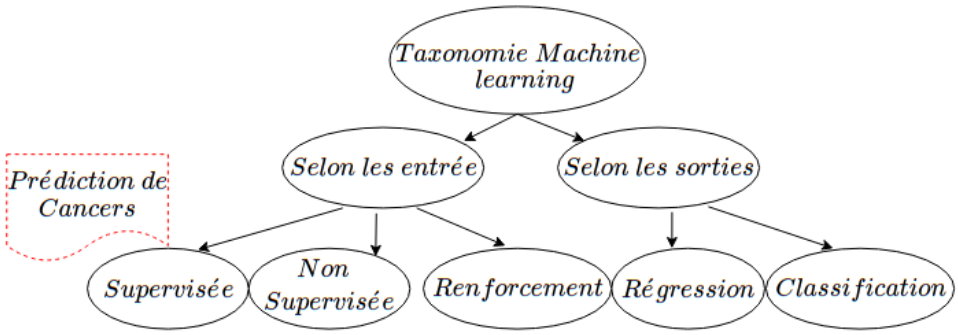


FIGURE 2.7 – Taxonomie générale du ML

2.2 Classification supervisée

La classification supervisée a pour objectif d'approcher au mieux une réalité préalablement quantifié/estimées. Les valeurs en sortie sont donc explicitement fournies par les données, elle se base essentiellement sur la modélisation bayésienne [25].

2.2.1 La méthode des K plus proches voisins

Dans La méthode des k plus proches voisins ou K-nn, la base de données d'apprentissages est constituée de couples (entrée, sortie). Pour classifier une nouvelle entrée x, le K-nn prend en compte les k exemples les plus proches de la nouvelle entrée x puis fournis en sortie la classe majoritaire des k-plus proches voisins[26]. Pour calculer les plus proches voisins, La distance la plus utilisée est celle d'Euclide formulée comme suit :

$$d\left(Y,X\right)=\sqrt{\sum_{i=1}^{n}\left(x_i-y_i\right)^2}$$

— X : vecteur de données de la nouvelle entrée.

— Y : vecteur de données issue de l'entraînement.

Exemple :X,Y peuvent représenter une image dont x_i y_i sont respectivement chaque pixel de l'image en entrée et celle issue de l'entraînement.

➜ La figure suivante illustre un exemple de 5-nn.

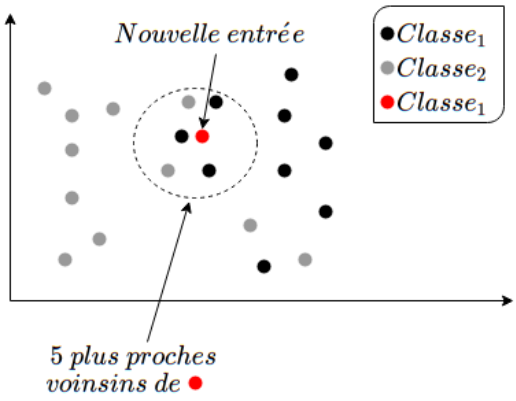


FIGURE 2.8 – Exemple d'un K-nn

Avantages et inconvénients du K-nn : L'apprentissage est rapide mais le programme est coûteux en calculs pour des jeux de données larges.

✓ **Remarque :** L'algorithme K-nn est également utilisé dans la classification non supervisée. Il est considéré comme l'une des plus anciennes méthodes de classification, ses avantages ont assuré sa pérennité tandis que ses inconvénients ont donné naissance à de nouveaux formalismes tels que le perceptron.

2.2.2 Le perceptron

Le concept du perceptron a été introduit en 1957 par Frank Rosenblatt [27], ce dernier s'est inspiré du neurone biologique pour modéliser un neurone formel. Afin de comprendre cette analogie, nous allons définir le neurone biologique et son fonctionnement primaire.

Le neurone biologique

Le rôle fondamentale d'un neurone est de recevoir, propager et transmettre un signale nerveux. Le message nerveux se propage dans un sens unique : des dendrites vers les terminaisons nerveuses en passant par l'axone.

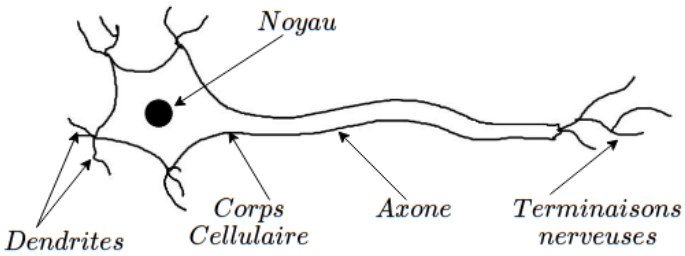


FIGURE 2.9 – Composition basique d'un neurone

Comme le montre la figure ci-jointe, un neurone intègre l'ensemble des messages reçus par ses dendrites. Le processus d'intégration est effectué au niveau du corps cellulaire. Le signe "+" correspond à un message positif (excitation) et le "−" à un message négatif (inhibition).

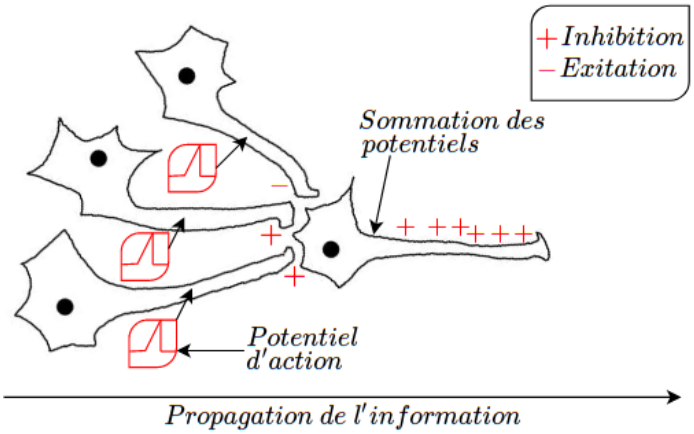
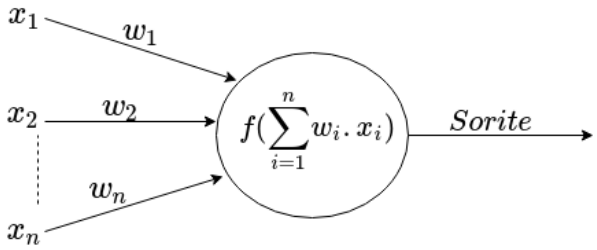


FIGURE 2.10 – La Propagation d'un message nerveux

✓C'est ce processus d'intégration/excitation/inhibition que le neurone formel reprend afin de traiter l'information.

Le neurone formel

Le neurone comme objet mathématique reçoit en entrée un vecteur de données X (les dendrites) pondéré par des poids w (potentiels d'action/inhibition).
Le centre du neurone formel représente la membrane cellulaire biologique où la somation des messages est effectuée. Une fonction mathématique de seuillage permet de modéliser cette sommation (fonction d'activation). Le calcul est effectué de la manière suivante :



➥Soit $\sum_{i=1}^{n} w_i.x_i = z$

➥ Si $z \geq 0$ alors $f(z) = 1$
➥ Si $z < 0$ alors $f(z) = 0$

FIGURE 2.11 – Le neurone formel

Les sorties 0 et 1 représentent chacune une classe de prédiction pour le vecteur d'entrée X. On dit alors que le perceptron est un classifieur binaire[28].

Mise à jour des poids (loi de Widrow-Hoff)

Pour que les classes soient prédites correctement, les poids w sont mis à jour répétitivement jusqu'à obtention de la bonne prédiction. L'ajustement des poids est effectué par la formule suivante :

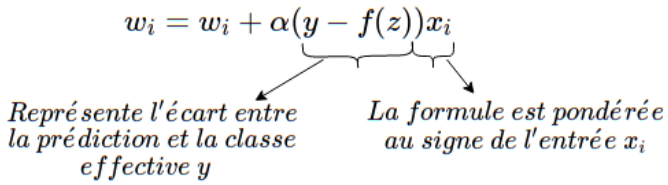


FIGURE 2.12 – Mise à jour de Widrow-Hoff

Remarque : La règle d'apprentissage de Widrow-Hoff compte parmi les premières règles d'apprentissage avec la règle de Hebb qui stipule que lorsque deux neurones sont excités conjointement, ils créent ou renforcent un lien les unissant[32].

2.2.3 Estimation de l'erreur : La méthode des moindres carrés

La méthode des moindres carrés compte parmi les premières techniques utilisées afin de calculer et minimiser le taux d'erreur. Celle-ci mesure la distance entre les prédictions et les valeurs effectives de la manière suivante :

$$Loss = \sum (y_i - f(z_i))^2$$

➥ Le but est de trouver les paramètres w_i qui permettent de minimiser la fonction $Loss$. On cherche à obtenir les valeurs w_i pour lesquels la dérivée de la fonction $Loss$ par rapport à w est nulle (minimum global).

2.2.4 La descente de gradient classique

La descente de gradient classique consiste à tracer une courbe de la perte de toutes les données d'entraînement en fonction des paramètres. La courbe est tracée à partir de la fonction suivante :

$$\sum_{i=1}^{n} loss(y_i, f(z_i))$$

Une fois cette somme calculée et la courbe tracée, la mise à jour des poids est effectuée comme suit [36] :

$$w_i = w_i - \alpha \frac{\partial \sum loss(y_i, f(z_i))}{\partial w_i}$$

Inconvénients

- À chaque étape de la mise à jour, tout l'ensemble d'entraînement doit être parcouru pour calculer la moyenne avant de faire la mise à jour d'un poids w.
- Le gradient par rapport à la somme de toute les erreurs est trop grand, la mise à jour d'un poids devient alors conséquente et ne garantit pas l'accès à un poids optimale.

2.2.5 descente de gradient stochastique

La descente de gradient stochastique est une solution aux inconvénients de la méthode classique , celle-ci consiste à rechercher un poids optimale en mettant à jour un paramètre à partir d'un seule exemple d'entraînement successivement comme suit [36].

$$w_i = w_i - \alpha \frac{\partial loss(y_i, f(z_i))}{\partial w_i}$$

Inconvénients

— La mise à jour des poids est trop sensible aux données individuellement étant donné que la mise à jour est effectuée pour un seul exemple d'entraînement.

— La convergence vers un minimum prend beaucoup plus de temps.

2.2.6 La descente de gradient mini-batch

La méthode mini-batch consiste à mettre à jour un poids à partir d'un lot de taille m d'exemples d'entraînement, chaque itération complète sur la Data-set représente une "epoch".

$$w_i = w_i - \alpha \frac{\partial \sum_i^m loss(y_i, f(z_i))}{\partial w_i}$$

L'ensemble d'entraînement est parcouru au bout de $\frac{N}{m}$ batchs = 1 epoch.
N étant Le nombre d'exemples dans notre ensemble d'entraînement[44].

La figure ci-dessous résume le parcours d'un ensemble N d'entraînement à travers les trois méthodes de descente de gradient.

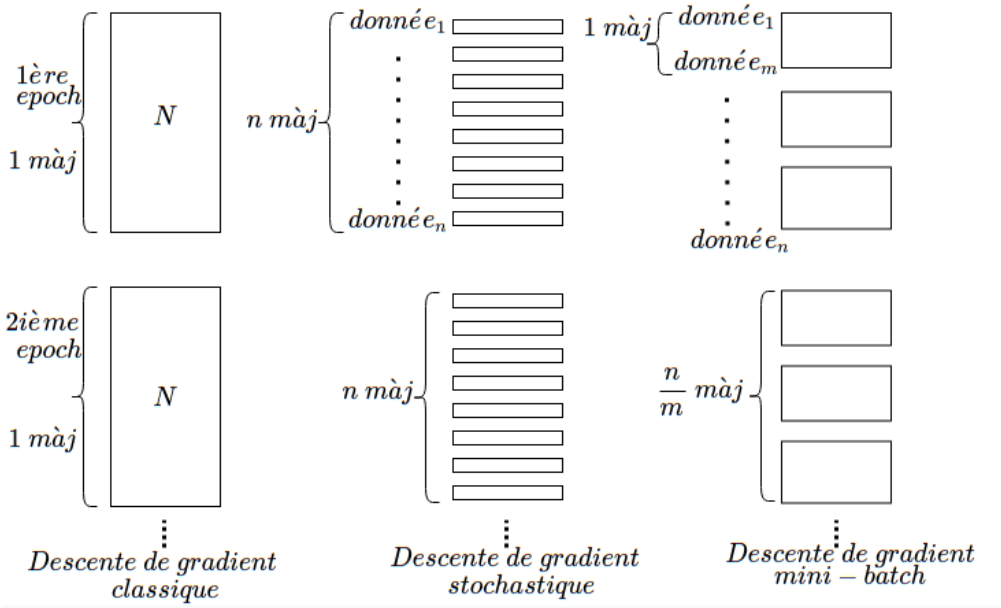


FIGURE 2.13 – Systématique de descente de gradient

2.3 Taux d'apprentissage α et optimisation

L'optimisation consiste à chercher les paramètres qui minimisent la fonction d'erreur. L'optimisation est également une recherche du taux d'apprentissage α optimale, et cela afin de converger de manière rapide et efficace vers un w optimale.
Le choix de la valeur de α peut être assez fluctuant, il est considéré comme un hyperparamètre à optimiser car si la valeur est trop petite, la réduction de l'erreur sera très lente tandis que si elle est trop grande, des oscillations divergentes peuvent en résulter. Soit la fonction de perte quadratique suivante[35].

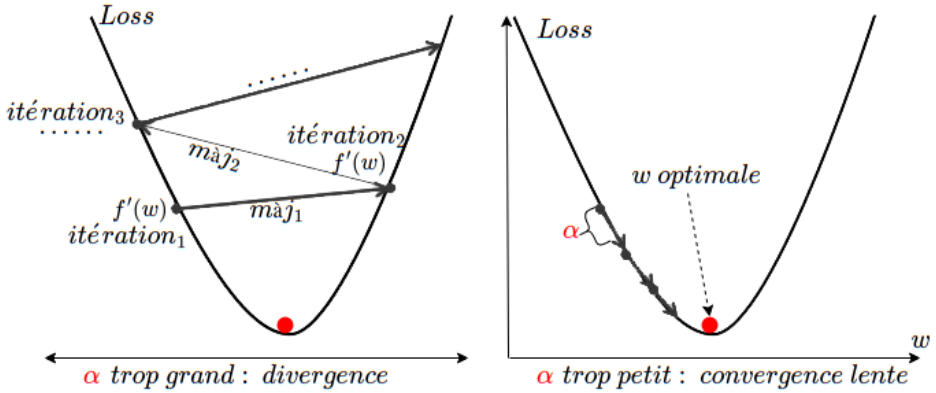


FIGURE 2.14 – Exemple de convergence pour différents α

✔ Généralement, le taux d'apprentissage est choisi dans l'intervalle suivant [29] :

$$\alpha \in [10^{-6}, 1[$$

2.3.1 Adam optimizer

L'optimisation Adam implémenté d'ores et déjà par des bibliothèques python combine deux autres algorithmes d'optimisation : SGD avec momentum et RMSprop, tous deux utilisent un lissage exponentiel.

Remarque : Adam est un algorithme d'optimisation qui est apparu en 2015 [38], il ne fait donc pas partie des méthodes classiques du ML mais des méthodes modernes, nous l'avons introduit dans ce chapitre en tant que solution pour une descente de gradient efficace.

Lissage exponentiel

Le lissage exponentiel est une méthode empirique de lissage et de prévision de données chronologiques affectées d'aléas. Il donne aux observations passées un poids décroissant exponentiellement avec leur ancienneté.
Le paramètre β est un facteur de lissage compris entre $[0, 1]$ en d'autres termes, s_t peut être vu comme une moyenne pondérée entre la valeur actuelle y_t et la valeur lissée précédente s_{t-1}. le lissage diminue quand β augmente.
Dans le cas limite où $\beta = 1$, la série lissée est identique à la série brute[30]. Le lissage est effectué comme suit.
.

$$s_0 = y_0$$
$$s_t = \beta y_t + (1 - \beta) s_{t-1}$$

Exemple : lissage exponentiel simple. Données brutes : températures moyennes quotidiennes à la station météo de Paris-Montsouris (France) du 01/01/1960 au 29/02/1960 [31]. Données lissées avec le facteur $\beta = 0, 1$

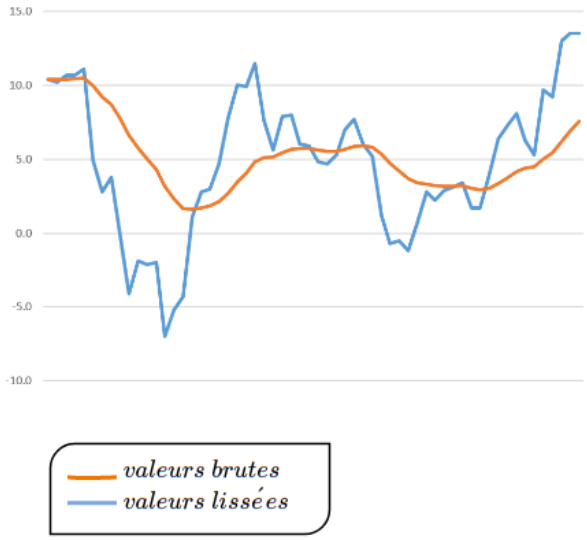


FIGURE 2.15 – Résutat d'un lissage exponentiel ($\beta = 0, 1$).

✔L'intuition ayant poussé l'introduction d'un lissage exponentiel dans l'optimisation de descente de gradient est de minimiser les oscillations (aléas) pour une convergence rapide.

SGD avec momentum

La SGD momentum est effectué par la formule suivante [38].

$$v_0 = 0$$
$$v_t = \beta_1 \times v_{t-1} - (1 - \beta_1) \times \frac{\partial Loss}{\partial w}$$
$$w_{t+1} = w_t - \alpha v_t$$

exemple : la figure suivante illustre de manière fictive une descente de gradient lissée avec momentum. Les oscillations en bleu et orange représentent les différents pas de gradients après chaque mise à jour de poids.

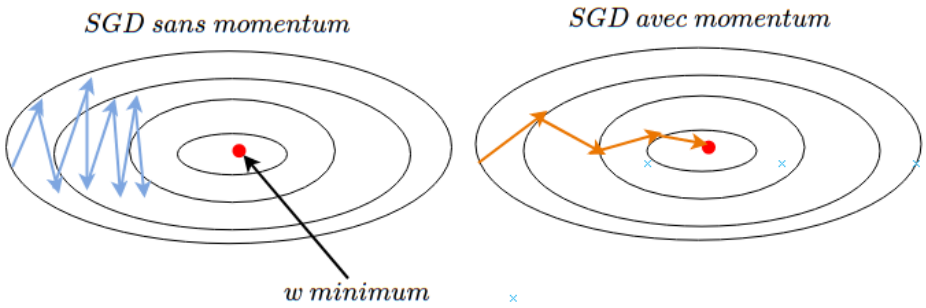


FIGURE 2.16 – SGD momentum dans un espace de gradient convexe

Remarque : la figure ci-haut est une représentation en 2D d'une fonction de perte convexe (sous forme de bole) de dimension supérieur.

RmsProp

Rmsprop utilise également le lissage exponentiel de plus, il offre une mise à jour automatique du taux d'aprentissage. Au fur et à mesure des itérations, α est divisé afin de converger de manière sûre vers un minimum. La formule est la suivante [38].

$$s_0 = 0$$

$$s_t = \beta_2 \times s_{t-1} - (1 - \beta_2) \times \frac{\partial Loss}{\partial w}^2$$

$$w_{t+1} = w_t - \frac{\alpha}{\sqrt{s_t + \epsilon}} . \frac{\partial Loss}{\partial w}$$

Remarque : (ϵ) est utilisé afin de ne pas obtenir un cas indétermimé de division $\frac{\alpha}{0}$ lorsque $s_0 = 0$.

exemple : la figure ci-jointe illustre de manière fictive une descente de gradient lissée par Rmsprop, le pas de gradient est diminué au fur et à mesure des itérations.

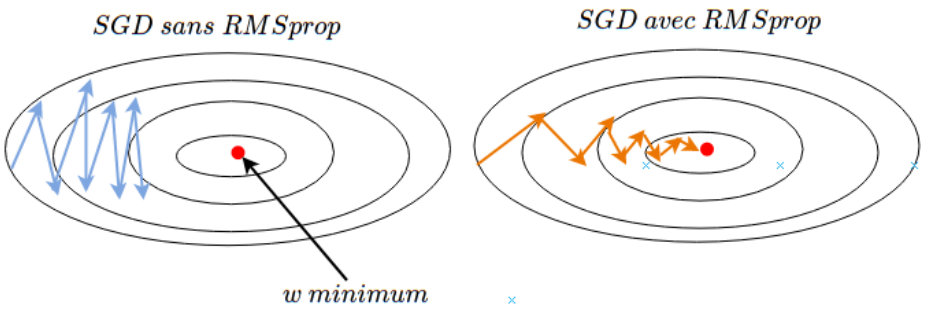


FIGURE 2.17 – SGD RmsProp dans un espace de gradient convexe

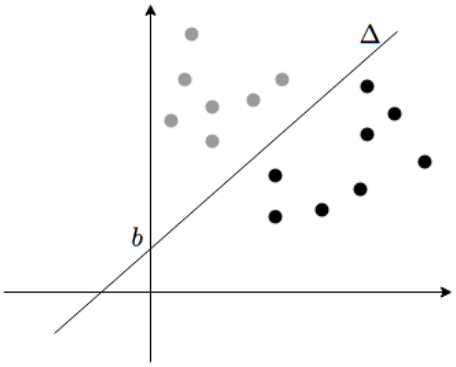
De manière générale, on peut dire que Momentum réduit les fluctuations pour converger de manière rapide et RmsProp réduit la taille du pas de gradient afin de converger de manière sûre.

Adam : L'omptimisation Adam tire profit des avantages de RmsProp et Momentum, elle est effectuée comme suit.

$$w = w - \alpha \frac{v_t}{\sqrt{s_t + \epsilon}} \times \frac{\partial Loss}{\partial w}$$

Le biais b

La majorité des perceptrons (et réseaux de neurones) comprennent un biais b ajouté à la sommation. Dans un plan à 2 dimensions, le biais a pour but de passer d'une fonction linéaire à une fonction affine. Cette dernière permet d'ajuster la droite Δ par rapport à l'axe y comme le montre la figure suivant [39].



➥ $\Delta = \sum_{i=1}^{n} w_i.x_i + b = ax + b$
➥ Δ est une fonction affine.

FIGURE 2.18 – Exemple d'un biais dans une classification binaire

Remarque : le biais b est mis à jour de la même façon que les poids w.
Concrètement, un biais dans un réseau de neurones permet de valider ou invalider un groupe de neurones avant la mise en traitement par la fonction d'activation.

Inconvénients du Perceptron

Le principale obstacle d'un perceptron est qu'il en résulte une seule droite afin de séparer un problème linaire. Il est donc pas possible de classifier un problème qui n'est pas linéairement séparable comme suit

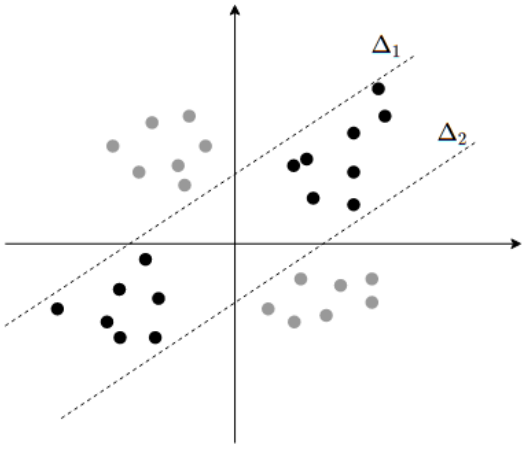


FIGURE 2.19 – Classification à deux neurones

On remarque qu'il faut deux droites afin de séparer les points gris et noirs, cela représente la combinaison de deux perceptrons (un réseau de neurones).

Conclusion

Dans ce chapitre, nous avons défini le Machine Learning en générale et la classification supervisée en particulier.
Nous avons également défini un des premiers algorithmes de classification supervisée (le K-nn), ses avantages et inconvénients ayant donné naissance au neurone formel.
Le plus important à retenir dans un neurone formel est la mise à jour des poids, la fonction de perte, descente de gradient mini-batch et son optimisation car toutes ces notions participent à la synthèse de notre solution finale.
Les inconvénients de l'algorithme du perceptron ont permis la création d'une extension de ce dernier : les réseaux de neurones.

Dans le chapitre suivant, nous étudirons les réseaux de neurones et leurs différentes fonctionnalités afin de classifer une image.

Chapitre 3

Les réseaux de neurones

Introduction

Le perceptron monocouche est un classifieur linéaire or, la pluralité des problèmes de classification ne sont pas linéairement séparable. Un algorithme k-nn dispose d'une surface de décision non linéaire [39], il est donc une alternative potentielle au perceptron pourtant, un k-nn est très coûteux en calculs et mémoire. Combiner plusieurs perceptrons afin de créer une non-linéarité est la solution la plus optimale : ce sont les réseaux de neurones.

Dans ce chapitre, nous présenterons la structure d'un réseau de neurones, son fonctionnement et les obstacles que rencontre son apprentissage.

3.1 Structure d'un réseaux de neurones

Un réseau de neurones est généralement composé d'une succession de couches.

Chaque couche L_i dispose d'une entrée correspondant à la sortie d'une couche antérieure L_{i-1}.
Chaque couche L_i est composée de n neurones. À chaque synapse (entrée x_i) est associé un poids synaptique w_{ij}.
Au niveau de chaque neurone, une sommation des x_i et w_{ij} est effectuée puis passée à une fonction fonction d'activation f, le résultat est affecté à la couche suivante (propagation en avant).
Les couches qui se situent entre la couche d'entrée et celle de sortie sont les couches cachées. La figure suivante illustre sommairement un réseau de neurones pour classifier une image, chaque pixel de l'image est une entrée pondérée.

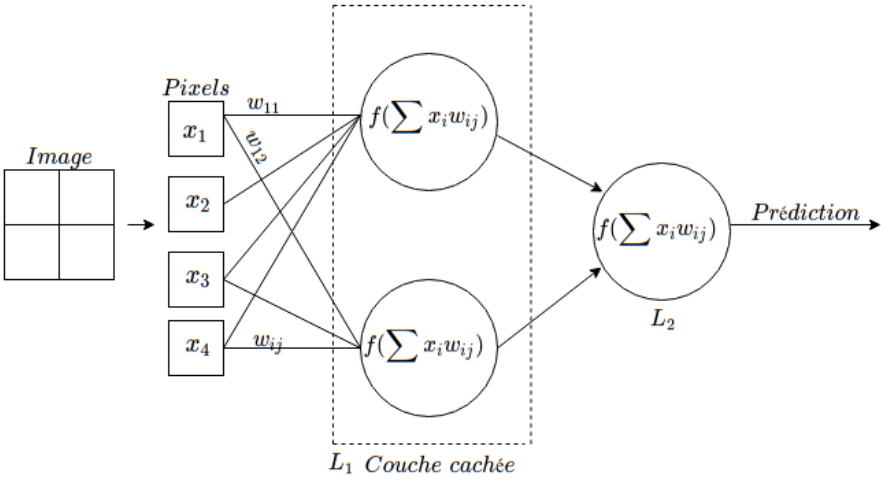


FIGURE 3.1 – Fonctionement générale d'un réseau de neurones

3.2 Les fonctions d'activation

La fonction d'activation traite l'information au niveau d'un neurone. Elle prend des formes mathématiques différentes selon les besoins d'un modèle. Son rôle est particulièrement prépondérant dans le calcul de gradient et la performance du réseau. parmi ces fonction on cite : La marche de Heavisite, Sigmoïde, softmax et Relu.

3.2.1 La marche de Heaviside

La marche de Heaviside doit son nom à sa forme en escalier discontinue en zéro. Elle permet de transformer les entrées négatives en 0 et les positives en 1 de la manière suivante :

$$f(x) = \begin{cases} \text{classe } 0 & \text{si } x < 0 \\ \text{classe } 1 & \text{si } x \geq 0 \\ f'(0) = +\infty & \end{cases}$$

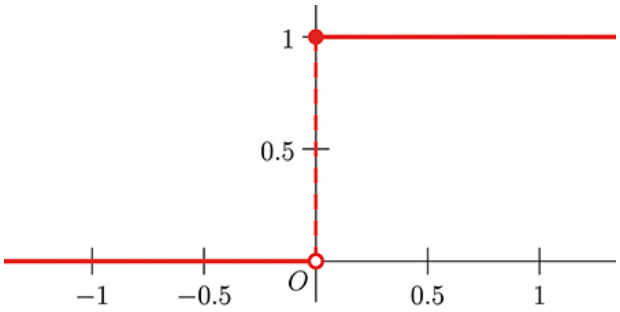


FIGURE 3.2 – Graphe et formule de la fonction Heaviside

Remarque :
Malgré la simplicité de la fonction Heaviside, elle n'est plus couramment utilisée de nos jours dus à la non-dérivabilité de celle-ci en zéro de plus, elle ne produit que deux valeurs de classification (0 ou 1), elle ne convient donc pas à problèmes de classification multiples [39].

3.2.2 La fonction Logistique Sigmoïde et softmax

fonction Sigmoïde

La fonction logistique est l'une des premières fonctions d'activation utilisée pour pallier aux problèmes de la fonction Heaviside. Celle-ci transforme ses entrées en une probabilité d'appartenance à une classe et non la classe elle-même [39]. Les sorties sont donc comprises entre $[0, 1]$.
$p(y = 1/x)$ veut dire : probabilité d'avoir y=1 sachant une entrée x (modélisation bayésienne).

$$p(y = 1/x) = \sigma(x) = \frac{1}{1 + e^{-a}}$$

$$\text{Avec } a = \sum w_{ij} x_i$$

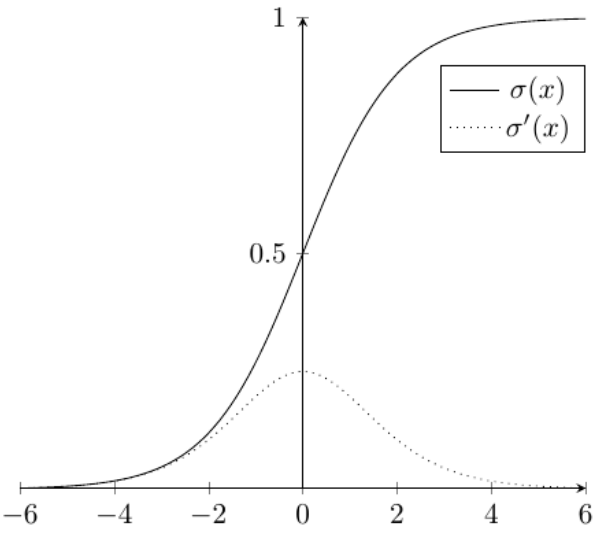


FIGURE 3.3 – graphe et formule de la fonction Sigmoïde

Inconvéniants :

— Comme le montre la figure ci-haut, la fonction sature à ses extrémités ce qui annule la dérivée. Ce problème peut engendrer un des plus grands obstacles des réseaux de neurones actuels : l'evanescence de gradients.

— La fonction Sigmoïde est utilisée que pour les classifications binaires car la somme des probabilités que fourni celle-ci est égale à 1 qu'en cas de prédiction à deux classes.

La fonction Softmax

La fonction Softmax est multiclasse, elle permet de calculer une probabilité z_i proportionnellement à une série de valeurs z_i.

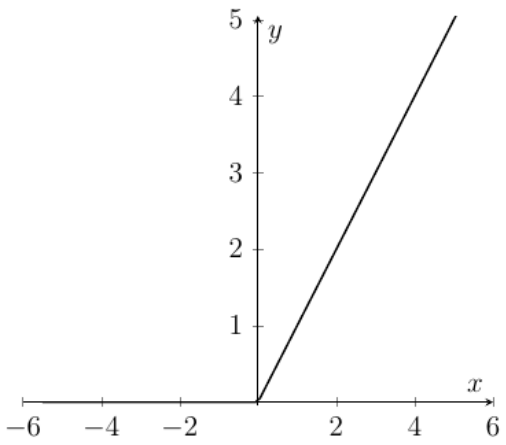
$$p(y = 1/x) = softmax(x) = \frac{e^{z_i}}{1 + e^{\sum z_k}}$$

$$\text{Avec } z = \sum x_i w_{ij}$$

Remarque : la fonction Softmax est une fonction Sigmoïde à deux classes.

3.2.3 La Fonction Relu

Relu pour "Rectified linear unit" est une fonction linéaire par morceau ce qui lui offre une non linéarité globale comme suit :



$$f(x) = \begin{cases} 0 & \text{si } x < 0 \\ x & \text{si } x \geq 0 \end{cases}$$

FIGURE 3.4 – Graphe et formule de la fonction Relu

Avantages :

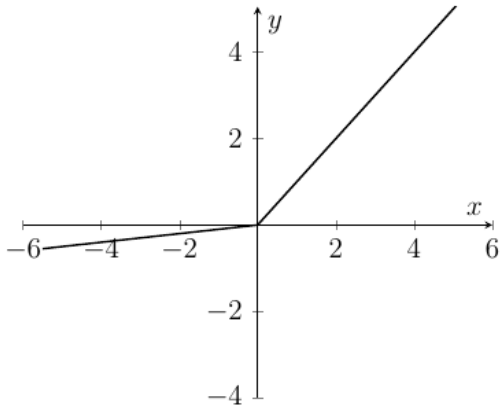
— la dérivée ne sature pas au-delà de 0 (elle ne s'annule pas).

inconvénients :

— La dérivée de la fonction à partir des valeurs négatives est de 0. Le problème est donc résolu qu'à moitié.
✔ Pour combler la dérivée aux valeurs négatives, la fonction 'Leaky Relu' est proposée.

Leaky relu

Leaky Relu est une forme générale de Relu, α est considéré comme un hyper paramètre du réseau :



$$f'(x) = \begin{cases} \alpha x & \text{si } x < 0 \\ x & \text{si } x \geq 0 \end{cases}$$

Remarque : Dans la figure ci-haut, $\alpha = 0,1$.
Relu et Leaky Relu représentent les fonctions d'activation les plus utilisées pour les couches cachées[40].

3.3 L'estimation de l'erreur : La Cross-Entropy

Dans la plupart des réseaux de neurones actuels essentiellement les réseaux de neurones convolutifs, l'utilisation de la fonction de Softmax requière une autre estimation de l'erreur. La mesure de distance entre probabilités étant impossible, la Cross-Entropy (entropie croisée) mesure une dissimilarité ou une "divergence" entre deux probabilités : c'est une variante de la divergence de Kullback Leibler [42].

$$H(p,q) = -\sum p(x) log[q(x)]$$

➥p : valeur effective y
➥q : valeur calculée $f(z)$

Intuission :
Plus $H(y, f(z)) \simeq 0$ plus y et $f(z)$ ont la même valeur similairement,
plus $H(y, f(z))$ est grand plus les valeurs y, $f(z)$ divergent.

Remarque : le vecteur d'étiquettes (valeurs effectives) se présente sous forme binaire (0 ou 1). Cette écriture est le "One hot encoding" : le but est de faciliter les calculs et de mettre l'étiquette ciblée à 1 et les autres à 0 comme illustré dans l'exemple suivant.

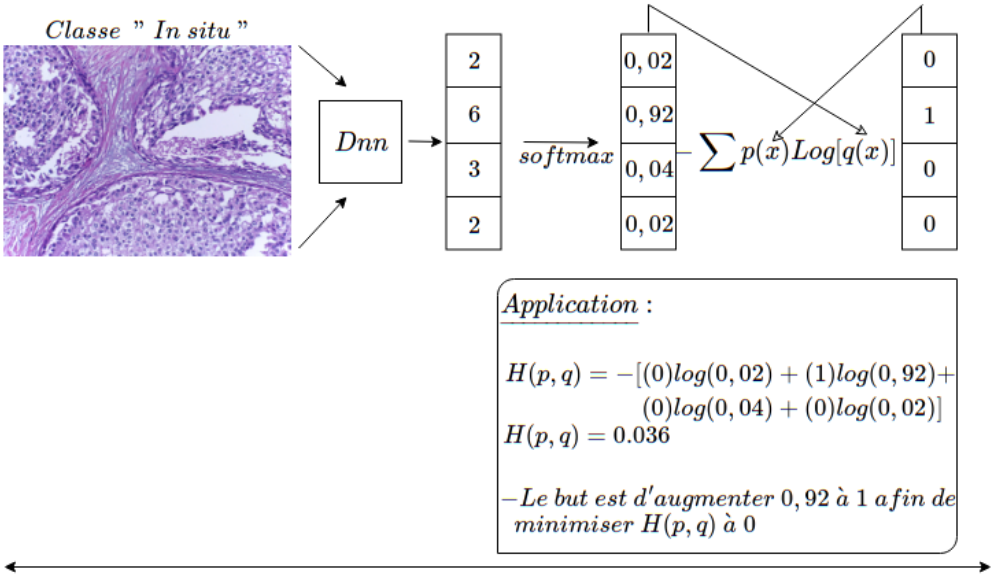


FIGURE 3.5 – Exemple applicatif de la Cross-Entropy sur une histologie

Le one-hot encoding

Un encodage one-hot ou encodage "1 parmi n" consiste à représenter des étiquettes en utilisant pour chacune des valeurs dont la représentation binaire n'a qu'un seul chiffre.
Exemple : Pour un encodage "1 parmi 4" pour quatre états possibles, on prendra les valeurs binaires 0001, 0010, 0100, 1000 pour représenter respectivement 1,2,3,4.

Avantages et inconvénients
L'avantage principal de cet encodage en ML est qu'il facilite les calculs notamment en Cross-Entropy. Son inconvénient est qu'il faut au minimum n bits pour représenter n états, ce qui conduit à une augmentation linéaire du nombre de chiffres par rapport au nombre d'états [43].

3.3.1 La logvraisemblance négative

La vraisemblance

Théoriquement, un réseau de neurones utilisant des probabilités est ajusté grâce à la vraisemblance des observations y par rapport aux prédictions $f(z_i)$. Elle est donnée par la formule suivante :

$$\prod_{i=1}^{n} f_i(z)^{y_i}$$

➥ soit le vecteur à gauche correspondant aux prédictions d'un exemple et le vecteur à droite correspondant aux observations. Le calcul de la vraisemblance entre ces deux derniers est effectué comme suit :

$$\underbrace{\begin{bmatrix} normal & 0,03 \\ bénin & 0,2 \\ in\ situ & 0,7 \\ invasif & 0,02 \end{bmatrix}}_{f(z)} \underbrace{\begin{bmatrix} normal & 0 \\ bénin & 0 \\ in\ situ & 1 \\ invasif & 0 \end{bmatrix}}_{y}$$

$$\prod_{i=1}^{n} f(z_i)^{y_i} = 0.03^0 \times 0.2^0 \times 0.7^1 \times 0.02^0 = 0.7$$

L'intuission est la suivante :

Si l'observation y_i est vrai alors :
$f(z)^y = f(z)^1 = f(z)$
➥ Maximiser $f(z)$ qui est ici de (0.7), le but est de pousser sa probabilité à 1.

Si l'observation y_i est fausses alors :
$f(z)^y = f(z)^0 = 1$
➥ Diminuer les valeurs (0.03), (0.2), (0.02) dans le but de réduire leurs probabilités à 0.

La log-vraisemblance négative et la Cross-entropy

Le produit des probabilités en vraisemblance engendre des résultats trop petits pour des calculs coûteux. Une des principales caractéristiques des logarithmes et qu'ils transforment le produit en somme facilitant grandement les calculs. La log-vraisemblance (log-likehood) négative est alors utilisée à la place de la vraisemblance de la manière suivante :

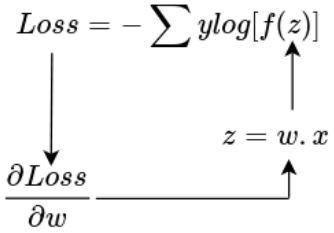
$$log(\prod_{i=1}^{n} f_i(z)^{y_i}) = \sum y_i log[f(z)] = -H[y_i, (z_i)]$$

Résultat : La log-vraisemblance négative est l'équivalent de l'entropie croisée ainsi, maximiser la log- vraisemblance négative revient à minimiser l'entropie croisée.

Remarque : l'entropie croisée est actuellement l'une des mesures d'erreur les plus utilisée dans les réseaux de neurones en général et les réseaux de neurones convolutifs en particulier[41].

3.4 La rétropropagation (backward propagation)

La rétropropagation du gradient de l'erreur (back propagation) est un algorithme d'optimisation permettant d'ajuster les paramètres d'un réseau multicouches pour mettre en correspondance des entrées et des sorties étiquetées dans un ensemble d'entraînement. Nous savons que la fonction d'erreur dans la dernière couche Softmax est l'entropie croisée. Comme illustré ci-après, calculer la dérivée de cette dernière par rapport à w est coûteuse car la fonction de perte *Loss* ne dépend pas directement de w.



3.4.1 Dérivation en chaîne

Dans les réseaux à plusieurs couches, le nombre de paramètres est de l'ordre de milliers voir de millions, le calcul de la perte $\partial Loss$ par rapport à chaque w_{ij} est donc extrêmement dispendieux.
Cette opération est amplement simplifiée en exploitant les propriétés de la règle de dérivation en chaîne [39].
Le but est d'extraire graduellement la dérivée de l'erreur par rapport aux poids en trois étapes comme suit.

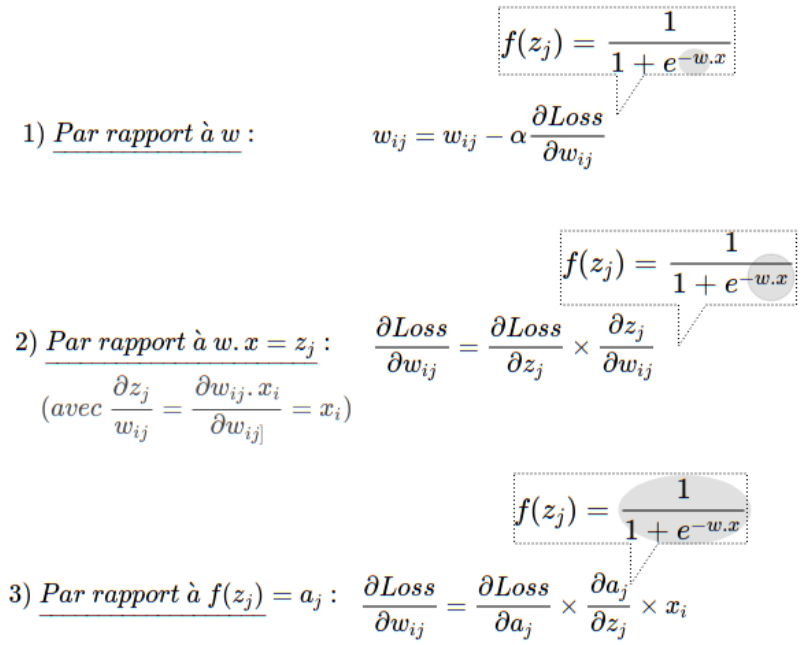


FIGURE 3.6 – Dérivation en chaîne

Rétropropagation

En reprenant l'expression obtenue en dérivation en chaîne, la rétro propagation s'effectue de couche en couche en partant de la dernière comme suit.

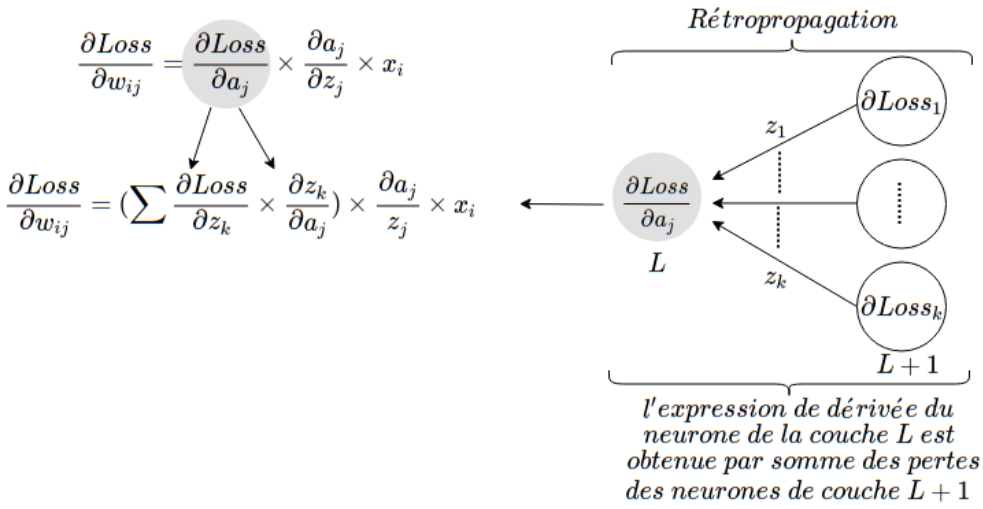


FIGURE 3.7 – Rétropropagation de la mesure d'erreur *Loss*

Calculs finaux : L'expression issue de la propagation de l'erreur permet d'aboutir aux calculs suivants :

$$1)\ \frac{\partial Loss}{\partial w_{ij}} = (\sum \frac{\partial Loss}{\partial z_k} \times \frac{\partial z_k}{\partial a_j}) \times \frac{\partial a_j}{z_j} \times x_i$$

$$2)\ \frac{\partial z_k}{\partial a_j} = \frac{\partial w_{ik} a_i}{\partial a_i}, (i = j) => \frac{\partial w_{jk} a_j}{\partial a_j} = w_{jk}$$

$$3)\ \frac{\partial a_j}{\partial z_j} = \frac{\partial f(z_j)}{\partial z_j} = f(z_j)(1 - f(z_j), \textit{(Dérivée de sigmoïd)}$$

FIGURE 3.8 – Calculs finaux

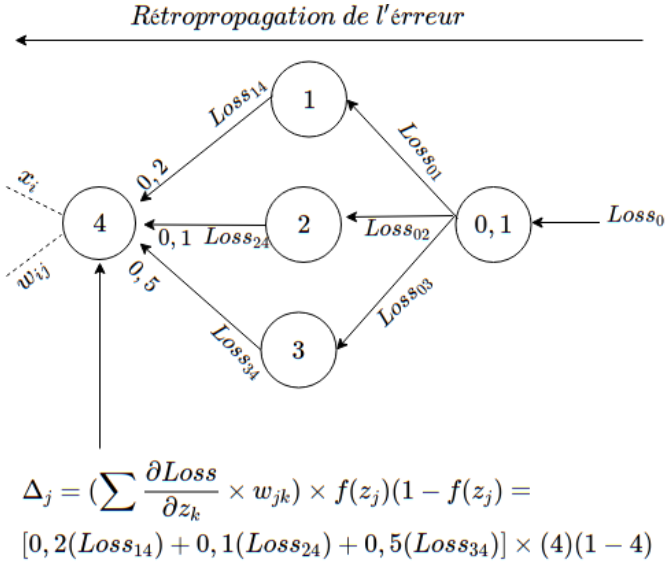
En remplaçant 2,3 dans l'expression 1 on obtient le résultat suivant.

$$\frac{\partial Loss}{\partial w_{ij}} = \underbrace{(\sum \frac{\partial Loss}{\partial z_k} \times w_{jk}) \times f(z_j)(1 - f(z_j)}_{\Delta_j} \times x_i$$

Pour finir, l'expression entière de la mise à jour des poids est la suivante.

$$w_{ij} = w_{ij} + \alpha(\Delta_j \times x_i)$$

Exemple de propagation de l'erreur : La figure 3.9 représente un exemple applicatif de la rétropropagation de l'erreur afin d'effectuer une mise à jour d'un seul poids :



$$\Delta_j = (\sum \frac{\partial Loss}{\partial z_k} \times w_{jk}) \times f(z_j)(1 - f(z_j)) =$$
$$[0,2(Loss_{14}) + 0,1(Loss_{24}) + 0,5(Loss_{34})] \times (4)(1 - 4)$$

FIGURE 3.9 – Exemple applicatif de la rétropropagation de la fonction *Loss*

3.4.2 Algorithme général

L'algorithme général dans l'apprentissage d'un réseau de neurones consiste à initialiser les poids et entrées, effectuer une "propagation avant" et calculer l'erreur de prédiction afin de la propager (rétropropagation) puis calculer les nouveaux poids jusqu'à satisfaction[39].

Tant que erreur $>$ 0 répéter :

/Initialisation des poids/
pour chaque poids w_{ij} de chaque couche faire :
$w_{ij} \in [-0,01, 0,01]$ par exemple
➥ $w_{ij} := valeur$

/Initialiser les neurones d'entrée/
pour chaque pixel p_i de l'image faire
➥ $x_i := p_i$

* $l = 1$ représente la couche d'entrée*
pour chaque couche $l = 2$ faire
pour chaque neurone j dans la couche l faire

/Propagation avant/
➥ $z_j := \sum w_{ij} x_i$
➥ $a_j := f(z_j)$

/Calculer l'erreur/
➥ $Loss = -\sum ylog(a_j)$

/Rétro-propagation/
l ici représente la dernière couche
pour chaque noeud j de l faire
calculer les dérivées des derniers neurones par raport à leurs entrées
➥ $\Delta_j := \frac{\partial Loss(z_j)}{\partial w_{ij}}$
$l - 1$ à partir de l'avant dernière couche
pour chaque $l = l - 1$ à 1 faire
calculer la dérivée de l'erreur pour chaque neud
➥ $\Delta_i := f(z_j))(1 - f(z_j) \sum w_{ij} \Delta_j$

/Mise à jour des poids/
pour chaque w_{ij} faire
➥ $w_{ij} := w_{ij} + \alpha(\Delta_j \times x_i)$

3.4.3 Conditions d'arrêt

Le nombre d'itérations est important car :

— S'il est trop faible, l'erreur n'est pas suffisamment réduite (sous-apprentissage).

— S'il est trop grand, le réseau devient spécifique aux données d'entraînement (sur-apprentissage).

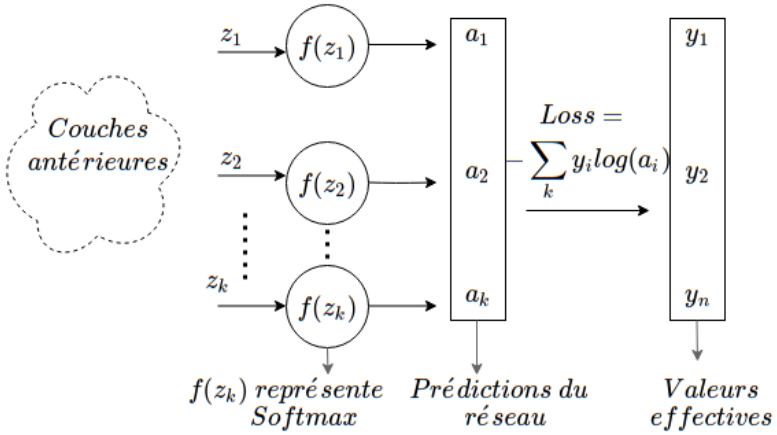
Il y a plusieurs conditions d'arrêt possibles :

— Lorsque les poids w se stabilisent (faibles mises à jour aux dernières itérations).

— Lorsque l'erreur a été assez réduite.

3.4.4 La Retropropagation avec softmax et cross-Entropy

L'exemple suivant représente l'opération de rétropropagation avec l'estimation de l'erreur Cross-Entropy pour la dernière couche $Softmax$.



Application :

$$\frac{\partial Loss}{z} = -\sum_k y_i \frac{\partial log(a_i)}{\partial z_k}, \text{ Loss ne dépend pas directement de } z \text{ donc :}$$

$$\frac{\partial log(a_i)}{\partial z_k} = \frac{\partial log(a_i)}{\partial a_i} \times \frac{\partial a_i}{\partial z_k}$$

$$\frac{\partial log(a_i)}{\partial a_i} = \frac{1}{a_i}, (\text{Dérivée de logarithme}) \quad \ldots\ldots\ldots 1$$

$$\frac{\partial a_i}{\partial z_k} = \begin{cases} \underbrace{a_i(1-a_i)}_{d_1} & si\ i = k \\ \underbrace{-a_i \times a_k}_{d_2} & si\ i \neq k \end{cases} \quad \ldots\ldots\ldots\ldots\ldots 2$$

- *Ici, a_i représente la fonction softmax, le résultat 2 représente donc l'expression de sa dérivée.*
- *En remplaçant 1 et 2 dans Loss on obtient :*

$$\boxed{\frac{\partial Loss}{\partial z} = -\sum_k y_i \frac{1}{a_i} \times (d_1 + d_2)}$$

FIGURE 3.10 – Exemple applicatif de rétropropagation de l'entropie-croisée

3.5 Les obstacles des réseaux de neurones

3.5.1 Sur-apprentissage, sous-apprentissage

La validation non-croisée

La première démarche de répartition du Data-set afin d'entraîner un modèle était d'utiliser un ensemble d'entraînement (70% de data Set) et un ensemble de test (30% Data-set) pour valider le réseau : c'est la validation non croisée. Avec cette approche, au fur et à mesure des itérations, l'erreur globale sur l'ensemble d'entraînement ($Loss_{train}$) continue de diminuer tandis que l'erreur sur l'ensemble de teste ($Loss_{test}$) augmente.

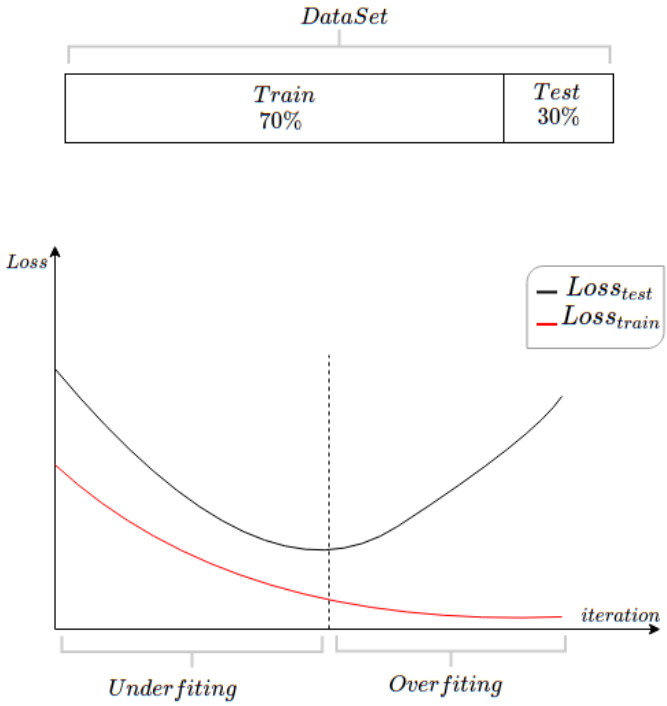


FIGURE 3.11 – Exemple de courbes en cas de validation non croisée

Ce phénomène est désigné par le terme sur-apprentissage (overfiting). Le modèle commence à mémoriser les images plutôt que de les prédire car les poids sont mis à jours trop de fois. Le réseau détecte alors des détails excédents.

Exemple : un modèle entraîné pour détecter une maison dans une image, fait du sur-apprentissage si il détecte les arbres, les nuages et d'autre détails superflus sur l'image analysée.

La validation croisée

La validation croisée ou (cross-validation) est une alternative d'échantillonnage afin d'éviter le sur-apprentissage.
Le principe est de subdiviser l'ensemble d'entraînement afin d'en soustraire un sous ensemble de validation, ceci étant pour introduire dans l'entraînement des informations (images) que le réseau ne connait pas encore pour créer de la variance et ainsi valider le modèle avant de passer au test.

Dans l'exemple suivant, on remarque une nette amélioration de l'erreur sur les sous-ensembles de données en utilisant la validation croisée.

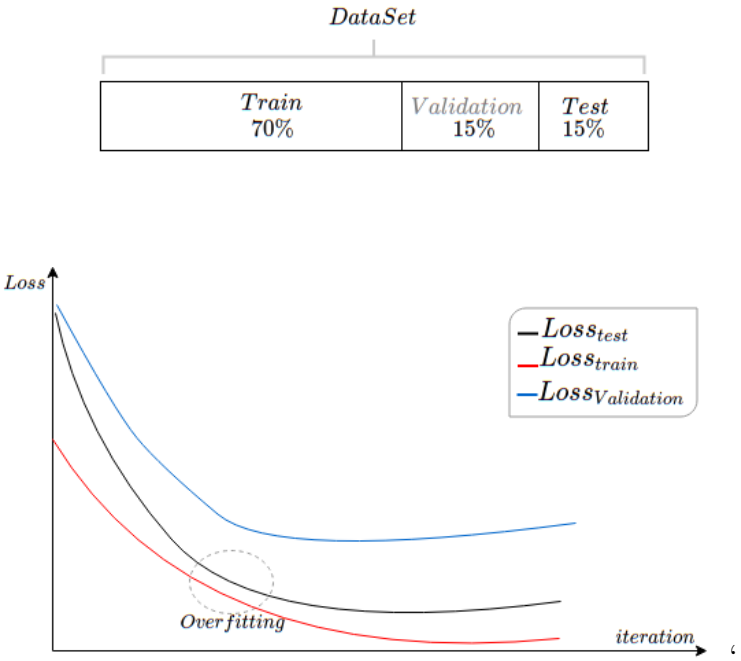


FIGURE 3.12 – Exemple de coubes en cas de validation croisée

Remarques : Dans la figure ci-haut, malgré l'utilisation de la validation croisée, on remarque la présence de sur-apprentissage. Afin d'éviter ce phénomène, plusieurs méthodes sont mises au point (régularisation).

Il est constaté ces dernières années que plus le réseau est profond (beaucoup de couches cachées) plus le modèle fait du sur-apprentissage [44].

3.5.2 La régularisation

Cette technique clé du ML vise à limiter le « sur-apprentissage » ou sous-apprentissage pour aboutir à de meilleures performances. Il existe de nombreuses formes de régularisation qui dépendent de l'objectif recherché et des hypothèses fixées sur le problème. Parmi ces formes on retrouve : Drop-out, Early stoping, L_1, L_2.

Le drop-out

Le Drop-out consiste à invalider des neurones de manière aléatoire, chaque neurone désactivé garde son poids mais ne contribue pas à la contabilisation de la couche suivante.

Remarques :

— Les biais ne sont pas invalidés.

— La dernière couche ne participe pas au Drop-out.

Régularisation L_1 et L_2

Un modèle avec moins de paramètres demande moins de puissance de calcul. Réduire les paramètres n'est par la seule solution pour rendre un modèle plus simple en effet, la régularisation dite L_1 ou L_2 permet de pénaliser l'impact d'un poids sur les réseaux car si ceux-là sont trop grands, ils ont un impact négativement grand parallèlement, si ils sont trop petits, ils ont un faible impact[46].

Régularisation L_1
Dans la régularisation L_1, au lieu de minimiser la fonction $Loss$, on cherche à minimiser la formule suivante.

$$Loss(w, f(z), y) = Loss(f(z), y) + \lambda \sum_i |w_i|$$

λ est considéré comme un hyperparamètre de pénalité car plus il est grand plus les poids faibles sont réduits à zéro[46].

Régularisation L_2
La régularisation L_2 est obtenue en minimisant la fonction d'erreur avec un terme de pénalité qui est le carré des poids w comme suit.

$$Loss(w, f(z), y) = Loss(f(z), y) + \lambda \sum_i w_i^2$$

Plus λ est grand plus les poids faibles tendent vers zéro.

Remarque : dans la régularisation L_2, les poids faibles n'atteignent jamais la valeur zéro[46].

3.5.3 Evanescence et explosion de gradient

L'evanescence de gradient (vanishing gradient) est un phénomène qui est initialement imputé aux fonctions d'activation dont les dérivées tendent vers 0 à leurs extrémités.
Nonobstant l'utilisation de fonctions qui résolvent ce problème (Leaky Relu), l'evanescence de gradient persiste et vient s'ajouter son opposé : l'explosion de gradient (exploding gradient). Tous deux sont causés par la profondeur abyssale qu'atteignent les réseaux de neurones de nos jours (d'ordre de centaines de couches).

Les recherches ont permis de détecter deux zones dans l'expression de la perte $\partial Loss$ de ce phénomène.

$$\frac{\partial Loss}{\partial w} = (\ldots\ldots\ldots \times w_{ij}) \times f'(z_j)\ldots\ldots\ldots$$

Dégradation liée à la profondeur du réseau — *Dégradation liée fonction d'activation*

Vanishing gradient et fonction d'activation

Dans la formule ci-haut, on retrouve la dérivée de la fonction d'activation résultante de l'application de règle de dérivation en chaîne. Cette dérivée tend vers zéro et toute l'expression s'annule.

1)• *Si* $z_j = -8$ *et* $f'(z_j) = f(z_j)(1 - f(z_j)$
alors $f'(-8) = 0,0002991 \approx 0$

• *En remplaçant dans le gradient on obtient :*

$$\frac{\partial Loss}{\partial w_{ij}} = \underbrace{(\ldots\ldots\ldots \times wij) \times 0 \times \ldots}_{=0}$$

Résultat : evanescence du gradient (gradient nul)

2)• *Si on remplace* 0 *dans la màj du poids on obtient :*

$$w_{ij} = w_{ij} \times \alpha \frac{\partial Loss}{\partial wij} ==> \quad w_{ij} = w_{ij} \times \alpha 0 \;\; Donc \;\; w_{ij} = w_{ij}$$

Résultat : pas de mise à jour du poids, le modèle n'apprend rien.

FIGURE 3.13 – Exemple applicatif d'un cas de gradient nul

vanishing gradient et profondeur du réseaux

Dans la formule de la dérivée de l'erreur, on retrouve une multiplication successive de poids ainsi :
Si $w < 1$, les gradients s'annulent.
Si $w > 1$ les gradients augmentent de manière quasi exponentielle.

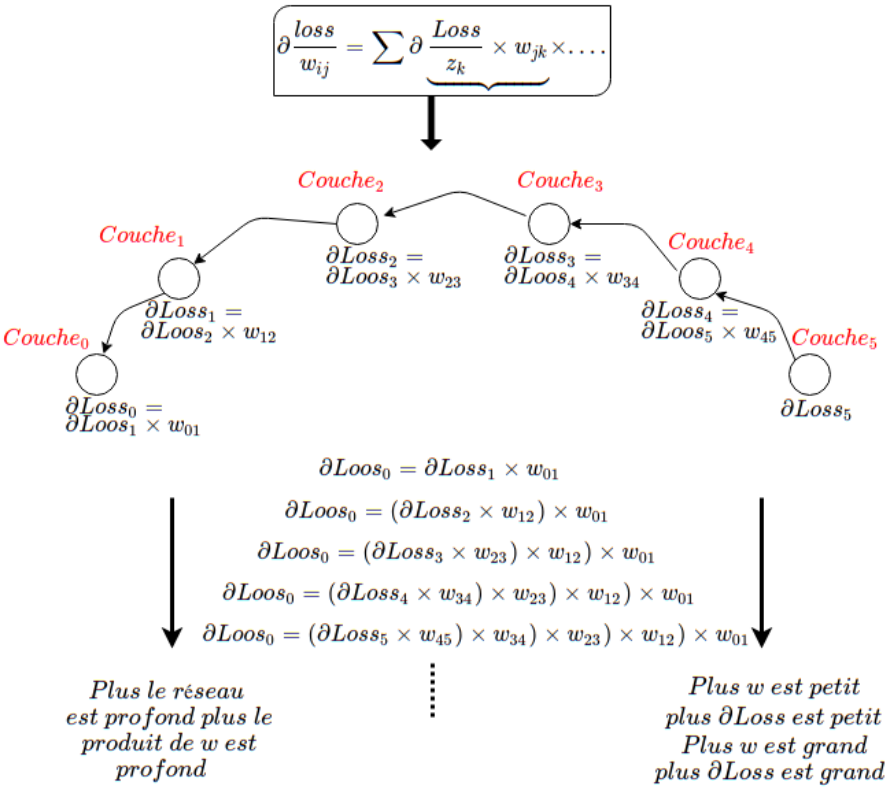


FIGURE 3.14 – Exemple d'une evanescence de gradient d'un réseau profond

3.5.4 Dégradation des réseaux très profonds

Théoriquement, plus un réseau de neurones est profond plus les résultats sont performants, cela est dû à l'augmentation de la complexité du problème traité.
En pratique, les travaux réalisés ces dernières années ont montrés que les réseaux profonds ont tendance à se dégrader par rapport à leurs homologues moins profonds. Il n'existe à ce jour aucune preuve formelle décrivant la raison de cette dégradation ni un modèle mathématique permettant de formaliser l'efficacité d'un réseau de neurones [47]. Il incombe donc aux utilisateurs d'étudier en profondeur les réseaux de neurones et leurs typologies afin de trouver une structure compatible à leurs problèmes de départ.

3.5.5 Réseaux de neurones et dimensionnalité de l'image

Les images tels que nous les connaissons aujourd'hui sont extrêmement coûteuses. Elles représentent un objet mathématique dans un espace de grande dimensions (d'ordre de million). Un tel espace est très vaste et contre intuitif, un réseau de neurones ne permet pas une analyse profonde et distinctive à moins d'atteindre une grande profondeur réseau.
Réduire la taille et l'expressivité en couleur des images ne permet pas de palier à cet obstacle car celles-ci restent très compliquées en terme de définition des formes.

Conclusion

Dans ce chapitre, nous avons étudié en profondeur les réseaux de neurones afin de déterminer des solutions applicatives à notre problème. Parmi les méthodes qui participent à l'élaboration de nos solutions on retrouve : la fonction softmax, Relu, Cross-entropy, la régularisation et la validation croisée.

Dans le chapitre suivant, nous étudirons le réseau de neurones convolutif et ses différentes fonctionalités.

Chapitre 4

Les réseaux de neurones covolutifs

Introduction

Un réseau de neurones convolutif (CNN) s'inspire du cortex visuel des mammifères afin d'analyser une image en profondeur. Cette architecture a été introduite par le chercheur "*Yann Le Cun*" Dans les années 90.

Dans ce chapitre, nous introduirons le réseau de neurones convoltif et les différentes algorithmes qui le distingue d'un réseau de neurones classique (DNN).

4.1 Motivations des réseaux de neurones convolutifs

Les principales motivations des CNN sont les suivantes :

1. Un DNN profond pour analyser une images complexe dégrade rapidement. Il faut trouver une méthode moins coûteuse et plus intuitive afin de faire du traitement d'image en général et la détection de forme en particulier.
2. Une image est composée de sous éléments, il faut donc exploiter cette hiérarchisation avec une détection de formes plus granulaire.
3. Les positions, tailles, couleurs.. etc sont compliqués à extraire pour les réseaux classiques afin d'avoir une compréhension générale de l'image.
4. Réduire La dimensionnalité des paramètres d'un DNN classique et celle des images .
5. S'inspirer de la biologie afin de créer une intelligence artificielle fut un succès, se tourner une seconde fois vers elle afin de modéliser une nouvelle IA peut être fructueux.

4.2 Le cortex visuel

Dans la définition suivante, à chaque étape, nous citerons entre parenthèses une notion analogue à celle des CNN.

L'œil perçoit la lumière (image en entrée) à travers la pupille puis véhicule l'information vers le cortex visuel qui se situe derrière la tête. Cette information est sous forme de signaux électriques dont la fréquence correspond à une information codée (convolution). Au cours de leurs trajets, les informations passent d'une fibre nerveuse à une autre grâce aux synapses (connexion de neurones). Le cortex visuel est constitué de plusieurs aires spécialisées dans le traitement des messages nerveux (les filtres). Selon la fréquence des signaux électriques émanant de l'information, chaque air interprète une caractéristique de l'image comme suit (cartes de caractéristiques)[48].

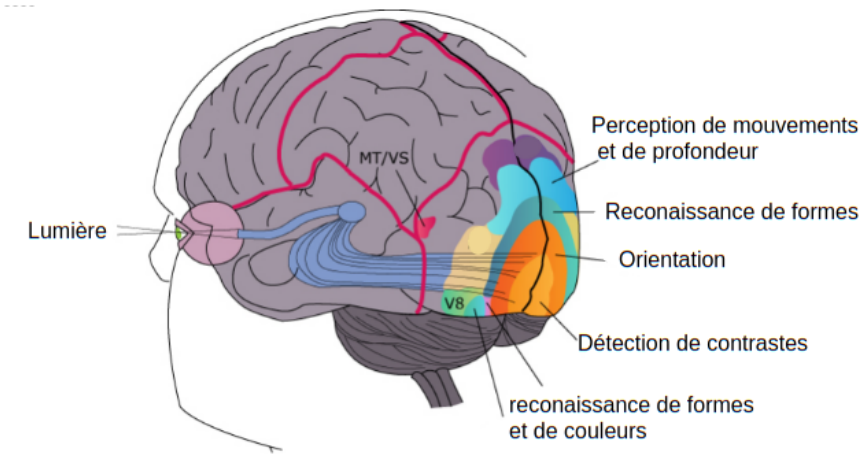


FIGURE 4.1 – Les différentes aires du cortex visuel

4.3 CNN et structure générale

Construire un CNN revient à superposer plusieurs couches de manière hiérarchique, chacune d'entre elles calcule une représentation abstraite de l'image au fur et à mesure [49]. L'opération se présente en général de la manière suivante.

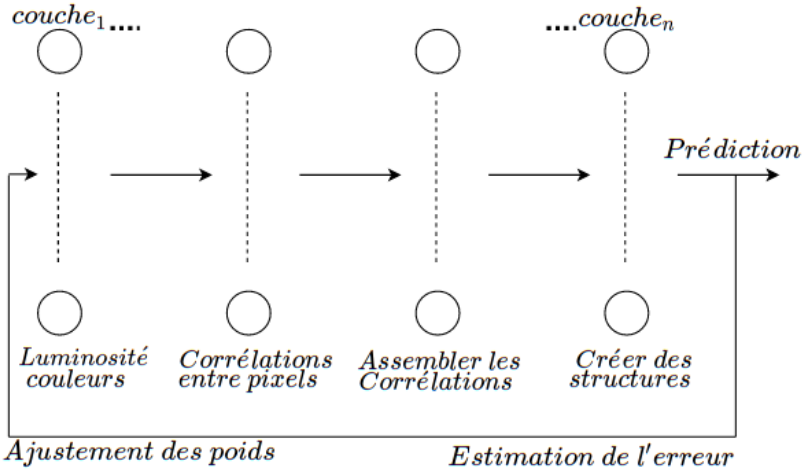


FIGURE 4.2 – Structure générale d'un CNN

4.4 CNN et fonctionnement

4.4.1 Les filtres (Kernel)

Un filtre est un petit groupe de neurones à partir desquels la couche suivante est connectée. Comme les filtres utilisés dans les traitements d'images classiques, ces derniers se distinguent des filtres ordinaires par leurs auto-apprentissages en effet, on ne connaît pas d'avance les paramètres qui les constituent, c'est au réseau de les apprendre [50].
➥Un filtre est une matrice de poids $n \times n$ qui analyse une partie $m \times m$ de l'image afin d'en extraire une valeur pertinente puis répéter le processus sur toute l'image.

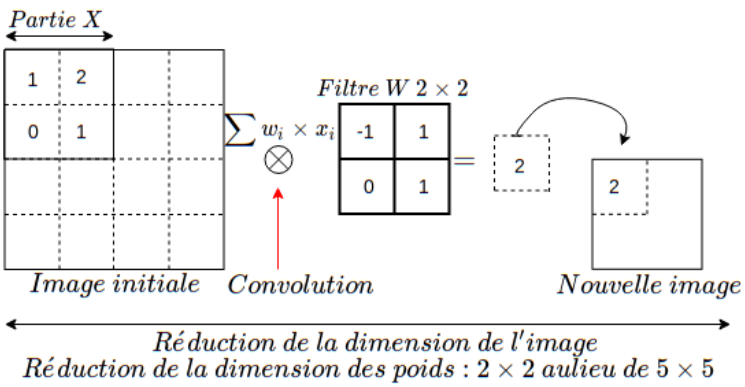


FIGURE 4.3 – Opération convolutive sur une image

4.4.2 Cartes de caractéristiques

Les cartes de caractéristiques (feature map) sont les images résultantes de l'opération convolutive.
Dans la pratique, plusieurs filtres sont appliqués à l'image de départ pour produire plusieurs cartes.
Chacune d'entre elles a pour rôle de détecter une particularité : contraste, lumière et autres en comparaison avec les régions du cortex visuel comme suit.

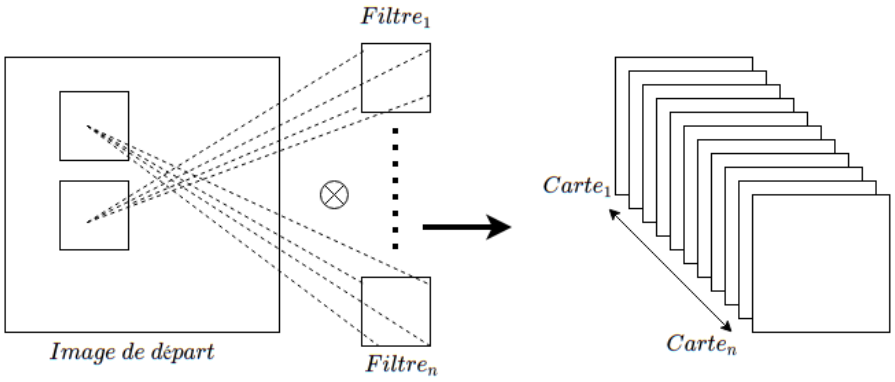


FIGURE 4.4 – Représentation générale des cartes de caractéristiques

La figure ci-après est un exemple d'une première convolution sur une image histologique d'un cancer du sein invasif. On remarque suite à une première couche de convolution une extraction générale des caractéristiques globales : couleur, contraste, luminosité...etc.

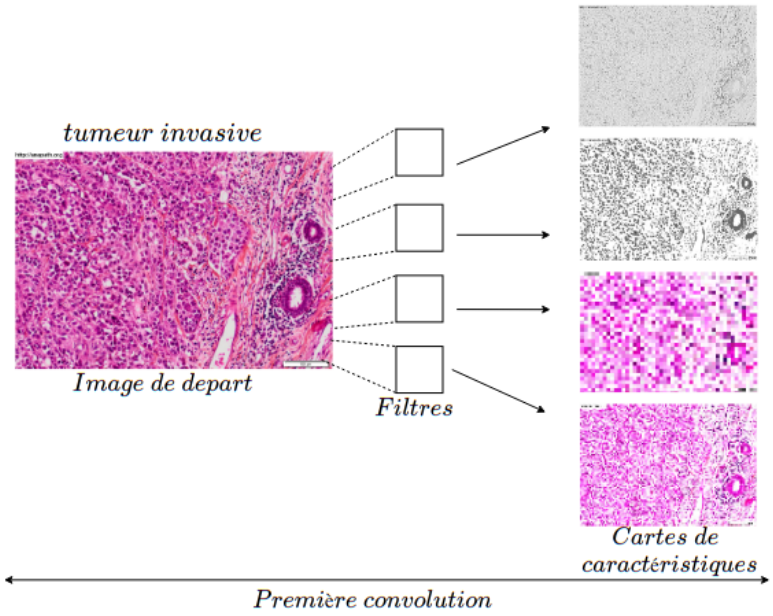


FIGURE 4.5 – Convolution sur une histologie

4.4.3 Profondeur de l'image et des filtres

Une image est pour une machine une matrice de $(n \times n \times k)$, n représente les lignes et les colonnes qui constituent ses pixels.
Chaque pixel dispose de 3 valeurs du système de codage RGB : c'est la dimension k. Par conséquent, un filtre est en réalité de taille $m \times m \times k$.

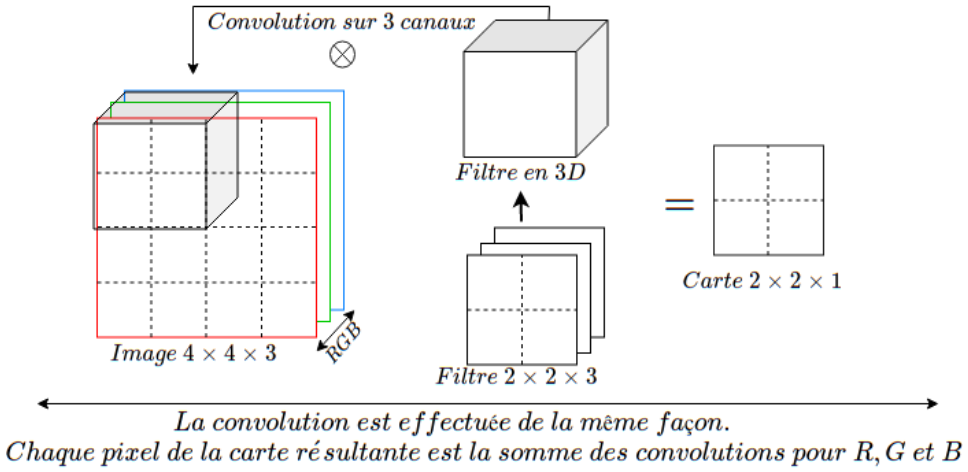


FIGURE 4.6 – Dimensions d'une image et des cartes de caractéristiques

Résultat : le nombre poids est drastiquement diminué en terme de lignes et colonnes mais augmenté en profondeur par le rajout de la 3 ème dimension k.
La dimension des images est également augmentée en profondeur à cause du nombre de cartes générées.
Un des objectifs primaires des CNN étant de réduire la dimension, celle-ci se voit augmentée[50].

4.4.4 Le Pooling

Le Pooling (mise en commun) est un algorithme visant à réduire la dimension d'une image tout en gardant des sous-parties importantes.
La fenêtre de Pooling est une matrice $m \times m$, qui parcour l'image afin de produire un seul pixel de sortie parmi $m \times m$ pixels voisins. Plusieurs méthodes de production de pixels sont mises en place en fonction de la pertinence de l'information voulue. Parmi ces méthodes, on compte l'average-Pooling, max et min-Pooling [44].

Average-Pooling

Le principe de l'average-Pooling est de prendre la moyenne des $m \times m$ pixels voisins. Dans le traitement d'image, on parle de " lissage". La fenêtre de Pooling la plus utilisée est de 2×2.

Max-Pooling

Le Max-Pooling permet d'extraire uniquement la plus grande valeur dans un ensemble de pixels. Ce type de Pooling est avantageux pour analyser une image dont le fond est noir. Le but est ainsi d'ignorer tous les pixels de faible valeur (en noir) et d'en extraire l'avant-plan (plus clair).

Min-Pooling

Le Min-Pooling permet d'extraire le pixel le plus faible parmi ses $m \times m$ pixels voisins. Cette méthode est adaptée dans les cas où le fond de l'image est clair et l'avant plan sombre.
➥ La figure suivante illustre les Différents résultats de Pooling et leurs effets réducteurs.

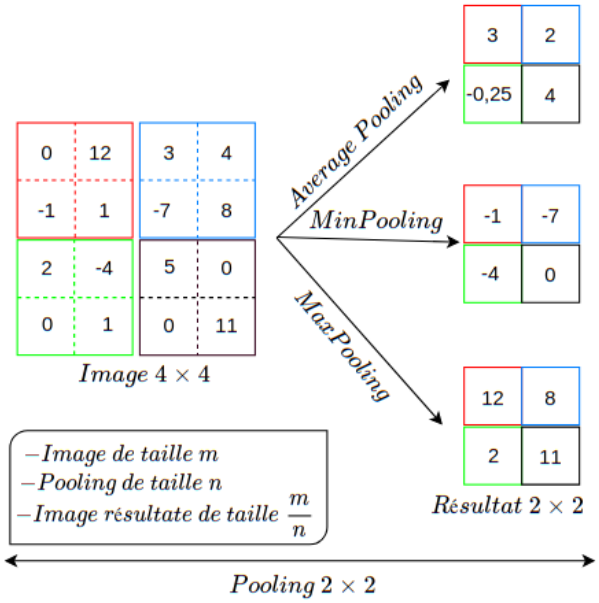


FIGURE 4.7 – Exemple applicatif des différents Pooling

Remarque : théoriquement, l'Average-Pooling semble être la méthode la plus adéquate car on ne perd ni trop ni peu d'informations. Cependant, les expériences ont montré que dans la plupart des problèmes de traitement d'images, le Max-Pooling semble être la plus adéquate. La figure suivante représente un test de différents Pooling sur une histologie.

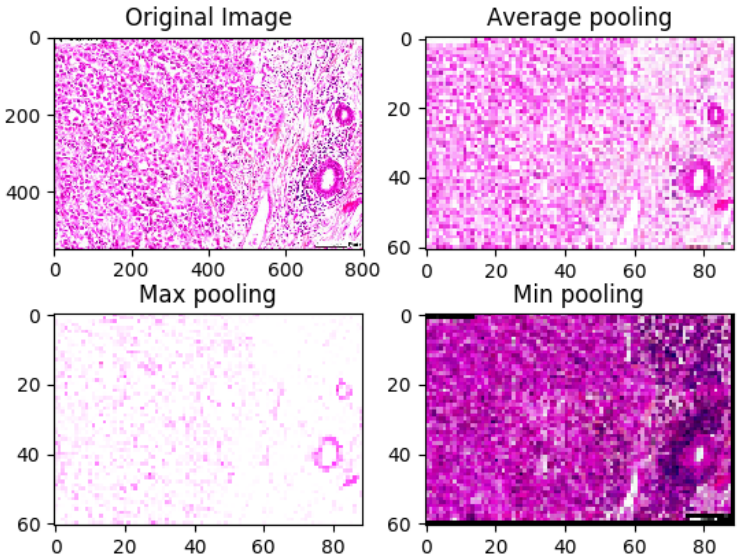


FIGURE 4.8 – Résultat de différents Pooling sur une histologie

Résultat : Le Max-Pooling permet d'extraire de meilleurs contours afin de détecter les anomalies (grosseur de cellules) tandis que l'Average-Pooling ne fait que lisser l'image. Quant au Min-Pooling, il permet que de réduire la luminosité de l'image, les contours sont donc négligés.

4.4.5 Le Stride

L'opération de convolution s'effectue en déplaçant successivement la fenêtre du filtre dans l'image. Le Stride est donc la taille du pas de déplacement en matière de ligne et colonne qu'on veut effectuer pour parcourir l'image. **Remarque** : Les

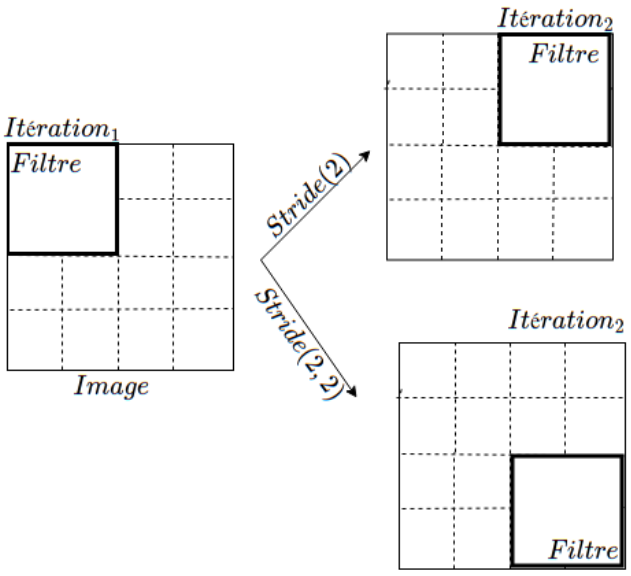


FIGURE 4.9 – Stride(ligne,colone)

Strides les plus utilisées sont de $(1, 1)$ ou $(2, 2)$ afin d'avoir une meilleure corrélation entre pixels.

4.4.6 Le zero-Padding

Le Zero-Padding est une technique qui permet d'ajouter des pixels à valeur nulle à chaque côté des frontières de l'image. Le but est de compléter celle-ci dans le cas où le Stride et la taille du filtre ne sont pas compatibles à celle de l'image.

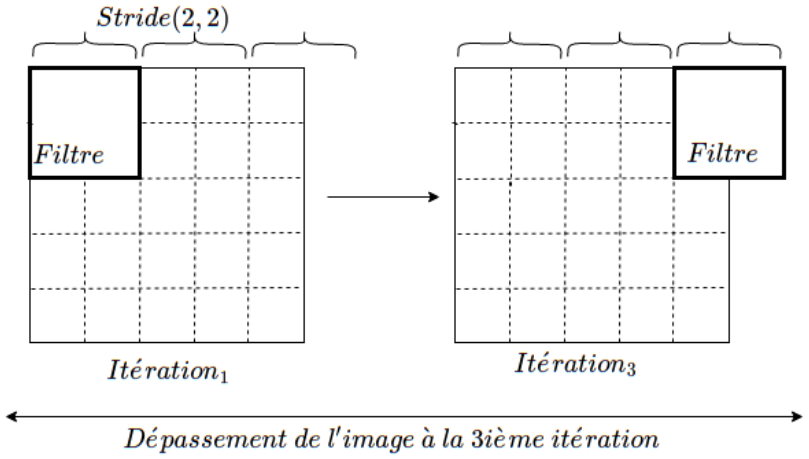


FIGURE 4.10 – Exemple de dépassement d'une image par un filtre

Les pixels aux extrémités de l'image sont faiblement traités par rapport aux pixels au centre suite au déplacement du filtre. La figure suivante illustre ce phénomène.

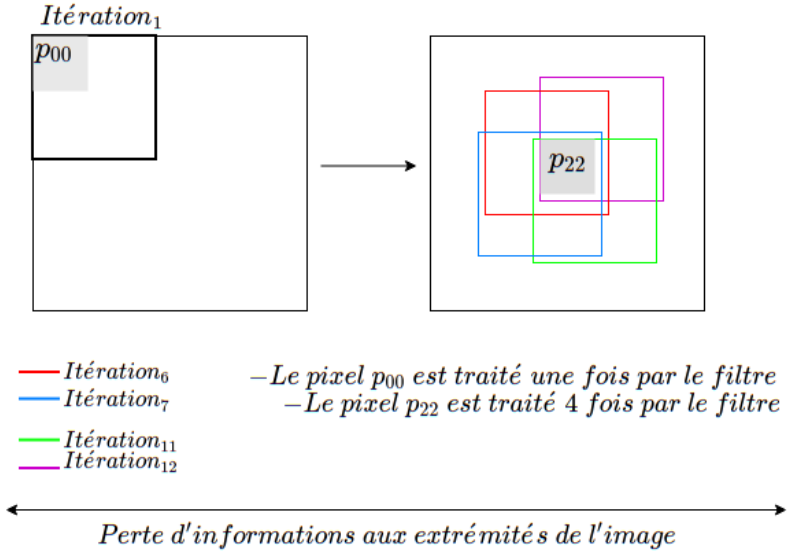
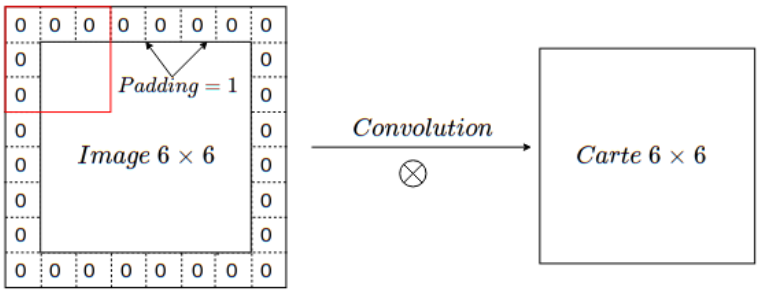


FIGURE 4.11 – Exemple de filtre passant par un pixel

Padding-valid, Padding-Same

Le Padding-Valid est caractérisé par l'absence de Padding. Quand la taille du filtre et de l'image adhère, il n'y a pas nécessité de rajouter des zéros. Le Padding-Same est introduit de sorte à avoir la taille de l'image en entrée égale à celle en sortie. C'est le plus utilisé car tous les pixels seront traités le même nombre de fois par le filtre [50].



Soit n : taille de l'image
f : taille du filtre
p : taille du padding
s : taille du stride

– Taille de la carte résultante est : $\frac{n+2p-f}{s}+1$

– Taille du padding – same est calculé comme suit : $p=\frac{f-1}{2}, (s=1)$

=> Application : $p=\frac{3-1}{2}=1$

– Taille du padding quand $s \neq 1$: $p=\frac{s(n-1)+f-n}{2}$

FIGURE 4.12 – Exemple applicatif d'un Padding-Same

4.5 Fonction Relu

L'opération de convolution sur une image impose de la linéarité à celle-ci, or les images sont naturellement non linéaires. Afin d'avoir une meilleure interprétation, Relu permet de casser cette linéarité. Pratiquement, la fonction relu supprime tous les éléments noirs, en ne gardant que ceux qui portent une valeur positive (les couleurs grises, blanches et autres).

De plus, quand deux couches de convolution sont appliquées, les poids des filtres de la 2ème convolution peuvent prendre soit des valeurs positives ou négatives. Si la carte de caractéristique de la 1ère convolution est négative, le produit d'une réponse négative et d'un poids de filtre négatif produira une valeur positive. Pourtant, le produit d'une réponse positive et d'un filtre positif produira également une valeur positive. En conséquence, le système ne peut pas différencier ces deux cas [51].

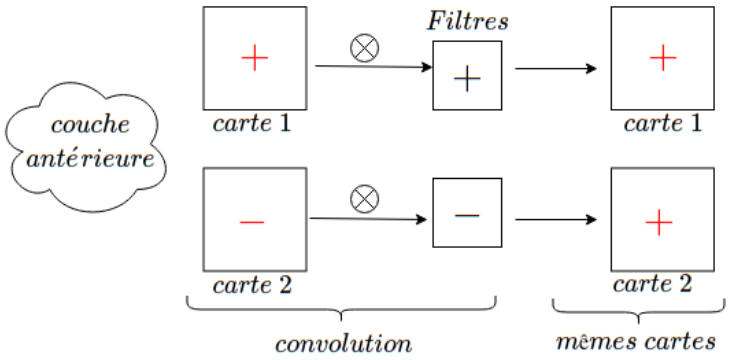


FIGURE 4.13 – Opération convolutive sans Relu

La figure suivante illustre l'impacte de la réctification linéaire sur les cartes de caractéristiques résultantes en effet, deux filtres de signes différents produisent deux cartes différentes.

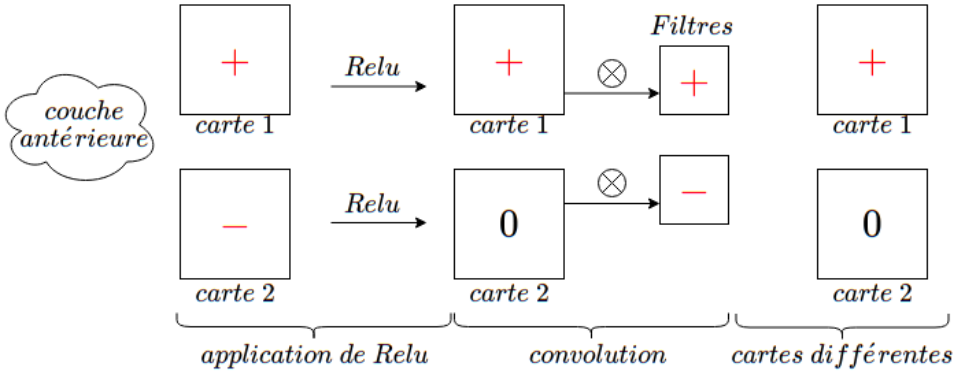


FIGURE 4.14 – Opération convolutive avec Relu

4.6 DNN final

Un CNN est une succession de couches convolution/Relu appliquées aux cartes de caractéristiques résultantes. Ces opérations s'achèvent par un réseau de neurones complètement connecté afin d'assembler toutes les informations apprises et ainsi effectuer les opérations classiques d'un DNN à savoir : prédiction, estimation de l'erreur, propagation de gradients puis mise à jour des poids.

4.7 Structure générale d'un CNN

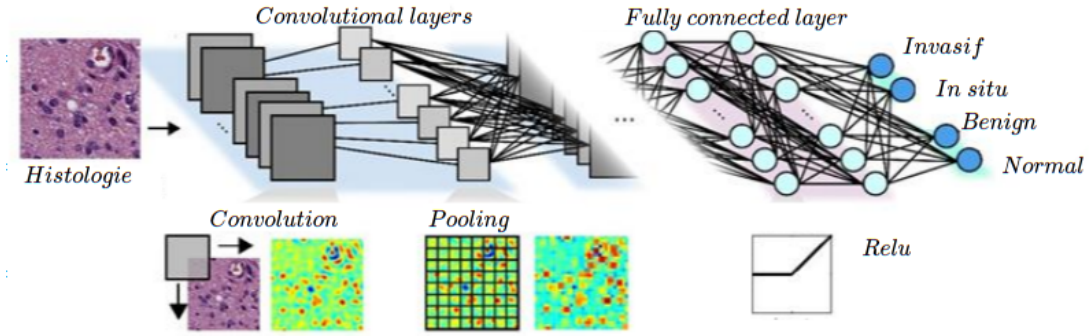


FIGURE 4.15 – Architecture générale d'un CNN

4.8 Objectif fondamentale d'un CNN

L'histologie étant un domaine contre intuitif pour les non-biologistes, nous avons choisi d'autres images pour illustrer le but de tout CNN traitant ces dernières.
La figure ci-après [52] montre que pour chaque partie de l'image, n sous parties sont constituées et détectées par des filtres. La phase de test consiste alors à appliquer tous les filtres appris pour détecter la présence ou non de formes voulues (yeux, nez, bouche..et autres). On remarque que plus les convolutions avancent, plus l'image est reconstituée en entier par les fitres.

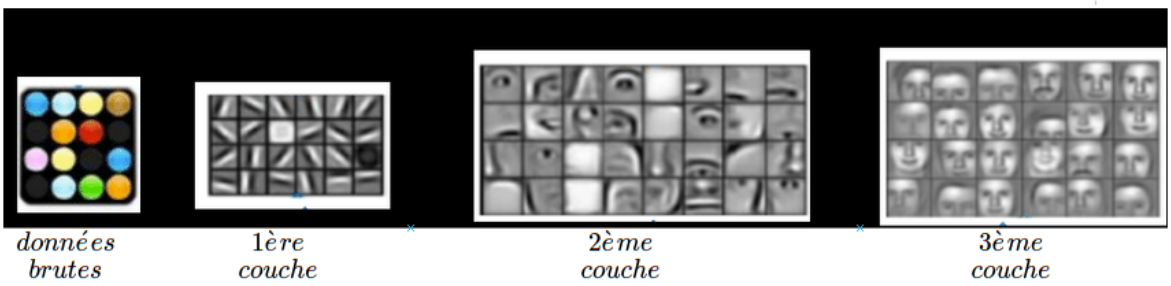


FIGURE 4.16 – Application d'un CNN pour la reconaissance faciale

Conclusion

Dans ce chapitre, nous avons étudié les fonctionnalités détaillées d'un réseau de neurones convolutif. Le plus important réside dans la différence entre un CNN et un DNN classique par la présence de filtres, de convolutions et les couches de pooling intercalées. Un CNN effectue grâce à cela une détection de formes plus granulaire qu'un DNN classique.

Les notions qui participent aux solutions CNN que nous proposeront sont les suivantes : la fonction Relu, Max-Pooling, Padding-Same, Stride(1,1).

Dans le chapitre réalisation, nous présenterons les différents modèles et méthodes appliquées sur notre base de données histologique, chaque solution sera suivie d'une discussion qui aboutira pas à pas à un modèle final.

Chapitre 5

Réalisation

Introduction

Une des plus grandes problématiques des réseaux de neurones est qu'il subsiste un écart conséquent entre la théorie et la pratique. Les méthodes étudiées dans les chapitres précédents ne garantissent pas un bon résultat empirique : tout dépend des données à disposition, de l'architecture du réseau, du matériel et autres.
Afin de réduire cet écart, nous avons opté pour une application graduelle de techniques que nous avons jugés potentiellement efficaces pour notre problème de départ.

Dans ce chapitre, nous présenterons nos résultats les plus importants, les obstacles rencontrés ainsi que les solutions appliquées afin d'aboutir à l'objectif visé à savoir un taux de précision satisfaisant pour un domaine aussi exigent que la classification d'images (histologies) du cancer du sein. Pour finir, nous définirons les différents outils ayant permis la construction de notre application. La synthèse de la réalisation est sommairement illustré dans la figure suivante.

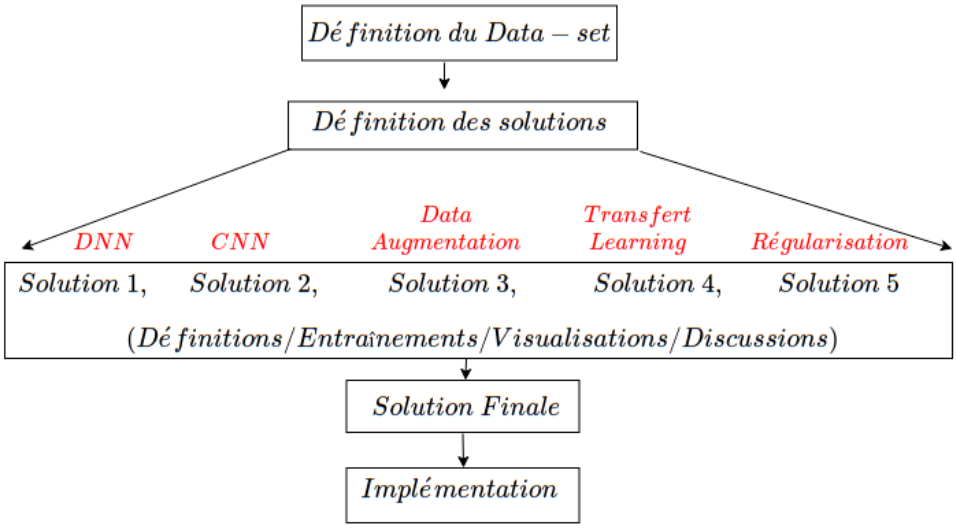


FIGURE 5.1 – Synthèse générale de la réalisation

5.1 Description du Data-set

Le Data-set est constitué de 2 dossiers : Data-set d'entraînement et de test.

5.1.1 Data-set d'entraînement

Le Data-set d'entraînement comporte quatre sous dossiers intitulés (en anglais) : normal, benign, invasive, insitu. Chacun d'eux correspond à une classe de cancer du sein et contient 100 images histologiques par classe.
Le Data-set d'entraînement constitue la base sur laquelle le modèle doit être entrainé afin de généraliser la solution et détecter le type de cancer du sein sur de futures images microscopiques issues d'un appareil médical d'histologie.

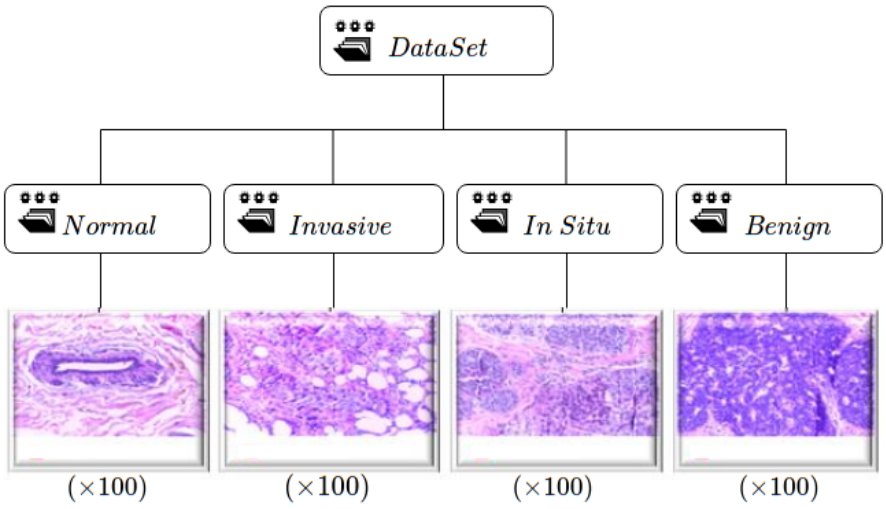


FIGURE 5.2 – Hiérarchie de l'ensemble d'entraînement

Les caractéristiques de chaque image sont les suivantes :

— Définition : 2048 pixels de longueur 1536 pixels de largeur.

— Chaque pixel est représenté par 3 valeurs numériques entre [0,255] qui correspondent respectivement à l'intensité des couleurs fondamentales RGB : rouge, vert, bleu.

— Taille = 20 mégaoctets.

— Extension : .tif (un format d'image utilisé dans l'imagerie médicale)
✔ la taille du Data-set d'entraînement = 13 gigaoctets.

5.1.2 Data-set de test

Pour vérifer la solution finale des compétiteurs, un Data-set de test a été fourni par l'organisation. Ce dossier contient 100 images appartenant aux différents types de cancers du sein, elles respectent la même définition et le même format que les images du data-set d'entraînement.

— Taille du Data-set de validation = 5 gigaoctets.

5.1.3 Prétraitement des données

Faute de moyens matériels (CPU, RAM capacité <= 8 gigaoctets), il est pratiquement impossible d'effectuer le traitement d'images d'une telle définition (2048*1536 pixels) et d'entraîner des modèles de ML très profonds capables de détecter les composantes élémentaires d'une histologie.

Pour remédier à ce problème, nous avons opté pour la solution **Cloud** gratuite de Google car elle offre les éléments suivants : **Google collaboratory ,Tensorflow 2.3.0** (dernière version).

5.1.4 Google collaboratory (colabs)

Google collabotary est un éditeur de code et de texte en ligne qui permet d'exécuter du code python et d'afficher le résultat sur des pages web interactives, ses capacités sont les suivantes :

— Un espace de stockage temporaire de taille >= 70 gigaoctets.

— différents processeurs de type CPU,GPU,TPU.

— Une RAM de taille >= 12 gigaoctets.

Remarques :

— Ces ressources sont extensibles en fonction des besoins et du nombre d'utilisateurs qui sont connectés à la fois.

— Les données peuvent être récupérées à partir de l'espace de stockage de notre machine locale, de l'espace collaboratif Github , de Google Drive ou de la mémoire temporaire de Google Colabs .

FIGURE 5.3 – Interaction machine/drive/google colabs

— afin d'accélérer le temps d'exécution des programmes et d'exploiter au maximum la puissance de calcul de cet environnement, on a uploadé notre Data-set d'entraînement vers Google Drive puis importé celui-ci vers la mémoire temporaire de Google Colabs et ceci à chaque session d'exécution.

5.1.5 Tensorflow 2.3.0 (dernière version)

La dernière version de tensorflow offre une bibliothèque qui permet de recharger directement les Data-sets d'entraînement et de validation d'une manière simple et rapide, il s'agit de la librairie **ImageDataGenerator**.
Pour effectuer le chargement , notre Data-set doit respecter la structure imposée par cette bibliothèque comme suit.

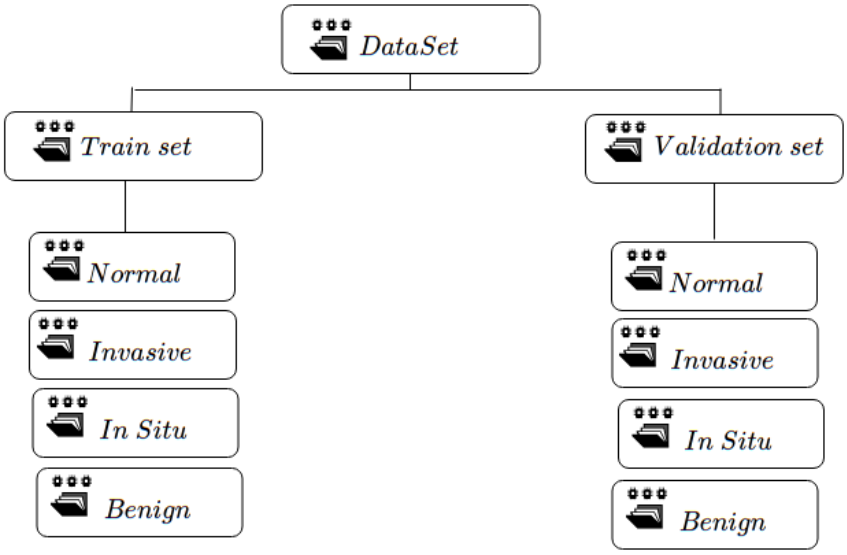


FIGURE 5.4 – Hiérarchie Tensorflow de Data-set entraînement

Afin de réaliser la structure ci-haut, nous avons développé une fonction qui permet de créer cette hiérarchie de dossiers dans la mémoire temporaire de **google collabs** et d'importer les images **google drive** puis les trier d'une manière aléatoire pour enfin les répartir sur les Data-set d'entraînement et de validation.

Remarque : Dans le but d'éviter d'altérer les données définitivement sur l'espace de stockage, d'autres modifications sur les données seront faites au fur et à mesure qu'on les injecte dans le modèle et cela au cours de la phase d'entraînement et de validation (redimensionnement, ajustement de couleur, normalisation de l'image..etc).

Le traitement des donnés achevé, nous pouvons désormais concevoir n'importe quel modèle et effectuer l'entraînement sur notre Data-set.

5.2 Première solution : Réseau de neurones profond (DNN)

Dans cette première approche, nous avons utilisé plusieurs réseaux de neurones complètement connectés. La différence réside dans la profondeur et taille de ces derniers.

5.2.1 Architecture générale

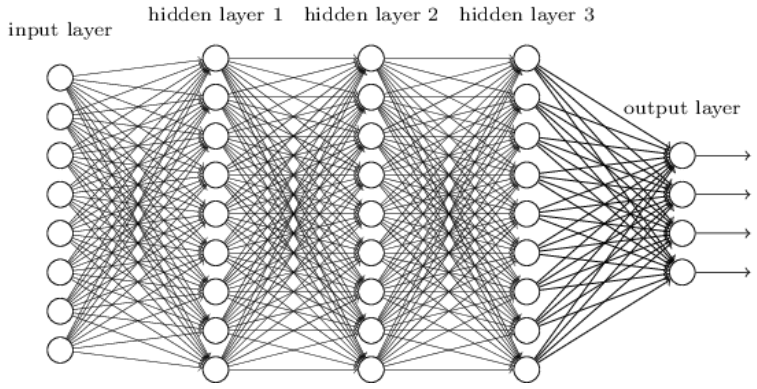


FIGURE 5.5 – Structure générale d'un DNN (3 couches)

5.2.2 Détails d'architectures

La précisions "accucary" permet de calculer le taux d'erreur d'un modèle lors d'une phase de test. (accuracy de 40% par exemple veux dire que sur 100 images, l'algorithme prédit 40 d'entre elles correctement).

	Modèle 1	Modèle 2
Nombre de couches	**4 (2 cachées)**	**5**
Nombre d'itérations	**79**	**118**
Neurones par couches	**(300*300*3)*512*128*4**	**(300*300*3)*1024*128*4**
Répartition du Data-set	80% entraînement/ 20% valudation	80% / 20%
Taux d'apprentissage α	Par défaut (0,001)	Par défaut (0,001)
Taille du batch	32 entraînement / 16 validation	32 / 16
Algorithme d'optimisation	Adam	Adam
Régularisation	non	non
Taille de l'image	$(300 * 300 * 3)$	$(300 * 300 * 3)$
Résulat (accuracy)	**100%**entranînement / **43%** validation	**100%** / **48%**

5.2.3 Entraînement

```
Found 320 images belonging to 4 classes.
Found 80 images belonging to 4 classes.
start date and time :
2020-09-03 03:35:45
Epoch 1/600
10/10 [==============================] - 7s 698ms/step - loss: 253.5996 - accuracy: 0.2313 - val_loss: 293.5790 - val_accuracy: 0.2500
Epoch 2/600
10/10 [==============================] - 7s 657ms/step - loss: 234.5489 - accuracy: 0.2375 - val_loss: 245.6880 - val_accuracy: 0.2500
Epoch 3/600
10/10 [==============================] - 7s 660ms/step - loss: 145.3775 - accuracy: 0.2781 - val_loss: 65.1455 - val_accuracy: 0.2500
Epoch 4/600
10/10 [==============================] - 6s 644ms/step - loss: 48.2326 - accuracy: 0.2750 - val_loss: 42.3511 - val_accuracy: 0.2500
Epoch 5/600
10/10 [==============================] - 6s 637ms/step - loss: 27.9808 - accuracy: 0.2500 - val_loss: 19.4050 - val_accuracy: 0.2500

                                    .....

10/10 [==============================] - 7s 662ms/step - loss: 0.0055 - accuracy: 0.9969 - val_loss: 2.4294 - val_accuracy: 0.5250
Epoch 76/600
10/10 [==============================] - 6s 639ms/step - loss: 0.0221 - accuracy: 0.9906 - val_loss: 2.8861 - val_accuracy: 0.4875
Epoch 77/600
10/10 [==============================] - 6s 649ms/step - loss: 0.0320 - accuracy: 0.9844 - val_loss: 2.8294 - val_accuracy: 0.5000
Epoch 78/600
10/10 [==============================] - 7s 658ms/step - loss: 0.0313 - accuracy: 0.9875 - val_loss: 2.5256 - val_accuracy: 0.5250
Epoch 79/600
10/10 [==============================] - ETA: 0s - loss: 0.0061 - accuracy: 1.0000
Reached 99.9% accuracy so cancelling training!
10/10 [==============================] - 6s 641ms/step - loss: 0.0061 - accuracy: 1.0000 - val_loss: 2.6356 - val_accuracy: 0.4375
start date and time :
2020-09-03 03:45:06
```

FIGURE 5.6 – Entraînement de la première solution (DNN)

5.2.4 Visualisation des résultats

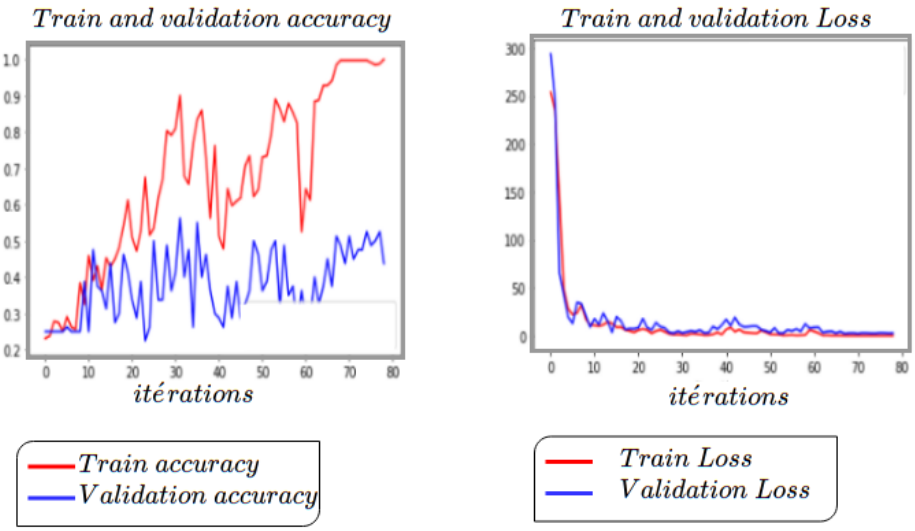


FIGURE 5.7 – Entraînement d'un DNN

5.2.5 Analyse et critique de la méthode

Le modèle sur-apprend les données d'entraînement et ne généralise pas le résultat aux données de validation également, plus le nombre de couches augmente plus le score de validation augmente cependant, pour atteindre le score souhaité le réseau doit être suffisamment profond, cela nécessite des ressources matériels très puissantes chose qui reste de nos jours fréquemment hors de portée. Dans la perspective de remédier à ce problème, nous utiliserons les CNN.

5.3 Deuxième solution : réseau de neurones convulutif (CNN)

Dans cette solution, nous avons conçu un réseau de neurones convolutif avec 3 couches à convolution suivie chacune par un max-pooling, d'une couche complétement connectée de 512 neurones et d'une dernière couche de classification de 4 neurones.

5.3.1 Sommaire du modèle

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 298, 298, 16)	448
max_pooling2d (MaxPooling2D)	(None, 149, 149, 16)	0
conv2d_1 (Conv2D)	(None, 147, 147, 32)	4640
max_pooling2d_1 (MaxPooling2	(None, 73, 73, 32)	0
conv2d_2 (Conv2D)	(None, 71, 71, 64)	18496
max_pooling2d_2 (MaxPooling2	(None, 35, 35, 64)	0
flatten (Flatten)	(None, 78400)	0
dense (Dense)	(None, 512)	40141312
dense_1 (Dense)	(None, 4)	2052

Total params: 40,166,948
Trainable params: 40,166,948
Non-trainable params: 0

FIGURE 5.8 – Sommaire de la deuxième solution (CNN)

5.3.2 Détails d'architecture

	Modèle 1	Modèle 2
Nombre de couches	5	5
Nombre d'itérations	**19**	**22**
Répartition du Data-set	**80% entraînement/ 20% valudation**	**70% / 30%**
Taux d'apprentissage α	Par défaut (0,001)	Par défaut (0,001)
Taille du batch	**32 entraînement / 16 validation**	**70 / 60**
Algorithme d'optimisation	Adam	Adam
Régularisation	non	non
Taille de l'image	$(300 * 300 * 3)$	$(300 * 300 * 3)$
Résulat (accuracy)	**99%**entranînement / **49%** validation	**99% / 52%**

5.3.3 Entraînement

```
Found 280 images belonging to 4 classes.
Found 120 images belonging to 4 classes.
start date and time :
2020-09-04 21:32:38
Epoch 1/600
4/4 [==============================] - 55s 14s/step - loss: 9.8369 - accuracy: 0.2357 - val_loss: 2.1165 - val_accuracy: 0.2500
Epoch 2/600
4/4 [==============================] - 29s 7s/step - loss: 1.7439 - accuracy: 0.2821 - val_loss: 1.4389 - val_accuracy: 0.2500
Epoch 3/600
4/4 [==============================] - 13s 3s/step - loss: 1.3890 - accuracy: 0.3179 - val_loss: 1.3621 - val_accuracy: 0.2500
Epoch 4/600
4/4 [==============================] - 5s 1s/step - loss: 1.3601 - accuracy: 0.2500 - val_loss: 1.3560 - val_accuracy: 0.2500
Epoch 5/600
4/4 [==============================] - 5s 1s/step - loss: 1.3320 - accuracy: 0.3750 - val_loss: 1.2938 - val_accuracy: 0.4000
Epoch 6/600
4/4 [==============================] - 5s 1s/step - loss: 1.2547 - accuracy: 0.4607 - val_loss: 1.1921 - val_accuracy: 0.5417
Epoch 7/600
4/4 [==============================] - 5s 1s/step - loss: 1.1715 - accuracy: 0.5643 - val_loss: 1.1381 - val_accuracy: 0.4833
Epoch 8/600
4/4 [==============================] - 5s 1s/step - loss: 1.2373 - accuracy: 0.4143 - val_loss: 1.4242 - val_accuracy: 0.2917
Epoch 9/600
4/4 [==============================] - 5s 1s/step - loss: 1.1415 - accuracy: 0.4857 - val_loss: 1.2277 - val_accuracy: 0.3333
Epoch 10/600
4/4 [==============================] - 5s 1s/step - loss: 1.0918 - accuracy: 0.6607 - val_loss: 1.1945 - val_accuracy: 0.4167
Epoch 11/600
4/4 [==============================] - 5s 1s/step - loss: 1.0023 - accuracy: 0.5571 - val_loss: 1.1272 - val_accuracy: 0.5083
Epoch 12/600
4/4 [==============================] - 5s 1s/step - loss: 0.8485 - accuracy: 0.7214 - val_loss: 1.0841 - val_accuracy: 0.5667
Epoch 13/600
4/4 [==============================] - 5s 1s/step - loss: 0.7695 - accuracy: 0.7429 - val_loss: 1.0975 - val_accuracy: 0.4500
Epoch 14/600
4/4 [==============================] - 5s 1s/step - loss: 0.7115 - accuracy: 0.7464 - val_loss: 1.0278 - val_accuracy: 0.5333

    .....
Epoch 15/600
4/4 [==============================] - 5s 1s/step - loss: 0.6220 - accuracy: 0.8107 - val_loss: 1.0925 - val_accuracy: 0.5417
Epoch 16/600
4/4 [==============================] - 5s 1s/step - loss: 0.5217 - accuracy: 0.8393 - val_loss: 1.1660 - val_accuracy: 0.4833
Epoch 17/600
4/4 [==============================] - 5s 1s/step - loss: 0.4533 - accuracy: 0.8679 - val_loss: 1.1318 - val_accuracy: 0.4917
Epoch 18/600
4/4 [==============================] - 5s 1s/step - loss: 0.3652 - accuracy: 0.9107 - val_loss: 1.0457 - val_accuracy: 0.5583
Epoch 19/600
4/4 [==============================] - 5s 1s/step - loss: 0.2708 - accuracy: 0.9429 - val_loss: 1.1421 - val_accuracy: 0.5167
Epoch 20/600
4/4 [==============================] - 5s 1s/step - loss: 0.2015 - accuracy: 0.9750 - val_loss: 1.1995 - val_accuracy: 0.5333
Epoch 21/600
4/4 [==============================] - 5s 1s/step - loss: 0.1531 - accuracy: 0.9786 - val_loss: 1.1501 - val_accuracy: 0.5000
Epoch 22/600
4/4 [==============================] - ETA: 0s - loss: 0.1108 - accuracy: 0.9929
Reached 99% accuracy so cancelling training!
4/4 [==============================] - 5s 1s/step - loss: 0.1108 - accuracy: 0.9929 - val_loss: 1.4435 - val_accuracy: 0.5167
end date and time :
2020-09-04 21:36:46
```

FIGURE 5.9 – Entraînement de la deuxième solution (CNN)

5.3.4 Visualisation des résultats

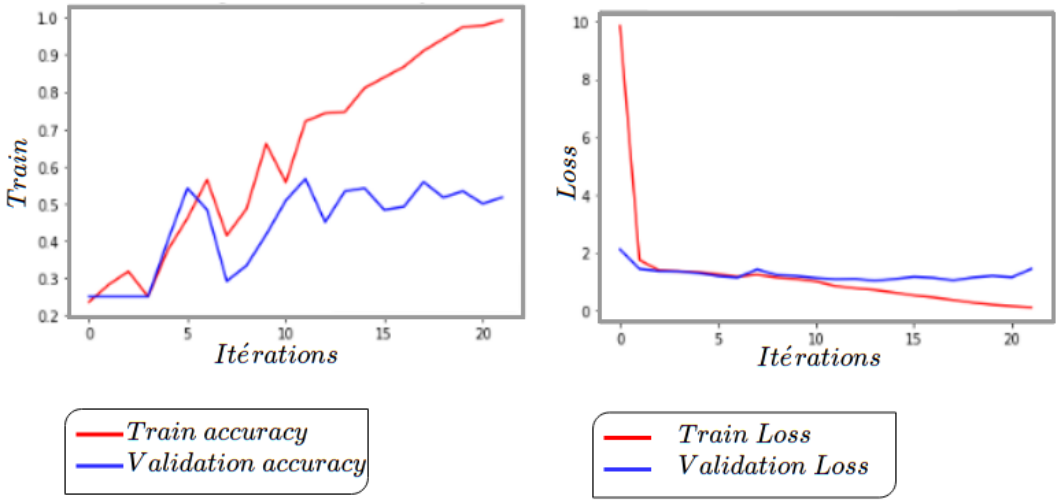


FIGURE 5.10 – Résultat de la deuxième solution (CNN)

5.3.5 Analyse et critique de la méthode

Nous remarquons une légère amélioration par rapport au modèle précédant que ce soit en terme de résultats ou en temps d'exécution car au bout de 22 itérations sur le Data-set, le modèle (CNN) a réussi à apprendre et distinguer entre les différents types de cancers du sein figurant dans les images du Data-set d'entraînement. Le modèle a également pu généraliser la solution à 52% sur le Data-set de validation.

Malgré cette amélioration, le résultat n'est pas satisfaisant étant donné l'écart entre le score de l'entraînement (99%) et le score de validation (52%) et cela même suite à restructuration du Data-set dans le 2 ème modèle (70% des données pour l'entraînement et 30 % pour la validation) : notre modèle est toujours en sur-apprentissage. ✔ Afin de résoudre ce problème, nous avons appliqué les solutions suivantes : Data augmentation, Régularisation, Transfer Learning.

5.4 troisième sollution : Data augmentation

La data augmentation (augmentation de données) est une sollution qu'il faut utiliser avec prudence. Son rôle est d'améliorer la qualité de nos prédictions de façon significative en augmentant artificiellement le volume de données d'entraînement. Elle consiste à appliquer des transformations aux données avant de les utiliser pour l'entraînement tout en conservant leurs structures. Ainsi, pour chaque image, les transformations peuvent être : des rotations, déformations, recadrages, changements de couleurs, ajout de bruit ou autres [55].

Exemple applicatif de la Data augmentation

L'application de data augmentation sur une histologie issue de nos donnnées abouti aux images suivantes.

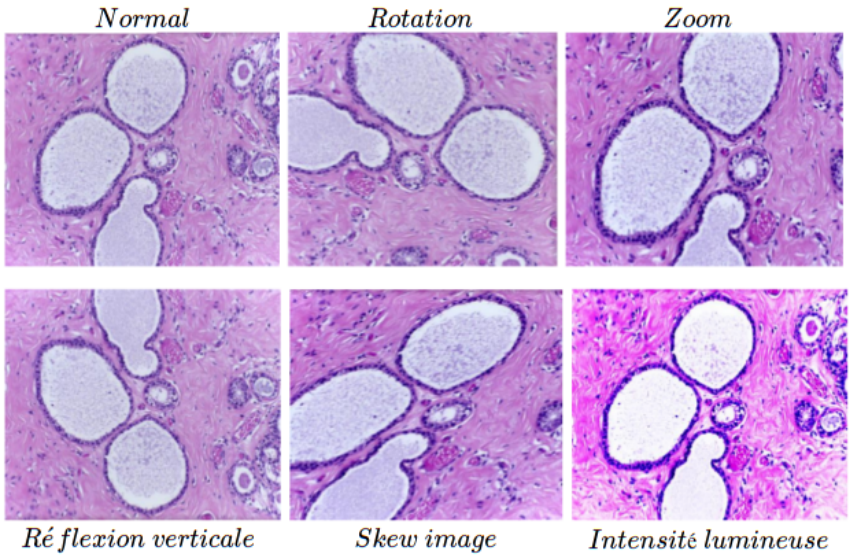


FIGURE 5.11 – Résultats d'une Data Augmentation sur une histologie

5.4.1 Entraînement

```
Found 280 images belonging to 4 classes.
Found 120 images belonging to 4 classes.
start date and time :
2020-09-04 22:49:22
Epoch 1/600
4/4 [==============================] - 108s 27s/step - loss: 8.2162 - accuracy: 0.2464 - val_loss: 1.9255 - val_accuracy: 0.2667
Epoch 2/600
4/4 [==============================] - 113s 28s/step - loss: 1.4970 - accuracy: 0.3000 - val_loss: 1.4334 - val_accuracy: 0.2500
Epoch 3/600
4/4 [==============================] - 88s 22s/step - loss: 1.3826 - accuracy: 0.3071 - val_loss: 1.3770 - val_accuracy: 0.2500
Epoch 4/600
4/4 [==============================] - 52s 13s/step - loss: 1.3854 - accuracy: 0.2607 - val_loss: 1.3575 - val_accuracy: 0.3750
Epoch 5/600
4/4 [==============================] - 49s 12s/step - loss: 1.3505 - accuracy: 0.3321 - val_loss: 1.3135 - val_accuracy: 0.2917

    ..................................
Epoch 596/600
4/4 [==============================] - 9s 2s/step - loss: 0.0966 - accuracy: 0.9607 - val_loss: 2.4974 - val_accuracy: 0.6500
Epoch 597/600
4/4 [==============================] - 9s 2s/step - loss: 0.0881 - accuracy: 0.9714 - val_loss: 2.4858 - val_accuracy: 0.6417
Epoch 598/600
4/4 [==============================] - 9s 2s/step - loss: 0.1368 - accuracy: 0.9500 - val_loss: 1.9429 - val_accuracy: 0.7000
Epoch 599/600
4/4 [==============================] - 9s 2s/step - loss: 0.1006 - accuracy: 0.9571 - val_loss: 1.9694 - val_accuracy: 0.6833
Epoch 600/600
4/4 [==============================] - 9s 2s/step - loss: 0.1304 - accuracy: 0.9607 - val_loss: 2.3442 - val_accuracy: 0.6917
end date and time :
2020-09-05 00:52:32
```

FIGURE 5.12 – Entraînement de la troisième solution (Data Augmentation)

5.4.2 Visualisation des résultats

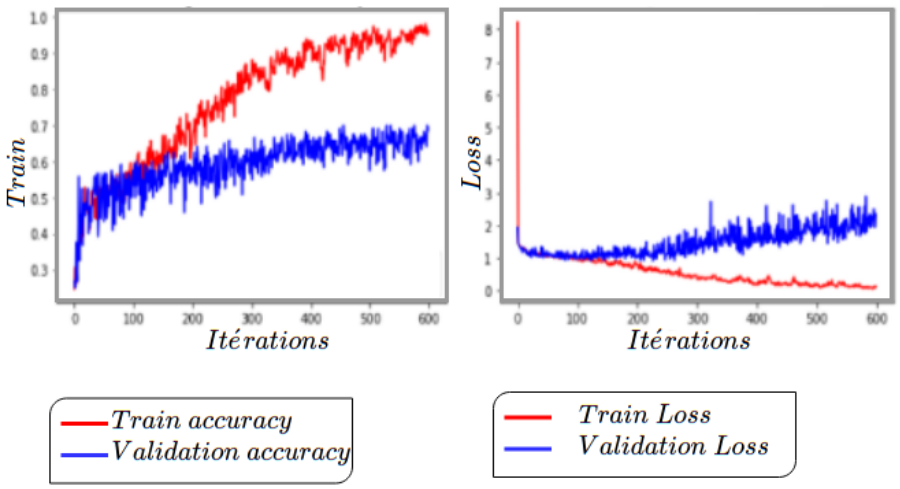


FIGURE 5.13 – Résultat de la troisième solution (Data Aaugmentation)

5.4.3 Analyse et critique de la méthode

Grâce à l'augmentation des données, nous avons réussi à augmenter le score de validation à 69% et réduire ainsi l'écart entre le score de l'entraînement (96%) et le score le score de validation à 27% malgré la lenteur de l'entraînement.
Remarque : L'écart est encore très grand de plus, le score de validation n'est pas optimal. Le problème de sur-apprentissage persiste, afin de diminuer la spécificité sur nos données, nous utiliserons dans la prochaine solution le Transfert Learning.

5.5 Qautrième solution : Transfer Learning

Lors de l'apprentissage d'un modèle de classification pour un domaine cible avec seulement une petite quantité de données, l'application des algorithmes ML conduit généralement à des classifieurs surdimensionnés avec de mauvaises compétences de généralisation. D'autre part, recueillir un nombre suffisant de données étiquetés manuellement peut s'avérer très coûteux.

5.5.1 Objectif du Transfert Learning

Les méthodes de Transfert Learning (transfert d'apprentissage) visent à résoudre ce type de problèmes en important des connaissances (poids) provenant d'un domaine source qui contient beaucoup plus de données pour faciliter la classification dans le domaine cible[56][57].
Dans notre cas, nous avons utilisé plusieurs modèles pré-entraînés sur des millions d'images récupérées sur Imagenet pour classifier plus de 1000 objets, un tel transfert est illustré sommairement comme illustré ci-après, chaque numéro représente une étape que nous allons décrire.

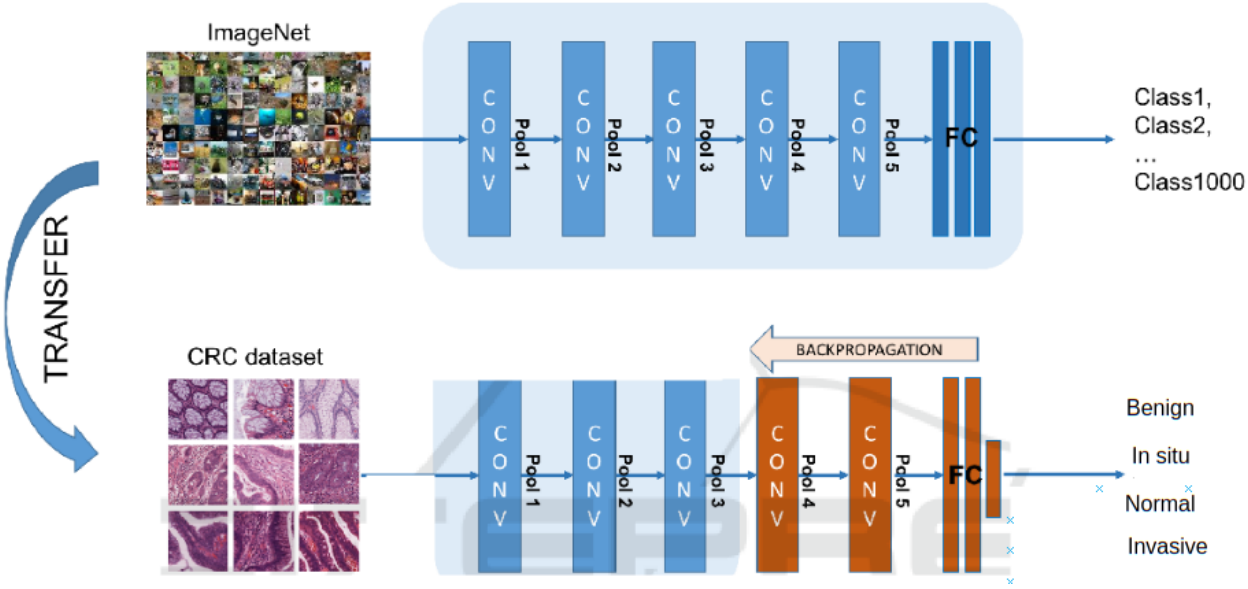


FIGURE 5.14 – Systématique générale d'un Transfert Learning

Imagenet est choisi car il représente la bases de données la plus riche disponible en ligne et en accès libre. Parmi les modèles testés sur Imagenet : $inception_{V3}$, Resnet , InceptionResnet , Nasnetlarge. Nous citerons $inception_{V3}$ dans la quatrième solution.

5.5.2 Archetitecture $incetion_{V3}$

$Inception_{V3}$ est un modèle de reconnaissance d'images couramment utilisé qui a démontré sur l'ensemble de données ImageNet une justesse supérieure à 78%[58]. Le modèle est constitué de composants de base symétriques et asymétriques incluant convolutions, avarage-pooling/max-pooling/concaténations et couches entièrement connectées. La normalisation par lots (batchNorm) est amplement utilisée dans le modèle et appliqué aux entrées d'activation. La perte est calculée via Softmax [59]. Un diagramme général du modèle est présenté ci-dessous.

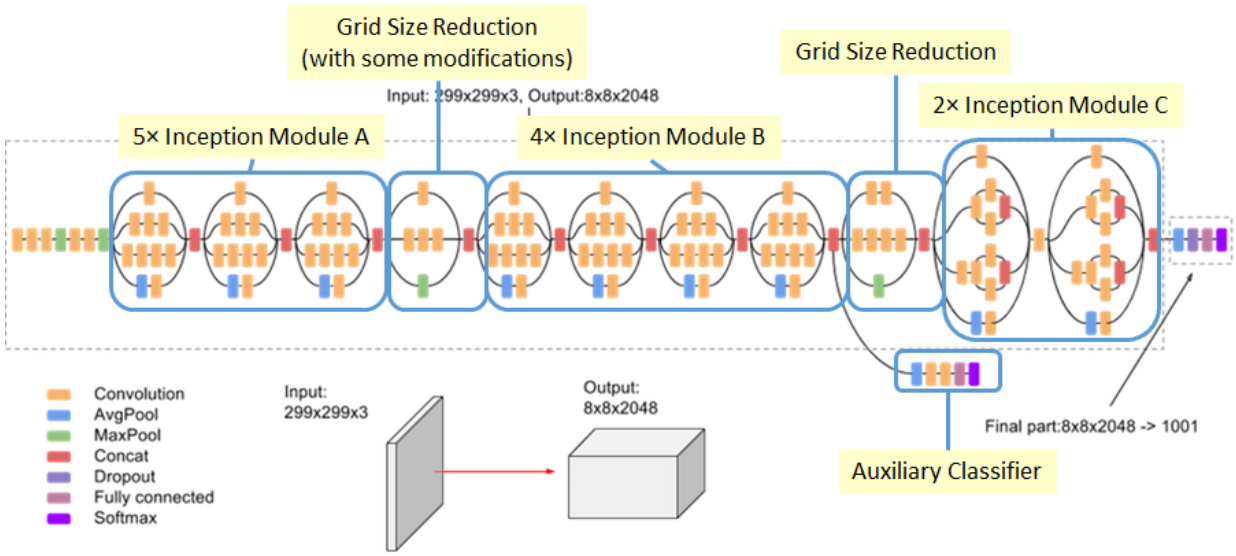


FIGURE 5.15 – Architecture de Inception version 3

5.5.3 Le transfert de $inception_{v3}$

L'opération Transfert Learning est effectuée en quatre étapes :

1. Récupérer les valeurs des poids w des différentes couches du modèle $inception_{v3}$ entrainé sur Imagenet et créer un nouveau réseau de neurones (couches complétement connectées).
2. Choisir une couche du modèle $inception_{v3}$ et initialiser le nouveau réseau avec les poids de celui-ci.
3. Alimenter le nouveau réseau de neurones avec les images de notre Data-set et lancer l'entraînement.
4. Mettre à jour uniquement les poids du nouveau réseau et continuer l'entraînement jusqu'à obtention du meilleur score possible.

La figure suivante illustre sommairement ce procédé.

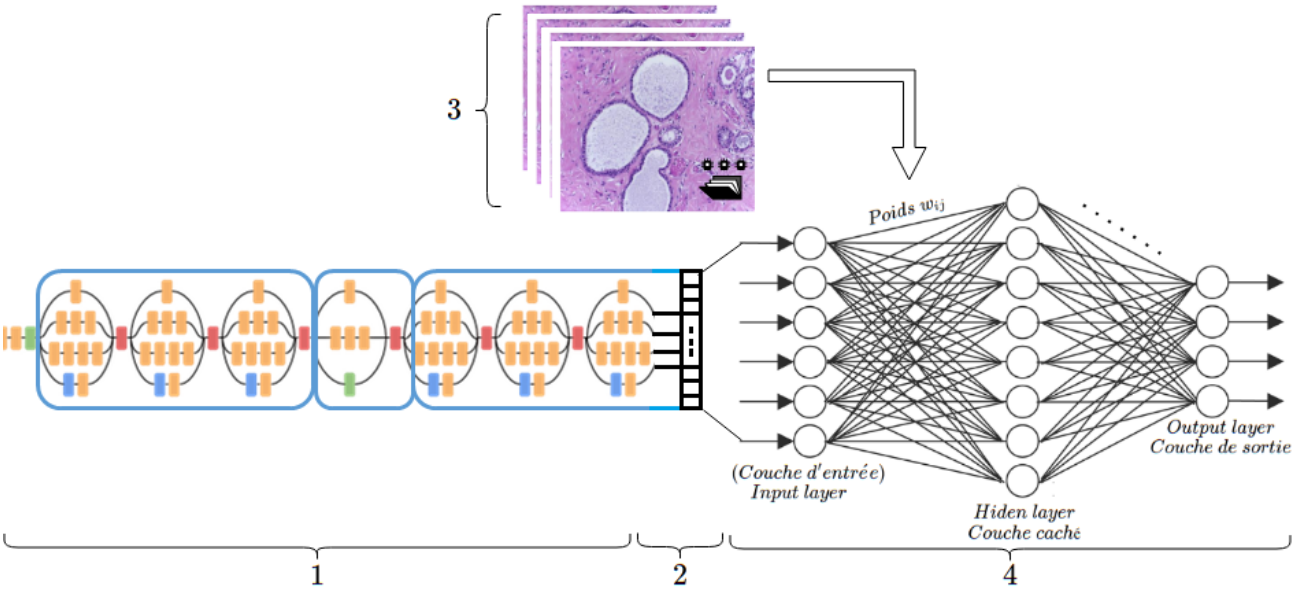


FIGURE 5.16 – Etapes générales d'un Transfert Learning

5.5.4 Détails d'architecture

	Modèle 1	Modèle 2
Nombre de couche	229	229
Taille du nouveau réseau	(17*17*768)*1024*4	(17*17*768)*1024*4
Nombre d'itérations	79	118
Neurones par couches	**30**	**108**
Répartition du Data-Set	**80% entraînement/ 20% valudation**	**70% / 30%**
Taux d'apprentissage α	Par défaut (0,001)	Par défaut (0,001)
Taille du batch	**32 entraînement** / **16** validation	**70 / 60**
Algorithme d'optimisation	Adam	Adam
Régularisation	non	non
Taille de l'image	$(300 * 300 * 3)$	$(300 * 300)$
Résulat (accuracy)	**92%**entranînement / **71%** validation	**100% / 77%**

5.5.5 Sommaire du modèle

```
_________________________________________________________________________________________________
Layer (type)                 Output Shape              Param #     Connected to
=================================================================================================
input_1 (InputLayer)         [(None, 300, 300, 3)      0
_________________________________________________________________________________________________
conv2d (Conv2D)              (None, 149, 149, 32)      864         input_1[0][0]
_________________________________________________________________________________________________
batch_normalization (BatchNorma (None, 149, 149, 32)   96          conv2d[0][0]
_________________________________________________________________________________________________
activation (Activation)      (None, 149, 149, 32)      0           batch_normalization[0][0]
_________________________________________________________________________________________________
conv2d_1 (Conv2D)            (None, 147, 147, 32)      9216        activation[0][0]
_________________________________________________________________________________________________
batch_normalization_1 (BatchNor (None, 147, 147, 32)   96          conv2d_1[0][0]
_________________________________________________________________________________________________
activation_1 (Activation)    (None, 147, 147, 32)      0           batch_normalization_1[0][0]
_________________________________________________________________________________________________
conv2d_2 (Conv2D)            (None, 147, 147, 64)      18432       activation_1[0][0]
_________________________________________________________________________________________________
flatten (Flatten)            (None, 221952)            0           mixed7[0][0]
_________________________________________________________________________________________________
dense (Dense)                (None, 1024)              227279872   flatten[0][0]
_________________________________________________________________________________________________
dense_1 (Dense)              (None, 4)                 4100        dense[0][0]
=================================================================================================
Total params: 236,259,236
Trainable params: 227,283,972
Non-trainable params: 8,975,264
_________________________________________________________________________________________________
```

FIGURE 5.17 – Sommaire de la quatrième solution (Transfert Learning)

5.5.6 Entraînement

```
Found 280 images belonging to 4 classes.
Found 120 images belonging to 4 classes.
start date and time :
2020-09-05 02:46:07
Epoch 1/300
4/4 [==============================] - 135s 34s/step - loss: 35.9246 - accuracy: 0.2500 - val_loss: 28.1783 - val_accuracy: 0.3250
Epoch 2/300
4/4 [==============================] - 112s 28s/step - loss: 29.1434 - accuracy: 0.3071 - val_loss: 20.3527 - val_accuracy: 0.3583
Epoch 3/300
4/4 [==============================] - 86s 21s/step - loss: 13.2953 - accuracy: 0.3786 - val_loss: 6.2967 - val_accuracy: 0.2583
Epoch 4/300
4/4 [==============================] - 56s 14s/step - loss: 5.2599 - accuracy: 0.2714 - val_loss: 1.8003 - val_accuracy: 0.4750
Epoch 5/300
4/4 [==============================] - 71s 18s/step - loss: 3.1890 - accuracy: 0.3429 - val_loss: 2.0320 - val_accuracy: 0.5500

    ......................
Epoch 104/300
4/4 [==============================] - 9s 2s/step - loss: 0.1188 - accuracy: 0.9500 - val_loss: 0.6995 - val_accuracy: 0.7833
Epoch 105/300
4/4 [==============================] - 9s 2s/step - loss: 0.0741 - accuracy: 0.9714 - val_loss: 0.7263 - val_accuracy: 0.7667
Epoch 106/300
4/4 [==============================] - 9s 2s/step - loss: 0.0587 - accuracy: 0.9893 - val_loss: 0.7469 - val_accuracy: 0.7583
Epoch 107/300
4/4 [==============================] - 9s 2s/step - loss: 0.0596 - accuracy: 0.9857 - val_loss: 0.7345 - val_accuracy: 0.7750
Epoch 108/300
4/4 [==============================] - ETA: 0s - loss: 0.0430 - accuracy: 1.0000
Reached 99% accuracy so cancelling training!
4/4 [==============================] - 9s 2s/step - loss: 0.0430 - accuracy: 1.0000 - val_loss: 0.7784 - val_accuracy: 0.7667
end date and time :
2020-09-05 03:37:04
```

FIGURE 5.18 – Entraînement de la quatrième solution (Transfert Learning)

5.5.7 Visualisation des résultats

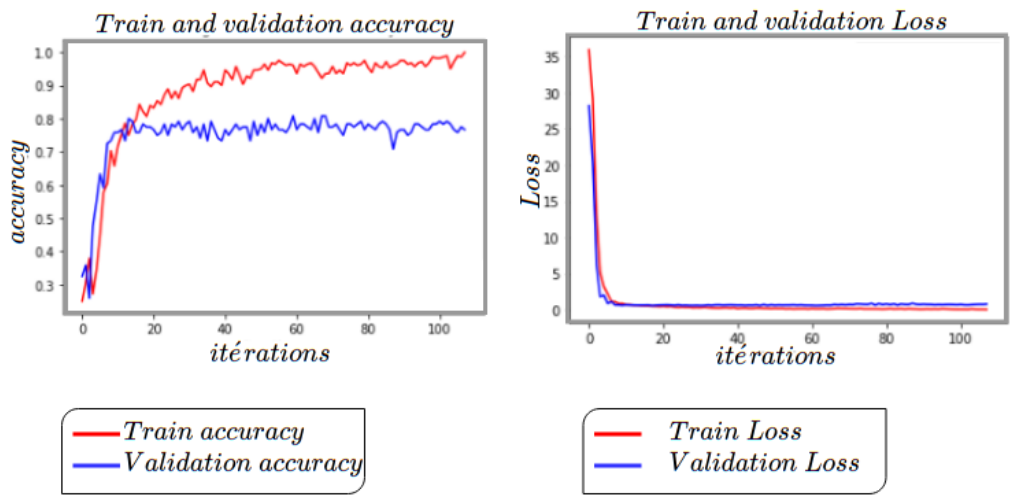


FIGURE 5.19 – Résultat la quatrième solution (Transfert Learning)

5.5.8 Analyse et critique de la méthode

Dans la plupart des cas, le Transfert Learning a permis d'atteindre un bon résultat au bout des premières itérations mais il n'assure pas la résolution du problème du sur-apprentissage de manière définitive, il est donc nécessaire d'appliquer d'autres méthodes de régularisation.

5.6 Cinquième solution : la régularisation

Toujours dans le but de réduire le problème du sur-apprentissage, nous avons combiné toutes les solutions précédentes (Data Augmentation et Transfert Learning) puis ajouté différentes techniques de régularisation : régularisation L_2, **Dropout**.

	Modèle 1	Modèle 2	Modèle 3	Modèle 4
Transfert learning	$Inception_{V3}$	$Resnet_{152-V2}$	$Inception - resnet_{V2}$	Xception
Régularisation	**Dropout(0.2)**	L_2 **(0,005)**	L_2 **(0,005)**	L_2 **(0,005)**
Nombre de couches	**229**	**505**	**727**	**120**
Nombre d'itérations	**50**	**420**	**308**	**270**
Nombre d'itérations	50	420	308	270
Répartition Data-set	**80% / 20%**	**70%/ 30%**	70%/ 30%	70%/ 30%
Taille du nouveau réseau	taille a	taille b	taille c	taille d
Taux d'apprentissage	Par défaut	Par défaut	Par défaut	Par défaut
Taille du batch	**32/16** (images)	**70/60**	70/60	70/60
Algorithme d'optimisation	Adam	Adam	Adam	Adam
Taille de l'image	300*300*3	**224*224*3**	300*300*3	**299*299*3**
Résultat	88%/78%	93%/80%	93%/82%	90%/86%

Remarques :
Dans le tableau ci-haut, les tailles du nouveau réseau pour les modèles sont :

- Taille a : (17*17*768)*1024*4
- Taille b : (14*14*1024)*128*4
- Taille c :(8*8*448)*128*4
- Taille d : (19*19*728)*128*4

La valeur du taux d'apprentissage α est initialisé par défaut à 0,001 en Tensorflow dans l'optimisation Adam[54].

5.6.1 Visualisation des résultats

5.6.2 Modèle 1

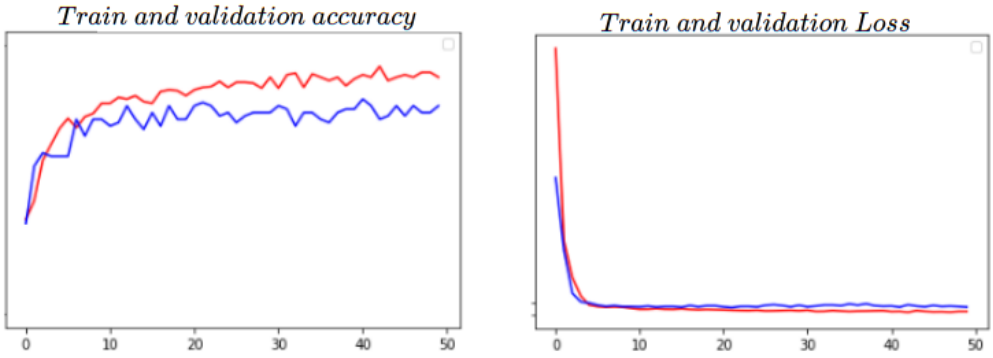


FIGURE 5.20 – Résultat du modèle 1 la cinquième solution (Régularisation)

5.6.3 Modèle 2

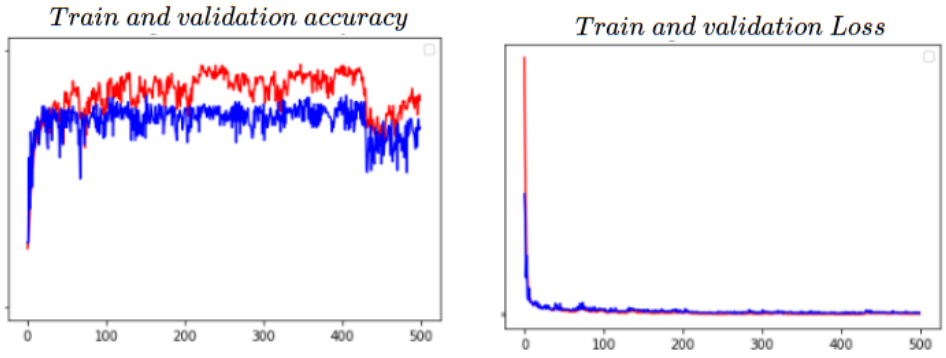


FIGURE 5.21 – Résultat du modèle 2 la cinquième solution (Régularisation)

5.6.4 Modèle 3

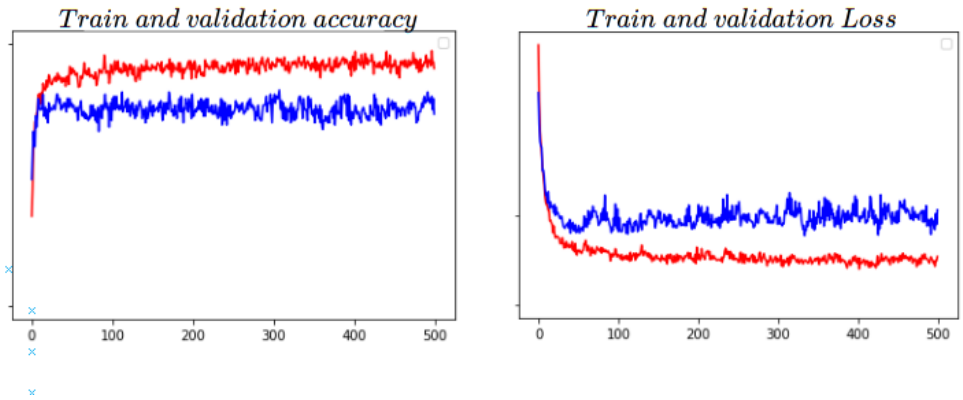


FIGURE 5.22 – Résultat du modèle 3 la cinquième solution (Régularisation)

5.6.5 Modèle 4

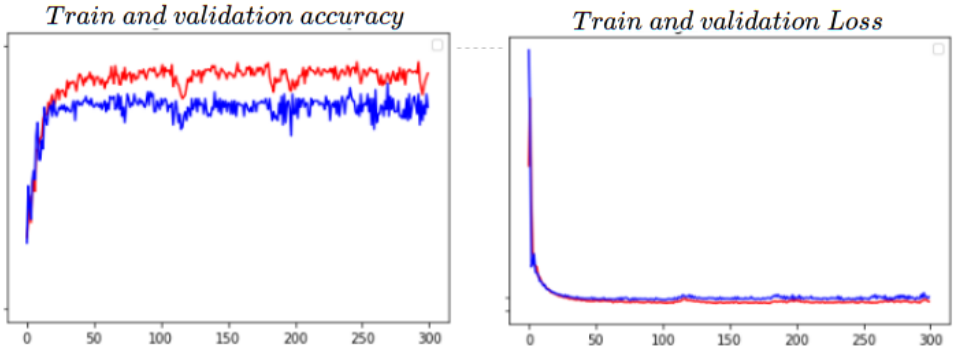


FIGURE 5.23 – Résultat du modèle 4 la cinquième solution (Régularisation)

5.7 Solution finale

5.7.1 Détails d'architecture

	Modèle 5
Transfer learning	**Nasnet-large**
Régularisation	regularisation L_2**(0.005)**
Nombre de couches	903
Nombre d'itérations	**571**
Répartition du Data-set	**70%**/ / **30%**
Taille du nouveau réseau	(11*11*432)*512*4
Taux d'apprentissage	Par défaut (0,001)
Taille du batch	**70** / **60**(images)
Algorithme d'optimisation	Adam
Régularisation	Non
Taille de l'image	331*331*3
Résultat (accuracy)	96% / 88%

5.7.2 Entraînement

```
start date and time :
2020-08-27 03:14:47
Epoch 1/600
4/4 [==============================] - 123s 31s/step - loss: 4522.7632 - accuracy: 0.2750 - val_loss: 6720.2607 - val_accuracy: 0.2500
Epoch 2/600
4/4 [==============================] - 111s 28s/step - loss: 4666.7886 - accuracy: 0.2107 - val_loss: 1843.5515 - val_accuracy: 0.2500
Epoch 3/600
4/4 [==============================] - 75s 19s/step - loss: 1774.1782 - accuracy: 0.2893 - val_loss: 1711.1575 - val_accuracy: 0.2500
Epoch 4/600
4/4 [==============================] - 47s 12s/step - loss: 1354.4069 - accuracy: 0.2500 - val loss: 474.5299 - val accuracy: 0.4083

                                    .....
4/4 [==============================] - 22s 5s/step - loss: 3.2087 - accuracy: 0.9536 - val_loss: 11.0564 - val_accuracy: 0.8417
Epoch 570/600
4/4 [==============================] - 22s 5s/step - loss: 3.1020 - accuracy: 0.9536 - val_loss: 9.9335 - val_accuracy: 0.8667
Epoch 571/600
4/4 [==============================] - ETA: 0s - loss: 3.4242 - accuracy: 0.9607
Reached 87% on validation accuracy so cancelling training!
4/4 [==============================] - 22s 5s/step - loss: 3.4242 - accuracy: 0.9607 - val_loss: 8.6633 - val_accuracy: 0.8750
end date and time :
2020-08-27 08:01:43
```

FIGURE 5.24 – Entraînement de la solution finale

5.8 Implémentation sur Tensorflow

5.8.1 Définition d'un réseau de neurones classique DNN

```
model = tf.keras.models.Sequential([
    tf.keras.layers.Flatten(input_shape=(300, 300,3)),
    tf.keras.layers.Dense(1024, activation='relu'),
    tf.keras.layers.Dense(512, activation='relu'),
    tf.keras.layers.Dense(128, activation='relu'),
    tf.keras.layers.Dense(4, activation=tf.nn.softmax)
])
```

FIGURE 5.25 – Implémentation DNN

5.8.2 Définition d'un réseau de neurones convolutif CNN

```
model = tf.keras.models.Sequential([
    tf.keras.layers.Conv2D(16, (3,3), activation='relu', input_shape=(300, 300, 3)),
    tf.keras.layers.MaxPooling2D(2, 2),
    # The second convolution
    tf.keras.layers.Conv2D(32, (3,3), activation='relu'),
    tf.keras.layers.MaxPooling2D(2,2),
    # The third convolution
    tf.keras.layers.Conv2D(64, (3,3), activation='relu'),
    tf.keras.layers.MaxPooling2D(2,2),
    tf.keras.layers.Flatten(),
    # 512 neuron hidden layer
    tf.keras.layers.Dense(512, activation='relu'),
    tf.keras.layers.Dense(4, activation=tf.nn.softmax)
])
```

FIGURE 5.26 – Implémentation CNN

5.8.3 Implémentation de transfert learning

```
# importer un modèle préentrainé
pre_trained_model = tf.keras.applications.NASNetLarge(
  include_top=False,
  weights='imagenet',
  input_shape=(331 ,331 ,3)
)
# désactiver le ré_entrainement de ce modèle
for layer in pre_trained_model.layers:
  layer.trainable = False
#récupérer les poids d'une couche
last_layer = pre_trained_model.get_layer('normal_concat_15')
last_output = last_layer.output

# redimentionner la sortie de la couche choisie en une dimention
x = layers.Flatten()(last_output)
# ajouter une couche complétement connecté avec 512 neuronnes et la foncion Relu
x = layers.Dense(512, activation='relu', kernel_regularizer=regularizers.l2(0.005))(x)
# Ajouter une dernière couche pour la classification multi_classes
x = layers.Dense  (4, activation='softmax')(x)

# définir le modèle
model = Model( pre_trained_model.input, x)
```

FIGURE 5.27 – Implémentation Transfert Learning

5.8.4 Augmentation de data augmentation

```
train_datagen = ImageDataGenerator(
  rescale=1./255,
  rotation_range=40,
  width_shift_range=0.2,
  height_shift_range=0.2,
  shear_range=0.2,
  zoom_range=0.2,
  horizontal_flip=True,
  fill_mode='nearest')
```

FIGURE 5.28 – Data augmentation

5.8.5 Compilation du modèle

```
model.compile(optimizer='adam',
              loss='categorical_crossentropy',
              metrics=['accuracy'])
```

FIGURE 5.29 – Implémentation de la compilation du modèle

5.8.6 Définition du chemin vers les Data-sets et la taille des batchs

```
validation_datagen = ImageDataGenerator(rescale=1./255)

# importer les images d'entrainement par batch de 70 images
train_generator = train_datagen.flow_from_directory(
    '/tmp/breast_cancer/training/',
    target_size=(331 ,331),
    batch_size=70,
    class_mode='categorical'
)
# importer les images de validation par batch de 60 images
validation_generator = validation_datagen.flow_from_directory(
    '/tmp/breast_cancer/testing/',
    target_size=(331 ,331),
    batch_size=60,
    class_mode='categorical')
```

FIGURE 5.30 – Implémentation du chemin des données

5.8.7 Entraînement du modèle

```
history = model.fit(
    train_generator,
    steps_per_epoch=4,
    epochs=1000,
    verbose=1,
    validation_data = validation_generator,
    validation_steps=2,
    callbacks=[callbacks]
)
```

FIGURE 5.31 – Entraînement de la solution finale

5.8.8 Visualisation des résultats

```
# ================================================================
# parcourir l'historique et afficher les graphs de  l'évolutions
# des scores et de l'erreur au cours de l'entrainement
# ================================================================

# PLOT LOSS AND ACCURACY
%matplotlib inline
import matplotlib.image  as mpimg
import matplotlib.pyplot as plt

acc=history['accuracy']
val_acc=history['val_accuracy']
loss=history['loss']
val_loss=history['val_loss']

epochs=range(len(acc)) # Get number of epochs
# Plot training and validation accuracy per epoch
#------------------------------------------------
plt.plot(epochs, acc, 'r', label="Training Accuracy")
plt.plot(epochs, val_acc, 'b', label="Validation Accuracy")
plt.title('Training and validation accuracy')
plt.xlabel("Iterations")
plt.ylabel("Accuracy")
plt.legend(loc='lower right')
plt.figure()
# Plot training and validation loss per epoch
#------------------------------------------------
plt.plot(epochs, loss, 'r', label="Training Loss")
plt.plot(epochs, val_loss, 'b', label="Validation Loss")
plt.title('Training and validation loss')
plt.xlabel("Iterations")
plt.ylabel("Loss")
plt.legend(loc='upper right')
plt.show()
```

FIGURE 5.32 – Résultat de la solution finale

5.8.9 Teste d'un modèle avec des images

FIGURE 5.33 – Implémentation de test

5.8.10 Interface web

Pour mettre notre modèle à la disposition des professionnels de la santé, nous avons développé une interface web simple permettant de charger des images microscopiques du cancer du sein, l'application permettra ensuite de prédire quel est le type de ce cancer.
Remarque : Le résultat de la prédiction ne sera pas nécessairement un cancer, la classe « Normal » permet d'orienter le spécialiste vers une pathologie fortement ressemblante et qui porte à confusion.

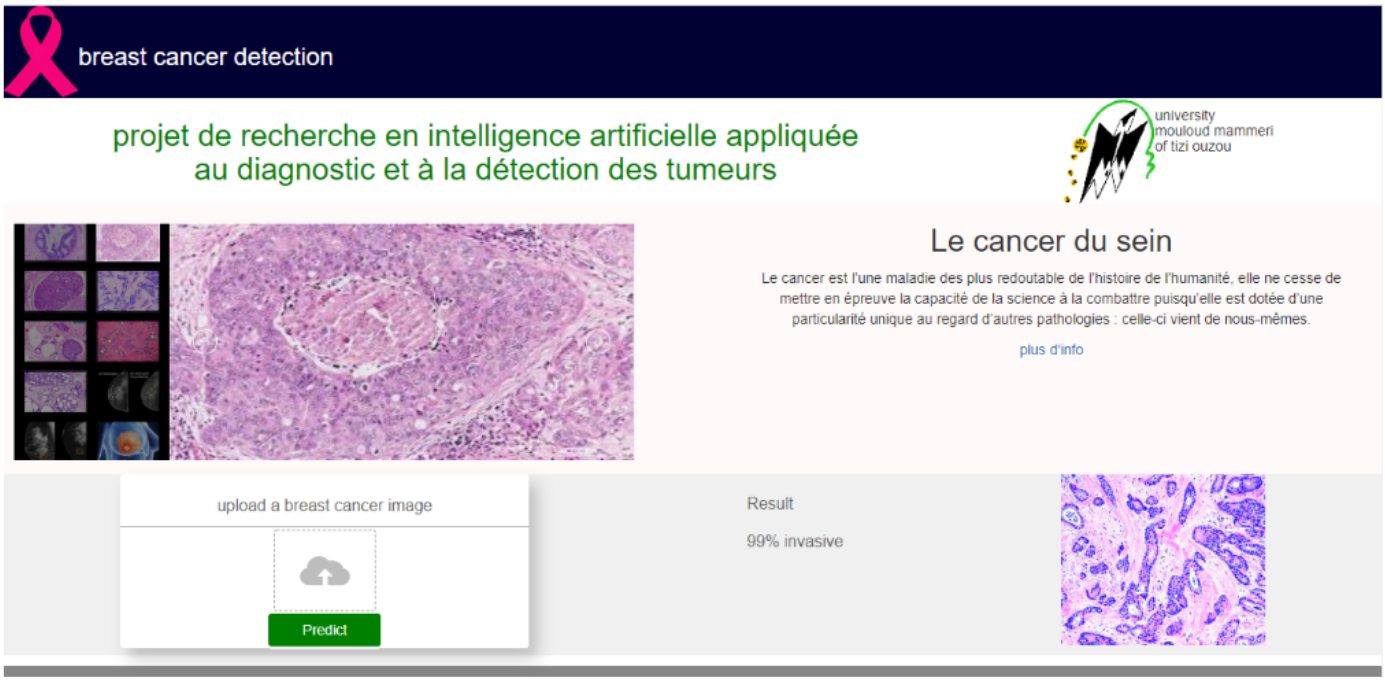


FIGURE 5.34 – Interface web

5.9 Autres outils utilisés

5.9.1 Le langage Python

Python est le langage de programmation le plus utilisé en ML depuis ces dernières années avec le langage R. Les raisons ayant motivé le choix de python sont les suivantes :

— Disponibilité d'un grand nombre de bibliothèques dédiées pour le ML.

— Simplicité et fluidité de syntaxe.

— Une large communauté active sur **Github**.

Remarque : Le nombre de ces bibliothèques est énorme, ce pourquoi il n'est pas toujours facile de les nommer et citer leurs fonctionnalités, nous citerons donc les plus importantes dans notre travail : matplotlib, Keras, numpy, tensorflow.

5.9.2 Numpy

Numpy est un paquet de traitement de tableaux à usage général. Il fournit des objets de tableaux multidimensionnels de haute performance et des outils pour travailler avec ces tableaux. Numpy est un conteneur efficace de données multidimensionnelles génériques. Il facilite les opérations mathématiques sur les tableaux et leur vectorisation. Cela améliore considérablement les performances et accélère le temps d'exécution.

5.9.3 Matplotib

Matplotlib est la bibliothèque de traçage pour Python qui fournit une API orientée objet pour intégrer des tracés. Grâce à Matplotlib, on peut tracer des histogrammes, graphes de fonctions..etc.
Dans notre cas matplotlib nous a permis de tracer les courbes d'apprentissage et de la fonction *Loss* pour visualiser l'avancement de nos modèles.

5.9.4 Tensorflow, Keras

Tensorflow est une bibliothèque d'intelligence artificielle qui aide les développeurs à créer des réseaux de neurones à grande échelle avec de nombreuses couches en utilisant des graphiques de flux de données.
Tensorflow facilite également la création de modèles Deep Learning, pousse l'état de l'art en ML/IA et permet un déploiement facile des applications alimentées en ML. Il est utilisé par Google, Coca-Cola, Twitter, Intel et autres.
On retrouve cette bibliothèque également en reconnaissance faciale voix et son, analyse de sentiments, Google Translate, Recommandation d'Amazon de Google et de Netflix et beaucoup d'autres domaines d'application de l'IA.

Keras est une bibliothèque de réseaux de neurones open-source en Python. Avec Keras, la modélisation statistique, le travail avec les images et le texte est beaucoup plus facile.

Dans notre cas Keras nous a permis de :

- Déterminer le pourcentage de précision.
- Définir la fonction de perte.
- Créer des couches de fonctions personnalisées.
- Traitement intégré des données et des images.
- Fonctions d'écriture avec blocs de code répétitifs : 20, 50, 100 couches de profondeur.

Conclusion

Nous avons présenté dans ce chapitre les modèles, librairies et configurations utilisés pour notre implémentation ainsi que les ensembles (entraînement,validation,test) que nous avons utilisé et leurs évaluations respectives.
La comparaison de nos résultats a confirmé empiriquement que les facteurs importants pour l'obtention de meilleurs résultats sont les suivants : le modèle utilisé (DNN,CNN), le nombre d'itérations (Early stopping), la taille de la base d'apprentissage/validation/test, la nature des données, la profondeur des réseaux , le choix d'une descente de gradient et taux d'apprentissage (Adam) , la gestion des poids (régulatisation L2 et Drop-out, Transfer-learning).
Un score de 88% est un résultat optimiste pour une classification de cancers toutefois, les approches que nous avons proposées sont à améliorer pour une généralisation à très grande échelle.

Conclusion générale

Synthèse

La classification automatique de cancers du sein vise à apporter des réponses pertinentes aux contraintes de ce domaine de la médecine. En effet, pour y répondre, nos travaux se sont ainsi orientés vers la prise en compte de notions détaillées des réseaux de neurones afin de viser directement des méthodes éfficaces.
Dans ce travail, nous avons présenté d'une façon globale le cancer du sein et l'apprentissage automatique, plus particulièrement les réseaux de neurones. Nous avons également présenté les architectures choisis qui répondent éventuellement à notre problématique notament en puissance de calculs, structure de données et choix des modèles. Les résultats obtenus démontrent un intérêt certain pour les méthodes que nous avons adopté pour réaliser notre système de classification néaumoins ; afin d'atteindre un niveau de sûreté plus satisfaisant, celui-ci nécessite une éventuelle amélioration car pour la sécurité des patients, ces algorithmes doivent être testés avec une multitude de paramètres, afin que l'on comprenne leurs forces et faiblesses.

Perspectives

Les perspectives découlant de nos travaux sont nombreuses, les plus importantes sont les suivantes.

- Créer notre propre architecture de réseau de neurones et l'entraîner avec des images brutes (sans compression et changement de dimension) sur un serveur puissant.
- Explorer d'autres paradigmes de l'apprentissage (SVM et autres), et cela afin d'avoir une meilleure compréhension et distinction des résultats.
- Utiliser d'autres types d'imageries médicales hormis les histologies et comparer les résultats, cela a été proposé pour prochaine compétition (2020) utilisant des images issues de l'immunochimie.
- Créer une application web ou mobile et la mettre à la disposition du personnel de la santé à fin d'enrichir le Data-set avec de nouvelles images et améliorer la précision de la prédiction.

Bibliographie

[1] Sofiane Akkouche, *Cancer du sein : Les inquiétantes statistiques des spécialistes* https://www.algerie360.com/, consulté le 20/04/2020

[2] https://iciar2018-challenge.grand-challenge.org/Home/?fbclid=IwAR2h6ao0eyGoGFwF1tIHy4B740tlRITh-Sqeb_8NrXfj75WFOm0PnZakUu0 consulté le 30/01/2020.

[3] Christophe Ginestier, Hasan Korkaya, Gabriela Dontu, Daniel Birnbaum, Max S. Wicha, Emmanuelle Charafe-Jauffret, La cellule souche cancéreuse, Un pilote aux commandes du cancer du sein, 2007

[4] https://www.cancer.ca/fr-ca/cancer-information/cancer-type/breast/breast-\tolerance9999\emergencystretch3em\hfuzz.5\p@\vfuzz\hfuzzcancer/breast-cancer-in-men/. consulté le 19/05/2020

[5] https://lemondeetnous.cafe-sciences.org/2014/07/comprendre-le-cancer-du-sein-partie-i/, consulté le 20/05/2020.

[6] https://www.cancer.ca/fr-ca/cancer-information/cancer-101/what-is-cancer/how-cancer-starts-grows-and-spreads/?region=on, consulté le 22/05/2020.

[7] traitements des cancers du sein, collection Guides patients Cancer info, INCa, octobre 2013,

[8] Elston C, Ellis I, Histopathology, 1991

[9] National Cancer Institute, Breast Cancer Treatment (Adult) (PDQ®)–Patient Version, november 21 201

[10] Biji Babu MDRDa, Bhawna Dev MD, DNBa, T. Mohanapriya MSb, *Bilateral mammary Paget disease in a young adult female,* 2018

[11] Olivier Hermine, *Cancer : le diagnostic*, https://www.fondation-arc.org/cancer-le-diagnostic, consulté le 29/0/2020 dictionnaire médicale flammarion 8 ème édition

[12] Olivier Hermine, *Cancer : le diagnostic*, https://www.fondation-arc.org/cancer-le-diagnostic, consulté le 29/0/2020 dictionnaire médicale flammarion 8 ème édition

[13] *Dictionnaire de médecine Flammarion*, 8e édition,4 novembre 2000

[14] Collège Français des Pathologistes (CoPath) *Généralités sur les tumeurs*, 2011/2012

[15] Jean-Michel André, Martin Catala, Jean-Jacques Morère. *Histologie : les tissus*, 2007 - 2008.

[16] Dr Jean-François Delaloye,Pr Hans-Anton Lehr *Faut-il opérer toutes les lésions prémalignes du sein ?*, 2007 - 2008.

[17] Jason Wasserman, *https ://www.mypathologyreport.ca/carcinome-canalaire-in-situ/*, 2007 - 2008.

[18] `https://www.bosnianpathology.org/bosnianpathology.org/teaching-cases/93-breast-pathology/invasive-carcinoma-nst/141-case-7-invasive-ductal-carcinoma-nst-high-grade-with-basal-\tolerance9999\emergencystretch3em\hfuzz.5\p@\vfuzz\hfuzzlike-phenotype` consulté le 01/01/2020.

[19] Keum Won Kim, Kyu Ran Cho1,Bo Kyoung Seo1, *Sonographic Findings of Mammary Duct Ectasia : Can Malignancy be Differentiated from Benign Disease*, 2010

[20] J. Lansac *La maladie fibrokystique des seins*, `http://umvf.cerimes.fr/media/ressWikinu/Gynecologie/College/Lansac-Maladie_fibrokytiques_sein.pdf` consulté le 19/03/2020

[21] Lise Verbeke, `https://www.cancer.ca/fr-ca/cancer-information/cancer-type/breast/breast-cancer/breast-cancer-in-men/`. consulté le 19/05/2020

[22] Chigozie Enyinna Nwankpa, Winifred Ijomah, Anthony Gachagan, and Stephen Marshall. *Activation Functions : Comparison of Trends in*, Nov 2018

[23] R. Sathya,Annamma Abraham. *Comparison of Supervised and Unsupervised Learning Algorithms for Pattern Classification*, (IJARAI) International Journal of Advanced Research in Artificial Intelligence,Vol. 2, No. 2, 2013

[24] Matthieu Zimmer, *Apprentissage par renforcement développemental. Intelligence artificielle [cs.AI]*, Université de Lorraine, 2018

[25] Joannes Vermorel, *Probabilité fréquentiste vs. probabilité bayésienne*, `https://www.lokad.com/fr/definition-entropie-croisee`. consulté le 6/06/2020.

[26] Oliver Sutton. *Introduction to k Nearest Neighbour Classification and Condensed Nearest Neighbour Data Reduction* February, 2012.

[27] F.Rosenblatt, *The Perceptron — a perceiving and recognizing automaton. Report 85–460–1.* 1957

[28] Claude Touzet. *les réseaux de neurones artificiels, introduction au connexionisme : cours, exercices et travaux pratiques.* EC2, 1992, Collection de l'EERIE, N. Giambiasi. ffhal-01338010f

[29] Leonard Blier,Pierre Wolinsk, Yann Ollivie. *Learning with Random Learning Rates*

[30] Laurence Broze et Guy Mélard, Université de Lille et Université Libre de Bruxelles. *Lissage exponentiel généralisé.* Octobre 1995.

[31] Par Gilles Mairet — Travail personnel, CC BY-SA 4.0. `https://commons.wikimedia.org/w/index.php?curid=65074666` consulté le 15/07/2020.

[32] Donald Olding HEBB. *The Organization of Behavior* New York, Wiley and Sons, 1949.

[33] Hugo Larochelle *Apprentissage automatique.* Département d'informatique, Université de Sherbrooken,2013.

[34] Rob Schapire,Frank Xiao *Theoretical Machine Learning.* April 11, 2013.

[35] Elad Hazan, *Optimization for machine learning.* https://simons.berkeley.edu/sites/default/files/docs/5914/berkeley-tutorial-part1.pdf. Princeton University, consulté le 02/07/2020

Jianli Feng, Shengnan Lu,Performance Analysis of Various Activation Functions in Artificial Neural Networks ,ICSP 2019

[36] Ricco Rakotomalala,Université Lumière Lyon *Principe de la descente de gradient pour l'apprentissage supervisé Application à la régression linéaire et la régression logistique* http://tutoriels-data-mining.blogspot.com/. Consulté le 10/07/2020.

[37] Aston Zhang, Zachary C. Lipton, Mu Li, and Alexander J. Smola. *Dive into Deep Learning.* Aug 11, 2020 Release 0.14.3

[38] Jimmy Lei Ba,Diederik P. Kingma *Adam : a method for stochastic optimization.* Publié en tant que document de conférence à l'ICLR 2015

[39] Hugo Larochelle *Apprentissage automatique.* Département d'informatiqu, Université de Sherbrooken,2013.

[40] Jianli Feng, Shengnan Lu. *Performance Analysis of Various Activation Functions in Artificial Neural Networks.* ICSP 2019.

[41] Rachel Draelos *Connections : Log Likelihood, Cross Entropy, KL Divergence, Logistic Regression, and Neural Networks.* https://glassboxmedicine.com/2019/12/07. consulté le 28/04/2020

[42] Claude E. Shannon, Warren Weaver, *The Mathematical Ttheory of communication.* 1964.

[43] Jason Brownlee *Why One-Hot Encode Data in Machine Learning* https://machinelearningmastery.com/why-one-hot-encode-data-in-machine-learning/. consulté le 15/06/2020.

[44] Aston Zhang, Zachary C. Lipton, Mu Li, and Alexander J. Smola. *Dive into Deep Learning.* Aug 11, 2020 Release 0.14.3

[45] Nitish Srivastava,Geoffrey Hinton,Alex Krizhevsky,Ilya SutskeverRuslan Salakhutdinoe. *Dropout : A Simple Way to Prevent Neural Networks from Overfiting.* juin 2014

[46] XU ZongBen1, ZHANG Hai1,2, WANG Yao1, CHANG XiangYu1,LIANG Yong3. $L_{1/2}$ *regularization.* juin 2014

[47] Kaiming He,Xiangyu Zhang, Shaoqing Ren, Jian Sun *Deep Residual Learning for Image Recognition.* 2015

[48] https://www.maxicours.com/se/cours/aires-visuelles-et-perception\tolerance9999\emergencystretch3em\hfuzz.5\p@\vfuzz\hfuzz-visuelle/. consulté le 30/04/2020

[49] Nguyên HOANG, *La formule du savoir*, juin 2018.

[50] Aston Zhang, Zachary C. Lipton, Mu Li, and Alexander J. Smola. *Dive into Deep Learning.* Aug 11, 2020 Release 0.14.3

[51] C.-C. Jay Kuo,University of Southern California, Los Angeles *Understanding Convolutional Neural Networks with A Mathematical Model.* 02/09/2016

[52] https://silhouetteofscience.com/diagrams-and-deep-neural-nets-\tolerance9999\emergencystretch3em\hfuzz.5\p@\vfuzz\hfuzzabstraction-in-science/ consulté le 27/06/2020.

[53] https://www.tensorflow.org/?hl=fr, consulté le 10/01/2020

[54] https://www.tensorflow.org/api_docs/python/tf/keras/optimizers/Adam, consulté le 10/09/2020.

[55] https://www.calyps.ch/analyse/data-face-data-augmentation/ consulté le 03/05/2020

[56] https://tel.archives-ouvertes.fr/tel-02065405 consulté le 19/08/2020

[57] Ying Lu,Université de Lyon *Transfer Learning for Image Classification* 2017

[58] Christian Szegedy,Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, Zbigniew Wojna, *Rethinking the Inception Architecture for Computer Vision.* 11 Decembre 2015.

[59] https://cloud.google.com/tpu/docs/inception-v3-advanced?hl=fr consulté le 01/08/2020.

[60] https://le-datascientist.fr/top-10-des-bibliotheques-python\tolerance9999\emergencystretch3em\hfuzz.5\p@\vfuzz\hfuzz-pour-la-data-science consulté le 01/08/2020.