Mémoire de projet de recherche

Interprétation des réseaux de neurones profonds

- Encadré par :

  Mme. ZAHRAOUI Farida

- Réalisé par :

  EL ROBRINI Medina : medina.elrobrini96@gmail.com

  RAMOUL Rayan Samy : raysamram@gmail.com

23/04/2020

Table des figures

Liste des tableaux

Introduction

Contexte

L'avancée de l'intelligence artificielle a permis de révolutionner le monde de la médecine de précision. Cette dernière également appelée médecine personnalisée, se focalise sur l'analyse des caractéristiques moléculaires et génomiques pour proposer un traitement adapté au patient.
Dans le but de mieux comprendre la maladie de la leucémie, des données sous forme d'expression de gènes sont utilisées afin d'essayer de trouver des sous-classes de cette maladie. Pour ce faire, nous utilisons l'apprentissage profond qui continue à faire ses preuves dans tout ce qui est traitement automatique du langage naturel, la bio-informatique, etc. Il serait donc intéressant de voir de plus près l'application du *deep learning* à la médecine de prédiction.

Problématique et objectifs

Le développement des modèles de *Deep learning* a permis d'améliorer grandement la performance des prédictions. Néanmoins, ces modèles soulèvent de nombreuses questions quant à leur interprétabilité. En effet, la précision des modèles ne suffit plus, aujourd'hui, la capacité à expliquer la prise de décision de ces algorithmes est tout aussi importante. C'est même une exigence minimale pour certains processus automatisés. Cette capacité à expliquer les modèles permet d'améliorer la compréhension des résultats mais aussi d'apporter de la crédibilité surtout dans certains domaines de la médecine où la rigueur et la précision sont de mises.

Pour répondre à ce besoin d'interprétation, nous utilisons une bibliothèque nommée *iNNvestigate* qui contient plusieurs méthodes d'analyse nous permettant de bien comprendre notre modèle. Cependant, cette bibliothèque a été conçu pour s'adapter uniquement aux entrées sous forme d'images.

Chapitre 1

Etat de l'art

1.1 Généralités

L'apprentissage automatique permet à un système d'apprendre à partir des données et non à l'aide d'une programmation explicite. Il en résulte un modèle*. Il y a plusieurs types d'apprentissage :

- **L'apprentissage supervisé :** l'apprentissage se fait à l'aide des données d'entrée et de sortie étiquetées. Le but étant de trouver une fonction d'approximation qui lie les entrées aux sorties. On distingue 2 types d'apprentissage supervisé majoritaires :
  — La régression* : Ayant pour but de prédire une donnée quantitative continue.
  — La classification : Où le modèle doit prédire une donnée discrète.
- **L'apprentissage non supervisé (Clustering) :** les données sont non étiquetées, de sorte que l'algorithme d'apprentissage puisse déterminer à lui seul des points communs parmis ses données d'entrée.
- **L'apprentissage semi-supervisé :** utilise un ensemble de données étiquetées et non étiquetées. Ce type d'apprentissage est assez intéressant étant donné que l'étiquetage des données peut s'avérer difficile.
- **L'apprentissage par renforcement :** se base sur un cycle d'expérience / récompense et améliore les performances à chaque itération, il a comme particularité de pouvoir être utilisé sans appel à des données humaines.

Il existe plusieurs méthodologies d'apprentissage automatique*, on cite : les réseaux de neurones, les arbres de décisions, les machines à support de vecteurs, etc.
Dans le cadre de notre étude, nous sommes dans le cas d'un apprentissage supervisé et nous nous intéressons particulièrement à la méthode des réseaux de neurones.

1.1.1 Les réseaux de neurones

Les réseaux de neurones ont été développés comme un modèle mathématique générique afin de modéliser les neurones biologiques. Ils comportent un certain nombre d'éléments de traitement d'information appelés neurones [5].

Chaque neurone reçoit des entrées et fournit une sortie, grâce à différentes caractéristiques :
— Des poids accordés à chacune des entrées, permettant d'en modifier l'importance de certaines par rapport aux autres. Ce sont des paramètres à estimer lors de la procédure d'apprentissage.

— Une fonction d'agrégation, qui permet de calculer une unique valeur à partir du produit scalaire du vecteur des entrées et des poids correspondants.
— Un seuil (ou biais) θ.
— Une fonction d'activation, qui associe à chaque valeur agrégée une unique valeur de sortie dépendant du seuil [6]. La sortie du neurone aura la formule 1.1

$$y = f(\sum_{i=1}^{n}(x_i * w_i) + \theta) \tag{1.1}$$

Notons que la fonction f représente la fonction d'activation et la somme en argument représente la fonction d'agrégation décrite plus haut. On distingue plusieurs types de fonction d'activation, parmi elles :

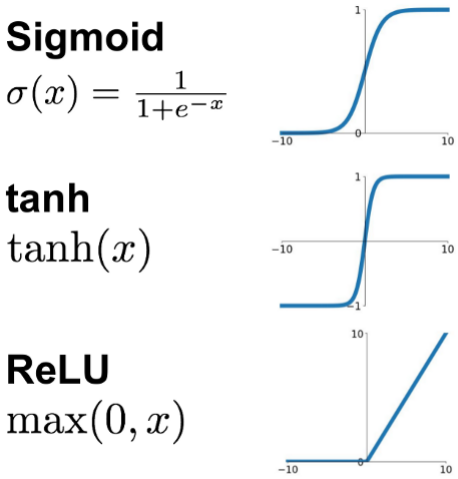


FIGURE 1.1: Fonctions d'activation

Ces différents éléments représentant un neurone sont illustrés dans la figure 1.2

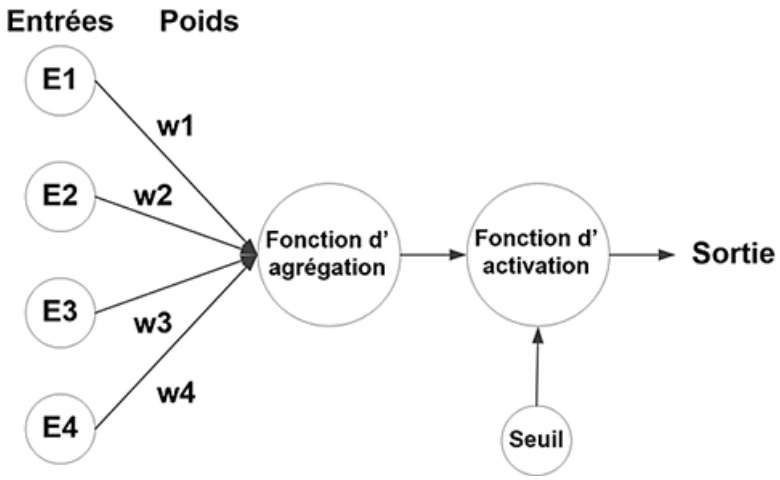


FIGURE 1.2: Neurone artificiel

Cependant, pour apprendre des fonctions plus complexes et pallier au problème des données qui ne sont pas toujours linéairement séparables, le perceptron multicouche ou Multi Layer Perceptron (MLP) est apparu. Ce dernier n'est autre qu'un ensemble de couches contenant des neurones artificiels comme le montre la figure 1.3

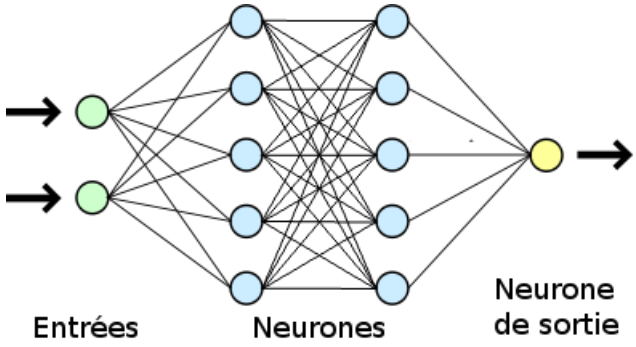


FIGURE 1.3: Perceptron multicouche

Nous remarquons une couche d'entrée contenant autant de neurones qu'il y a de variables dans le problème qu'on veut résoudre, des couches cachées et une couche de sortie qui fournit la réponse du système. Chaque neurone d'une couche cachée est connecté en entrée à chacun des neurones de la couche précédente et en sortie à chaque neurone de la couche suivante. Il est à noter que les fonctions d'activation peuvent être différentes d'une couche à une autre mais les neurones d'une couche doivent tous avoir la même fonction d'activation.

L'objectif des réseaux de neurones est d'ajuster les poids de telle sorte à minimiser la fonction d'erreur (*Cost function*)*. Il existe plusieurs types de fonctions d'erreurs comme par exemple l'erreur quadratique donnée par la formule 1.2

$$\frac{1}{n}\sum_{i=1}^{n}(prediction_i - y_i)^2 \tag{1.2}$$

- $prediction_i$: la prédiction du modèle
- y_i : la véritable sortie du modèle

Ainsi, le problème revient donc à trouver le minimum de cette fonction d'erreur. Au départ, les poids sont générés de manière aléatoire. Ils sont ensuite ajustés au fur et à mesure grâce à la descente du gradient. La modification des poids est propagée de la couche de sortie jusqu'à la couche d'entrée et c'est ce qu'on appelle la rétro-propagation du gradient. Il en résulte *un modèle* caractérisé par une matrice de poids et de biais.

Surapprentissage et Dropout

On parle de surapprentissage (overfitting) quand un modèle a trop appris les particularités de chacune des entrées fournies pour son entrainement. Il présente alors un taux de succès très important sur les données d'entraînement, au détriment de ses performances générales réelles.

Pour limiter ce genre de problèmes dans le cas des réseaux de neurones, on doit veiller à utiliser un nombre adéquat de neurones et de couches cachées, on peut donc utiliser la technique du dropout.

Le dropout est une technique qui est destinée à empêcher le surapprentissage en désactivant des unités dans un réseau de neurones. En pratique, les neurones sont soit désactivés avec une probabilité p ou gardés avec une probabilité de $1 - p$.

1.1.1.1 Les limitations des réseaux de neurones

Bien que l'apprentissage automatique a permis d'apprendre plusieurs concepts il reste limité quand il s'agit des problèmes plus complexes tel que la reconnaissance des images. Le passage à l'apprentissage profond a eu lieu suite aux limitations suivantes :

- Les réseaux de neurones ont un temps d'execution qui peut s'avérer très long.
- Les réseaux de neurones ont tendance à mal généraliser provoquant ainsi un surapprentissage. Pour empêcher cela, si le dropout n'est pas appliqué par exemple, il y aura une saturation de l'activation des neurones.
- L'apprentissage automatique nécessite des données structurées tandis que le système du deep learning est capable d'identifier lui-même les caractéristiques discriminantes. Dans l'apprentissage profond*, au niveau de chaque couche, un nouveau critère spécifique de l'objet est sélectionné servant de base pour décider de la classification retenue pour l'objet à la fin du processus.
- L'apprentissage automatique utilise peu de couches contrairement à l'apprentissage profond qui à travers ses multiples couches cachées permet d'apprendre plusieurs niveaux d'abstraction des caractéristiques surtout pour des applications qui nécessitent plusieurs filtres tels que de la reconnaissance d'images par exemple.
- Le manque d'interprétabilité des réseaux de neurones induit des restrictions quant à leur utilisation pratique. En effet, une bonne performance prédictive ne suffit pas mais doit s'accompagner d'explications de l'algorithme.

L'apprentissage profond a besoin d'une grande masse de données et donc nécessite un temps de calcul important, c'est pourquoi la vraie révolution est liée aux technologies actuelles notamment la parallélisation des calculs comme le montre l'exemple de Nvidia Cuda* et la quantité de données accessibles.

1.1.2 Les réseaux de neurones profonds

Les réseaux de neurones profonds sont des réseaux de neurones qui contiennent au moins deux couches cachées, leur architecture est donc plus complexe que le perceptron multicouche vu plus haut. Ils permettent d'apprendre plusieurs niveaux d'abstraction des caractéristiques.

Le passage aux réseaux de neurones profonds s'est révelé bien utile pour la résolution des problèmes complexes nécessitant un grand volume de données. Les avancées technologiques des ordinateurs niveau hardware notamment les GPU* (Graphics Processing Units) ont fourni une meilleure puissance de calcul conséquente et suffisante à l'apprentissage de ce type de modèle. Des processeurs spécifiques ont alors été développés, adaptés aux différentes phases de calcul des algorithmes.

Trois grandes familles de réseaux d'apprentissage profond ont été développés on distingue les réseaux de neurones convolutifs* (CNN) pour l'analyse d'images, les modèles séquentiels* (LSTM par exemple) utiles lorsqu'il y a une dimension temporelle ainsi que les auto-encodeur.

1.2 Intérpretation des réseaux de neurones profonds

1.2.1 Défintion

L'interprétabilité est la capacité d'expliquer ou de présenter des informations dans des termes humainement compréhensibles. Par exemple, cela peut se faire avec une représentation visuelle de l'importance des variables.

Aujourd'hui, l'explicabilité d'un modèle est devenue nécessaire au même titre que la performance. On peut distinguer trois niveaux d'interprétabilité dans les algorithmes d'apprentissage automatique [9] :

- **Haute interprétabilité** : ce niveau inclut les algorithmes de régression, les arbres de décision et les règles de classification traditionnelle.
- **Interprétabilité moyenne** : ce niveau inclut des algorithmes tels que les modèles graphiques.
- **Faible interprétabilité** : ce niveau inclut des techniques avancées d'apprentissage automatique telles que les SVM *, les méthodes d'ensembles et l'apprentissage profond. Au mieux, ils fournissent des informations sur l'importance des variables pour l'explicabilité du modèle.

Ces différents algorithmes d'apprentissage automatique selon leur degré d'interprétabilité sont présentés dans la figure 1.4

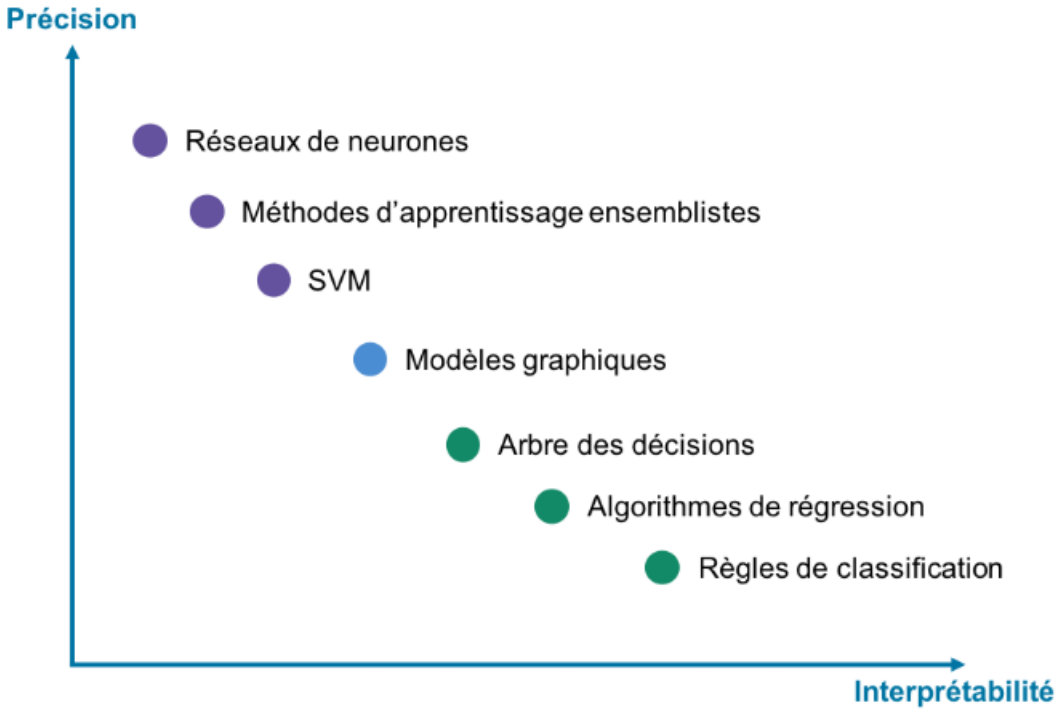


FIGURE 1.4: Précision des algorithmes en fonction de leur l'intepretabilité

Nous pouvons remarquer que les réseaux de neurones, utilisés dans notre travail, ont une très faible interprétabilité par rapport aux autres algorithmes.

Selon Chakraborty et al [10], Il existe plusieurs aspects de l'interprétabilité :

1.2.2 Interprétation des modèles par leur transparence (*Model transparency*)

L'étude de la transparence des modèles s'avère si importante qu'elle ne se limite pas qu'à expliquer les modèles mais peut aussi nous aider à les améliorer. Ce type d'interprétation peut être défini par les propriétés suivantes :

- Simultanéité : (Simultability) : la capacité de l'être humain à reproduire les mêmes étapes utilisées par le modèle et prédire la même sortie. Cela prouve que les changements effectués au niveau du modèle ont bien été compris par l'humain ce qui le rend interprétable.
- Décomposition (Decomposability) : Le fait de pouvoir justifier le choix des paramètres du model.
- Transparence de l'algorithme (Algorithmic transparency) : Le fait de pouvoir expliquer de manière simple et précise l'algorithme d'apprentissage.

1.2.3 Interprétation des modèles par leur fonctionnement (*Model functionality*)

Ce type d'inteprétation peut se faire par des moyens de description de modèles compréhensibles par l'homme :

- **Description textuelle** : La description du modèle se fait avec du texte et c'est un tout autre modèle qui se charge de cette tâche. Par exemple Krening et al [11] ont entraîné un modèle basé sur l'apprentissage par renforcement*. Ils ont ensuite construit un autre modèle qui sert à faire le lien entre l'état du modèle précèdent et une explication textuelle.
- **Visualisation** : un autre moyen assez parlant pour expliquer un modèle est de visualiser ses paramètres. L'une des approches pour ce faire s'appelle T-SNE (t-Distributed Stochastic Neighbourhood Embedding) [12]. Il s'agit d'une méthode permettant de représenter un ensemble de points d'un espace à grande dimension dans un espace de deux ou trois dimensions, le principe est similaire à l'ACP (analyse en composante principale).
- **Explication locale** : au lieu d'expliquer tout le modèle globalement, on explique les changements locaux causés par une entrée bien spécifique. Parmi les méthodes d'interprétation les plus connues on retrouve la méthode LIME [13]. Cette dernière est indépendante du modèle, ce qui signifie qu'elle peut être appliquée à n'importe quel algorithme d'apprentissage automatique. La technique LIME tente de comprendre le modèle en perturbant l'entrée des échantillons de données et voir comment les prédictions changent.
  La sortie de LIME refléte la contribution de chaque caractéristique à la prédiction d'un échantillon de données.

1.2.4 Interprétation des modèles par des méthodes basées sur le gradient

Les méthodes basées sur le gradient font parties des méthodes les plus en vogue. Elles permettent d'attribuer un score de pertinence R_i à un neurone. Elles sont propres à l'apprentissage automatique et s'appuient sur l'achitecture du modèle, ses poids et biais. Ces méthodes font de la

rétro-propagation à partir de la couche de sortie.

La bibliothèque Investigate* que nous avons utilisé implémente des méthodes basées sur le gradient tels que DeepLIFT, Gradients x Inputs et LRP (Layer-Wise Relevance Propagation) que nous verrons avec plus de détails dans la section 1.2.6.

1.2.5 Outils d'investigation INNVESTIGATE

Dans le cadre de notre étude, nous utilisons un outil d'interprétation de réseaux de neurones appelé iNNvestigate réalisé par Maximilian Alber et al [7]. Cette bibliothèque regroupe l'implémentation de plusieurs méthodes d'analyse, la plupart basées sur le gradient vu antérieurement.

La performance des méthodes d'interprétation dépend du modèle que l'on souhaite interpréter, c'est pourquoi une comparaison empirique est faite pour pouvoir évaluer ces méthodes et déterminer celle dont l'analyse du réseau de neurones profond est la plus pertinente.

La comparaison qualitative et quantitative des méthodes d'analyse

La bibliothèque reçoit en entrée un réseau de neurones préalablement entrainé et procède à l'analyse, notons qu'elle est particulièrement adaptée aux entrées sous forme d'image. Il en résulte une comparaison qualitative des méthodes d'interprétation. Malheureusement, cette comparaison qualitative peut laisser place à une certaine subjectivité, par exemple sur un classifieur d'image l'analyse a donné les résultats illustré par la figure 1.5

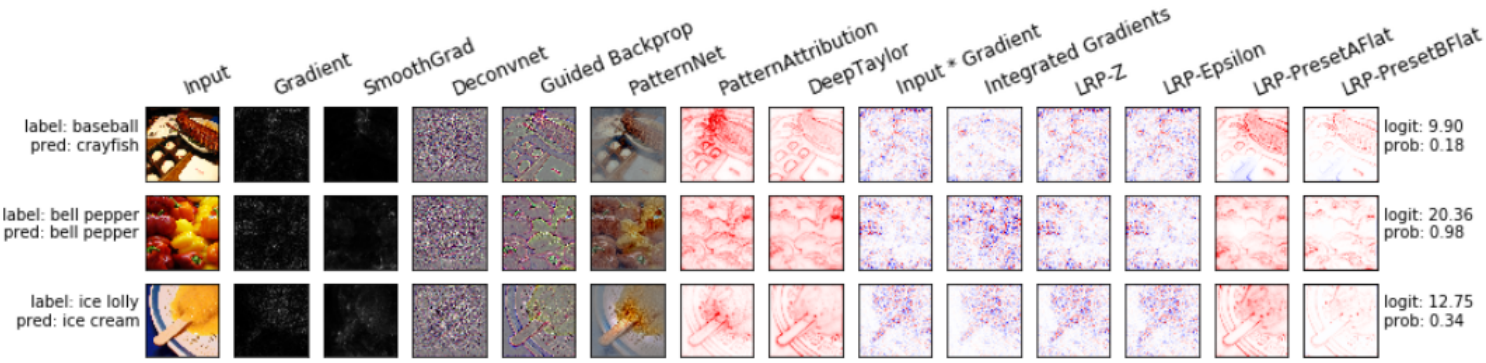


FIGURE 1.5: Comparaison qualitative des différentes méthodes d'analyse

A travers la figure 1.5, nous remarquons que la comparaison des méthodes d'analyse reste très approximative surtout pour des données vectorielles tel que les expressions de gènes*. Pour pallier à ce problème il serait interessant d'établir une comparaison quantitave plus représentative, c'est pourquoi une implémentation d'une méthode de perturbation est fournie [8] avec la bibliothèque.

L'intuition derrière cette méthode est qu'en perturbant les attributs identifiés comme importants par la méthode d'analyse, nous aurons une toute autre classification. Cela prouve qu'en effet, la méthode d'analyse a réellement identifié les attributs les plus pertinents. Il en résulte une bien meilleure compréhension des réseaux de neurones profonds.

1.2.6 Approche Layer-Wise Relevance Propagation (LRP)

Dans cette partie, nous allons nous intéresser à l'aspect théorique de la méthode d'interprétation qui a donné les meilleurs résultats dans la partie experimentation (Chapitre 2) et qui fait partie des méthodes basées sur le gradient les plus utilisées dans la littérature. Nous allons donc voir de plus près la méthode LRP pour Layer-Wise Relevance Propagation et ses différents paramètres.

1.2.6.1 Définition

La méthode LRP est une méthode d'interprétation utilisée pour expliquer un modèle, son algorithme de propagation de pertinence explique la prédiction en attribuant des scores de pertinence aux neurones importants de la couche d'entrée. [14].

Un exemple de résultats d'interprétabilité en utilisant la méthode d'analyse LRP appliqué au modèle de la reconnaissance de chiffres (MNIST) est donné dans la figure 1.6. La plupart des contours de chiffres sont identifiés comme pertinents (en rouge) et quelques pixels (en bleu) sont identifiés comme pas importants, cela correspond à une valeur de pertinence négative.

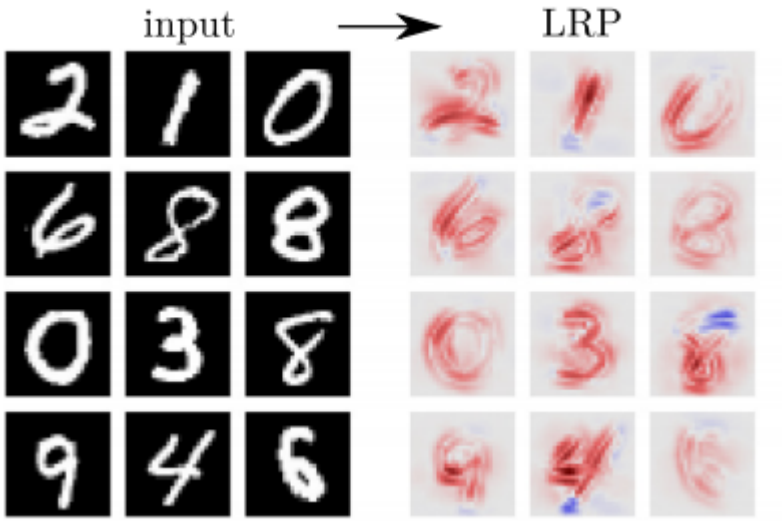


FIGURE 1.6: Interprétabilité à l'aide de la méthode LRP [16]

1.2.6.2 Principe de fonctionnement

L'algorithme de LRP consiste à rétro propager un score de pertinence d'un neurone de la couche de sortie, jusqu'à la couche d'entrée. L'objectif est d'identifier les neurones pertinents [16] .

LRP est basée sur le principe de conservation, ce qui signifie que la pertinence de toute sortie y est conservée par le processus de rétropropagation. La somme des pertinences des neurones d'une même couche est égale à la somme des pertinences des autres couches. Cette propriété est valable pour toutes les couches consécutives et par transitivité pour les couches d'entrée et de sortie.

La figure 1.7 illustre de cet algorithme

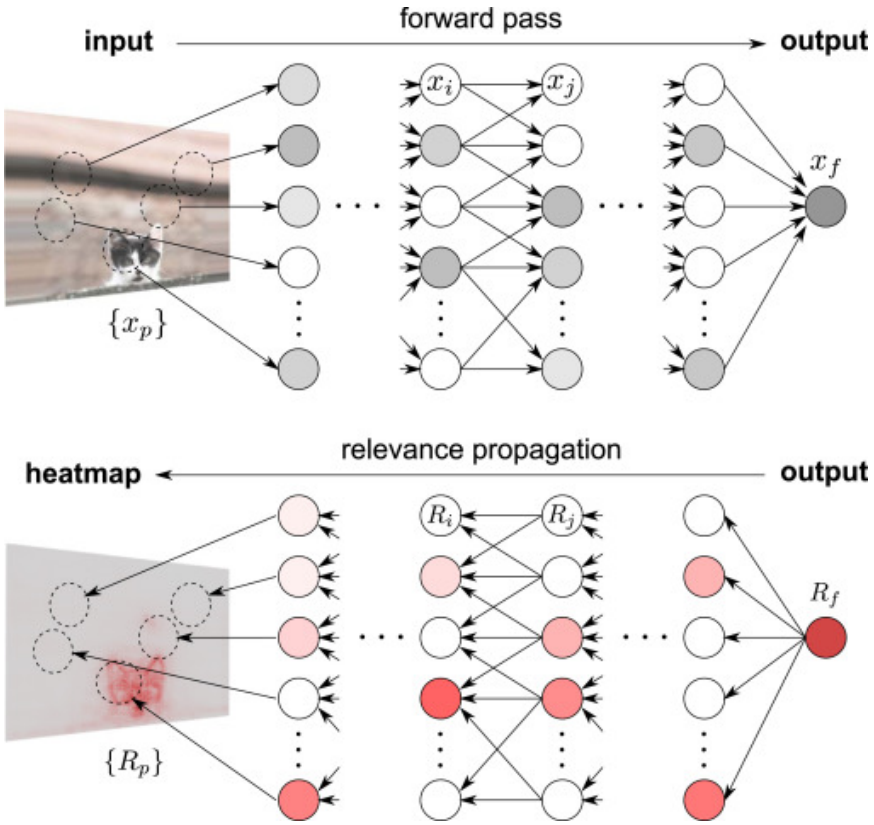


FIGURE 1.7: Procédure LRP [15]

A partir de la figure 1.7, nous pouvons constater que le modèle reçoit en entrée une image et donc une matrice de pixels, chaque pixel représente une caractéristique x_p en entrée. Le but de LRP est donc d'identifier quel pixel a contribué le plus lors de la prédiction. Pour ce faire, nous pouvons remarquer que l'algorithme passe par deux phases :

1. Phase de propagation (*forward pass*) : cette étape consiste à faire la prédiction à partir des données d'entrée. Il en résulte une sortie x_f considerée comme son score de pertinence R_f.
2. Phase rétro propagation (*relevance propagation*) : Cette opération se fait à partir du score de pertinence de la sortie R_f tout en appliquant de la rétro propagation vers les couches précédentes. L'objectif est d'identifier les neurones les plus pertinents, pour cela un score de pertinence est calculé pour chaque neurone. ce dernier prend en compte l'activation du neurone et son poids et le calcul se fait grace à des règles de propagation, l'une des plus connue est la règle $\alpha\beta$-rule qui est donnée par la formule 1.3 :

$$R_i = \sum_j \left(\alpha \frac{a_i w_{ij}^+}{\sum_j a_i w_{ij}^+} - \beta \frac{a_i w_{ij}^-}{\sum_j a_i w_{ij}^-} \right) R_j \tag{1.3}$$

- la couche i précède la couche j
- R_i : représente la pertinence totale des neurones de la couche i
- a_i : correspond à l'activation d'un neurone de la couche i
- w_{ij}^+ : correspond au poid positif entre un neurone de la couche i et un neurone de la couche j

- w_{ij}^{-} : correspond au poid négatif entre le neurone i et le neurone j. Il est à noter que l'injection de pertinence négative est contrôlée par des hyperparamètres.
- α et β sont deux paramètres avec comme contrainte $\alpha - \beta = 1$ et $\beta \geq 0$.

Par convention, on utilise la notation LRP $\alpha_n\beta_m$ quand α et β valent respectivement n , m. Par exemple LRP $\alpha_1\beta_0$ correspond à l'équation 1.4 :

$$R_i = \sum_j \frac{a_i w_{ij}^{+}}{\sum_j a_i w_{ij}^{+}} R_j \tag{1.4}$$

Différentes combinaisons des paramètres $\alpha\beta$ peuvent changer le résultat de l'interprétabilité. La figure 1.8 montre l'effet de variation de ces paramètres sur le modèle de la reconnaissance de l'écriture manuscrite (MNIST)

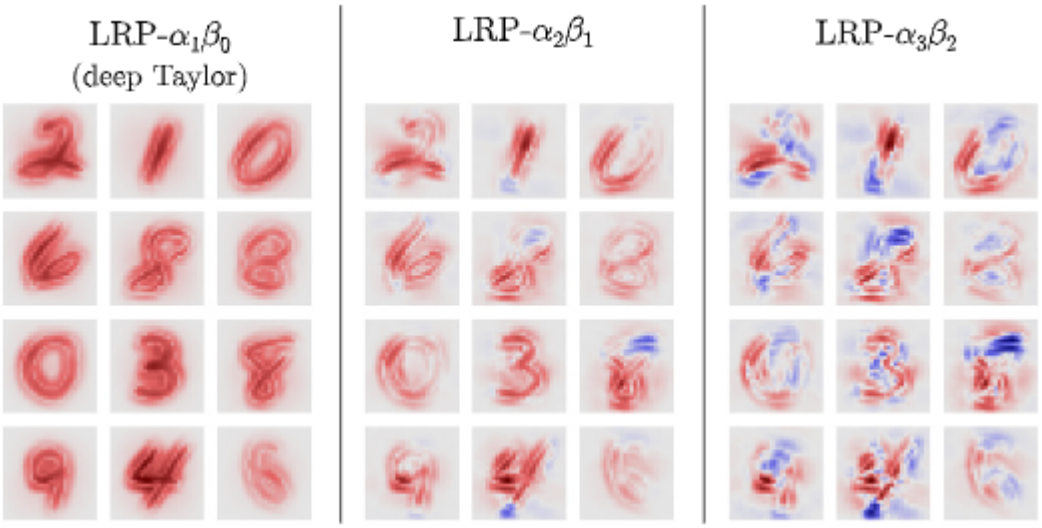


FIGURE 1.8: Variations de α, β [16]

En plus de la règle LRP $\alpha\beta$ vue dans l'équation 1.3 , la bibliothèque Innvistigate que nous avons utilisé comporte d'autres règles de propagation. Le tableau 1.1 nous montre ces dernières.

Nom	**Formule**	**Explication**
LRP $\alpha\beta$	$\mathrm{R}_i = \sum_j \left(\alpha \frac{a_i w_{ij}^+}{\sum_j a_i w_{ij}^+} - \beta \frac{a_i w_{ij}^-}{\sum_j a_i w_{ij}^-} \right) R_j$	La règle de base avec α et β comme paramètres
LRP- EPSILON	$\mathrm{R}_i = \sum_j \frac{a_i w_{ij}}{\sum_j \varepsilon + a_i w_{ij}} R_j$	Consiste à ajouter ε au dénominateur. Le rôle de ε est de négliger les pertinences lorsque les contributions à l'activation du neurone j sont faibles
LRP Z PLUS	$\mathrm{R}_i = \sum_j \frac{a_i w_{ij}}{\sum_j a_i w_{ij}} R_j$	Cas particulier de la règle LRP $\alpha\beta$ où $\alpha = 1$ et $\beta = 0$
LRP Z PLUS FAST	$\mathrm{R}_i = \sum_j \frac{a_i w_{ij}}{\sum_j a_i w_{ij}} R_j \quad w_{ij} \geq 0$	A la différence de LRP Z PLUS , LRP Z PLUS FAST considère que les poids positifs $w_{ij} \geq 0$
LRP W^2	$\mathrm{R}_i = \sum_j \frac{w_{ij}^2}{\sum_j w_{ij}^2} R_j$	Dans cette formule de propagation on ne prend pas en compte l'activation des neurones mais uniquement les carrés des poids

TABLE 1.1: Formules des différentes règles de propagation ([17])

Chapitre 2

Interprétation d'un réseau de neurones profond pour la prédiction du type de Leucémie

2.1 Description des données

Le problème étudié pour ce projet se situe dans le contexte de la détection de la Leucémie et son type, nous utilisons pour cela des données disponibles en ligne : Des investigations executées sur 11 laboratoires différents, où des mesures d'expressions de gènes ont été prises sur des patients distincts, et ce associé à leurs profils de Leucémie (leurs Types).

Le jeu de données fourni est encodé sous format GSE ; une sérialisation d'ensemble de fichiers GSM ; qui eux sont des expériences individuelles. Ces derniers contenant les données pertinentes à l'apprentissage.

Ainsi dans chaque expérience où une mesure est prise sur un patient, nous possédons un vecteur de caractéristiques associé à chaque expression de gène, et contenant les informations suivantes :

— Son identifiant dans la série.
— Une mesure de signal du génome prise selon le protocole DS.
— Une mesure faite avec le protocole DQN3.
— La valeur P associée à chaque signal mesuré.

Enfin pour une même expérience nous pouvons extraire des métadonnées dont les plus utiles pour ce projet sont :

— Le titre de l'expérience associée.
— La numérotation de l'experience dans sa série.
— Le nom de l'organisme qui a effectué l'experience.
— Le protocole utilisé.
— Le type de Leucémie associé au patient ce qui servira ainsi de label pour cette entrée.

La lecture de ce type de fichier se fait à l'aide de la librairie GEOparse de Python.

2.2 Protocole expérimental

2.2.1 Préparation des données

Pour optimiser le processus d'apprentissage certaines précautions sont à prendre, celles-ci amélioreront son efficacité et ses résultats :

— Les étiquettes du jeu de données sont sous forme de 18 catégories, qui peuvent être réduites à 6 catégories principales. Effectuer ce type de réduction (diminution de nombre de classes) permet d'obtenir un modèle bien plus efficace en y perdant cependant la spécification de sous-catégorie. Ces catégories sont representées dans le tableau 2.1

Classes	Sous-Classes	Nombre de patients
ALL	Mature B-ALL with t(8 ;14)	750
	Pro-B-ALL with t(11q23)/MLL	
	c-ALL/pre-B-ALL with t(9 ;22)	
	T-ALL	
	ALL with t(12 ;21)	
	ALL with t(1 ;19)	
	ALL with hyperdiploid karyotype	
	c-ALL/pre-B-ALL without t(9 ;22)	
AML	AML & AML with t(8 ;21)	542
	AML with t(15 ;17)	
	AML with inv(16)/t(16 ;16)	
	AML with t(11q23)/MLL	
	AML with normal karyotype + other abnomalities	
	AML complex aberrant karyotype	
CLL		448
CML		76
MDS		206
Non-leukemia		74

TABLE 2.1: Récapitulatif de la distribution des individus par types de Leucémie

— Encoder les classes sous un format de "One-Hot encoding" signifiant ainsi qu'une etiquette* sera sous format 6 bits, un bit associé à chaque classe. Ainsi pour un patient donné seulement un seul de ces bits sera mis à un. Ce type de transformation d'etiquettage permet non pas d'avoir un seul neurone en sortie du modèle mais plusieurs, améliorant ainsi la précision de la prédiction. Cette méthode a fait ses preuves dans l'amélioration de la précision dans le contexte de la classification [2].

2.2.1.1 Gestion des Données Désquilibrées

Les données du jeu de données étant desiquilibrées* , certaines mesures sont à prendre afin d'éviter un surapprentissage sur certaines classe, un modèle pouvant avoir tendance à maximiser son accuracy ($\frac{Nombre\ d'éléments\ prédits\ correctement}{Nombre\ d'éléments\ total}$) prédit le plus souvent la classe majoritaire.

Pour que le choix d'un modèle se fasse de manière pertinente il faut utiliser une fonction d'évaluation adaptée au jeu de données et ses caractéristiques, dans ce contexte de classes désequilibrées nous avons opté pour la mesure de précision (Formule 2.1) qui est une fonction prenant en compte les faux positifs ce qui permet de détecter si une classe se retrouve prédite majoritairement. De plus, il est important dans un but analytique d'effectuer le calcul d'une matrice de confusion qui représente pour les données : leurs véritables classes parallélement à leurs classes prédites.

$$PRECISION = \frac{\sum_{i=0}^{N} VP}{\sum_{i=0}^{N} VP + \sum_{i=0}^{N} FP} \tag{2.1}$$

- VP : Vrais Positifs.
- FP : Faux positifs.
- N = Nombre de patients (de données).

2.2.2 Processus d'Apprentissage du Réseau de Neurones Profond

Le processus d'apprentissage se fait en plusieurs étapes.

— Certains choix ont été pris concernant des hyperparamètres.

— L'architecture que nous avons choisi est le Perceptron multi-couches car c'est une des plus simples et compatible avec les méthodes d'interprétation.

— Pour les couches intérmediaires la fonction d'activation retenue est ReLU car elle est très utilisée dans plusieurs architectures populaires et est simple et fiable [1] et Sigmoid pour les 6 neurones de sortie.

Durant les experimentations nous avons fait varier le nombre de couches cachées entre 3 et 10, ainsi que le nombre de neurones par couche en avançant par palliers : 50, 100, 1000 et enfin 1500. De plus, afin d'éviter le problème des minimums locaux* nous avons construit pour une même architecture des modèles avec 3 initialisations différentes.

Au fur et à mesure de l'entrainement, le pas d'apprentissage α a dû être ajusté, car lorsque sa valeur était trop élévée (10^{-3} par exemple) cela induisait que le taux d'erreur ne diminuait plus, cependant si sa valeur était trop faible la convergence s'effectuait beaucoup plus lentement. Au final, la valeur de α nous ayant permis d'obtenir la meilleure convergence était de 10^{-4} mais ce pas étant assez petit nous avons augmenté le nombre d'epochs d'entraînement afin de permettre aux modèles de converger vers un minimum.

Différents algorithmes d'optimisations ont été testés : ADAM, Stochastic Gradient Descent ainsi que la Descente du Gradient classique. Cependant ADAM malgré son efficacité réputée [3] n'a pu atteindre que le maximum de 85% en précision, le Stochastic Gradient Descent quant à lui a atteint les 93%.
Le jeu de données a la particularité que pour chaque expression de gène deux types de mesures ont été prise : DS et DQN3, l'experimentation en effectuant l'apprentissage en utilisant les mesures DS n'a atteint que le score maximal de 60% de précision, et par conséquent nous avons opté pour l'utilisation de la seconde mesure. Enfin l'ajout de Dropout a permis l'amélioration de la précision, plus particulièrement pour la dernière classe qui est très peu présente dans le jeu de données et dont les résultats dans la matrice de confusion se sont améliorés.

2.2.3 Interpretation du Réseau de Neurones

L'étape d'interpretation consiste à déterminer grâce à l'algorithme choisi, quelles sont les variables les plus importantes et pertinentes pour le meilleur modèle dans sa tache de prédiction.

L'analyse avec chacun des algorithmes : gradient, smoothgrad, deconvnet, guided backprop, input gradient, integrated gradients, lrp.z, lrp.epsilon est effectuée et nous pouvons dés lors observer les scores de pertinence* attribuées à chacune des variables par ces analyseurs.

Ce procédé nous retournera donc un vecteur de la même taille que l'entrée du modèle (donc 53031 ou le nombre d'expressions de gènes par individu) et où chaque expression de gène a un score d'importance attribué.

2.2.4 Calcul de Corrélation

Dans le but d'évaluer de manière plus représentative les résultats obtenus par l'analyse de chacun des algorithmes nous appliquerons un calcul de matrice de corrélation, celui-ci nous permettra de déterminer parmis eux lesquels ont une relation linéaire considérant ainsi les mêmes variables comme importantes, ou inversement. Ainsi des variables peuvent être jugées importantes dans une méthode mais pas dans une autre. Il peut également n'y avoir aucune relation entre deux analyses.

2.2.5 Perturbation des Entrées pour la Comparaison des Méthodes

La perturbation est un procédé qui consiste à prendre les variables détérminées comme pertinentes par un algorithme d'analyse (ou interprétation) et en perturber la valeur afin d'observer l'effet résultant sur la précision des prédictions en moyenne. Cependant cette fonction sous iNNvestigate est développée pour les images, il a donc fallu l'adapter au contexte des données vectorielles sachant que le concept de voisinage n'a plus de réelle signification contrairement à l'image ou vouloir perturber une région signifierait affecter une zone ou carré de pixels précis. Il existe 3 types de perturbations :

- **Perturbation par des zéros** : on remplace les valeurs de variables en entrées par des zéros.
- **Perturbation par des valeurs aléatoire** : on remplace les valeurs par une valeur aléatoire.
- **Perturbation par des moyennes** : on calcule la moyenne de valeur des expressions gènes de l'individu, et on l'utilise pour remplacer ces variables importantes.

Nous avons effectué la perturbation de manière incrémentale sur les 1000 variables que chaque modèle considérait comme les plus importantes afin d'en observer l'impact sur la précision de celui-ci. Celle-ci nous permettra dans un premier temps de comparer les méthodes d'interprétation entre elles, et enfin la qualité des types de perturbation.

La qualité d'un algorithme d'analyse se définit par la vitesse à laquelle décroit la précision du modèle lorsque l'on perturbe les valeurs des caractéristiques que cet analyseur a identifié comme pertinentes [8].

2.2.6 Hyperparamètrage de LRP

2.2.6.1 Interpretation et Analyse

Afin d'approfondir notre analyse nous nous concentrerons sur la méthode Layer-Wise Relevance Propagation (ses différentes variantes) nous ajusterons ses hyperparamètres afin d'observer l'effet que cela aura sur la qualité de l'analyse.

2.2.6.2 Perturbation des expressions de gènes

Le processus de comparaison des paramètrages de LRP s'effectue de la même manière que la comparaison entre les différents algorithmes d'interpretation. Pour ce faire nous appliquerons la même méthodologie citée précedemment, dans un premier temps en effectuant une analyse en observant les proportions de variables considérées comme importantes, puis en les perturbant et observant les courbes résultantes.

2.2.7 Analyse par classe de Leucémie

2.2.7.1 Expressions de gènes pertinentes

Après ces étapes nous nous attarderons sur l'observation de l'analyse LRP en fonction des classes de Leucémie, afin de déterminer l'existence ou non d'une homogénéité dans les pertinences de variables d'une catégorie à une autre ; par exemple : les variables permettant de prédire une Leucémie de type ALL sont-elles les même que celles pour prédire une de type CLL ?

2.2.7.2 Analyse en composantes principales (ACP)

L'ACP ou Analyse en composantes principales est une méthode d'analyse de données permettant la projection de données ayant un grand nombre de variables (et faisant partie du même ensemble) sur un nombre réduit d'axes décorrelés que l'on appelle *Axes Factoriels*. Nous allons utiliser cette méthode afin de pouvoir mieux visualiser nos résultats graphiquement en représentant sur les deux axes factoriels résultants les analyses faites sur chacun des types de Leucémie.
Ainsi nous représenterons les 53031 pertinences de gènes fournies par LRP sur une dimensionalité réduite.

2.2.7.3 Partitionnement des Données

Le partitionnement de données (ou "Clustering" en Anglais) est une autre méthode d'analyse de données permettant de créer des groupes homogènes, l'objectif étant de regrouper les individus les plus similaires entre eux.

L'algorithme auquel nous ferons appel est le Partitionnement Hiéarchique ; se basant sur la distance euclédienne et regroupant les individus les plus proches entre eux, deux par deux et ce jusqu'à former un seul et même groupe ou arbre.

Nous l'appliquerons sur les pertinences de gènes fournies par LRP dans le but d'associer des signatures constituées de gènes à chaque type de leucémie. Ces gènes pourront alors être considérés comme biomarqueurs potentiels.

Chapitre 3

Résultats

3.1 Apprentissage du Réseau de Neurones Profond

Les résultats de précision obtenus par le processus d'apprentissage sont représentés dans le tableau 3.1

	Nombre de neurones				
Nombre de couches	50	100	1000	1500	2000
3	0.887	0.890	0.880	0.883	0.888
4	0.892	0.8808	0.879	0.8875	0.888
5	0.8891	0.9303	0.8933	0.8813	0.8811
6	0.90452	0.8921	0.8748	0.90120	0.86747
7	0.8970	0.864	0.89224	0.875	0.896

TABLE 3.1: Résultats

On peut constater que les résultats sont autours des 90% pour la plupart, cependant l'architecture performant le mieux est celle avec 5 couches cachées et 100 neurones par couche et par conséquent elle sera retenue pour la suite du projet.

3.2 Calcul de Corrélation

Après l'application des différents algorithmes d'analyse et interpretation sur notre réseau de neurones afin de mieux visualiser les variables que chaque algorithme définit comme pertinentes nous avons appliqué un calcul des coefficients de corrélations

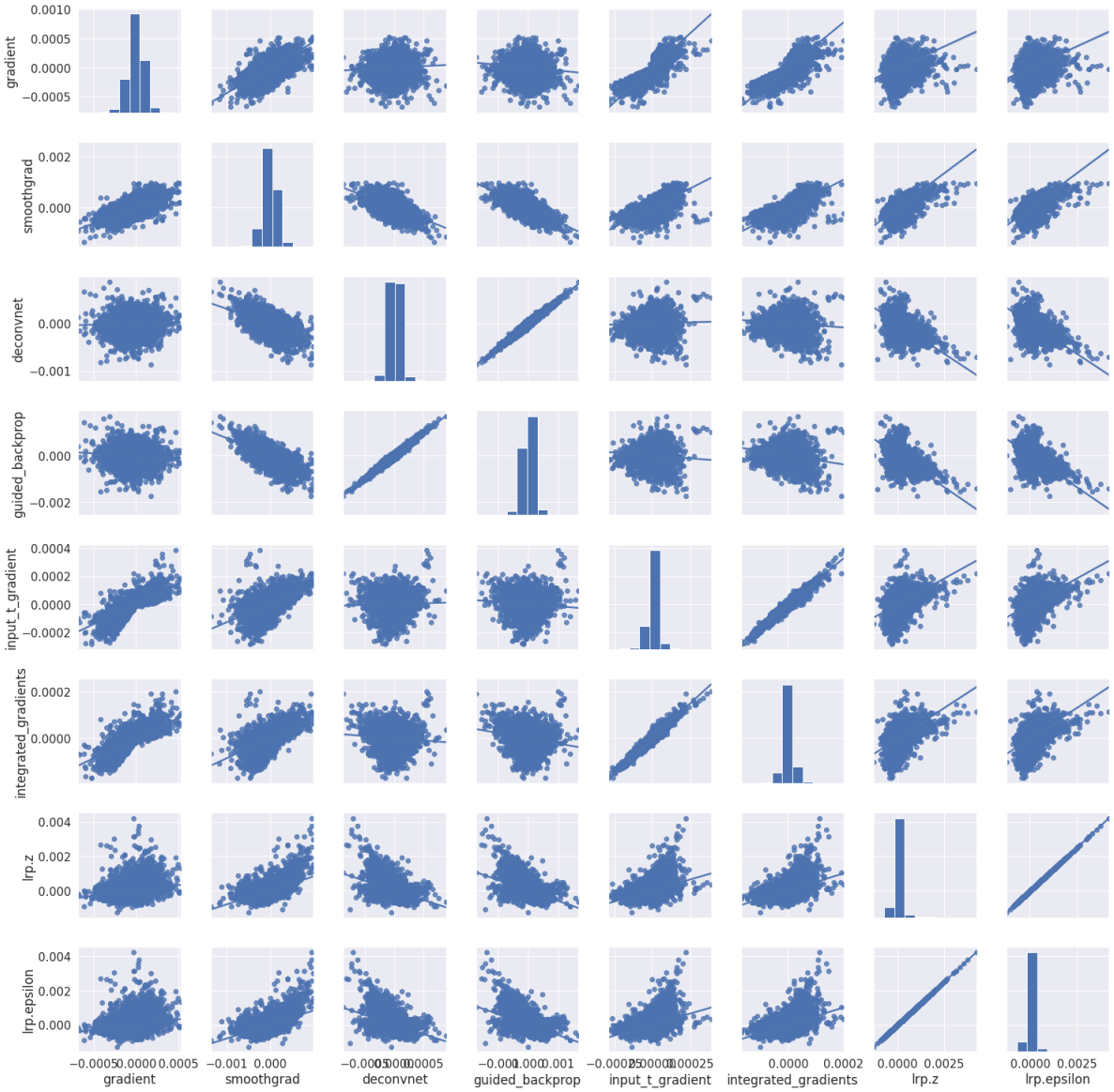


FIGURE 3.1: Correlations entre les algorithmes d'Interprétation

Dans la figure 3.1 on peut observer une forte corrélation entre les deux méthodes de LRP signifiant qu'elles considérent globalement les même variables importantes, pareil pour Guided Backprop et input t gradient. D'une autre part on remarque que les méthodes Deconvnet et Guided Backprop ont un coefficient se rapprochant du -1 ce qui veut dire qu'elles ont une relation inverse et que donc les variables que l'un définit comme importante, l'autre méthode aura tendance à la définir négligeable.

3.3 Perturbation des données

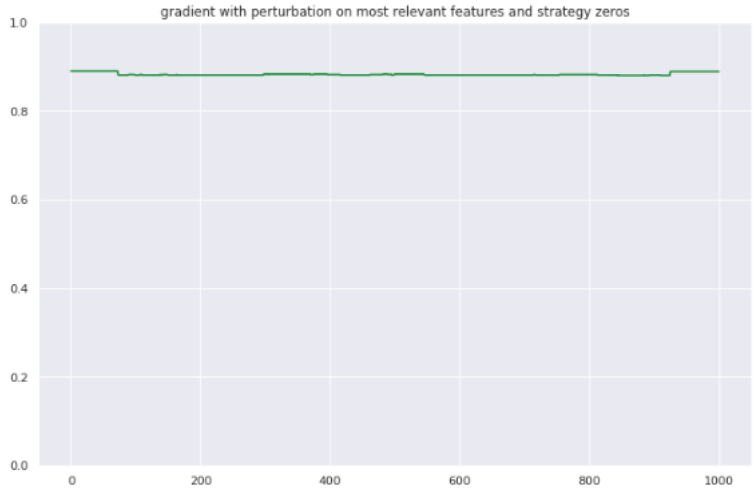
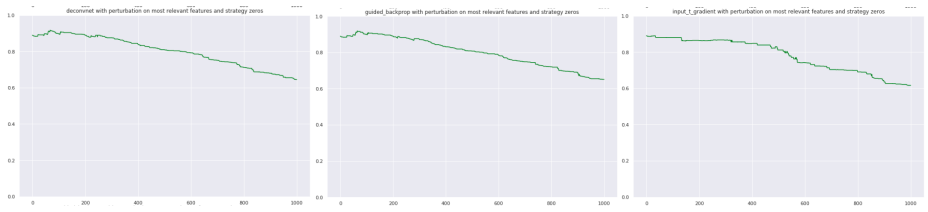


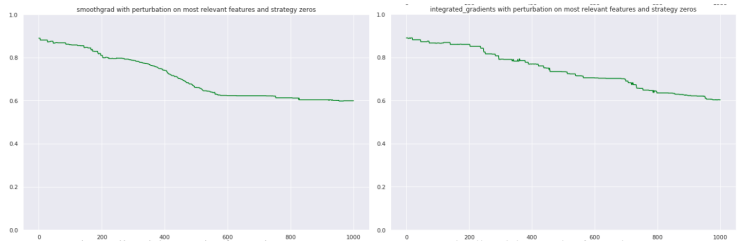
FIGURE 3.2: Gradient

D'abord on observe certaines méthodes dont la perturbation des entrées considérées importantes n'affecte point la précision de la prédiction c'est le cas pour la méthode de gradient (Figure 3.2) , et par conséquent on en déduit que l'analyse est erronée.

De plus on peut observer de par les résultats de la figure 3.3 certaines méthodes donnent décroissance constante au fur et à mesure des perturbations, résultat que l'on peut interpreter par une équivalence d'importance de ces variables selectionnées.



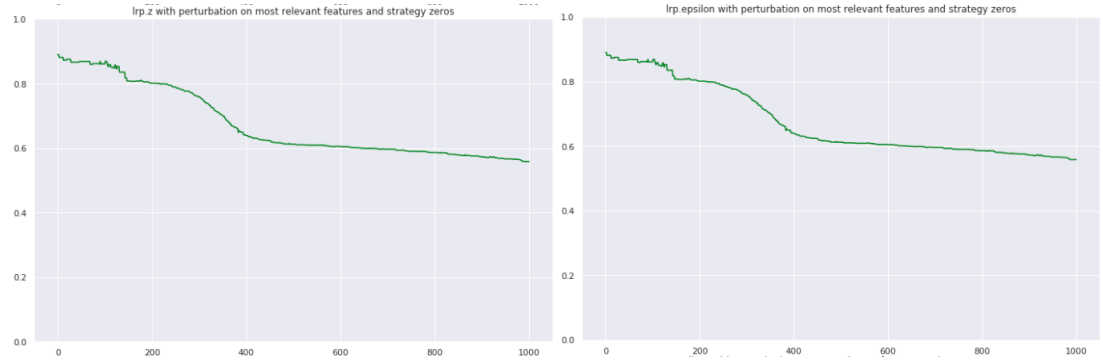
(a) Deconvnet (b) Guided BackProp (c) Input t gradient



(d) Smoothgrad (e) Integrated Gradients

FIGURE 3.3: Méthodes à décroissance régulière

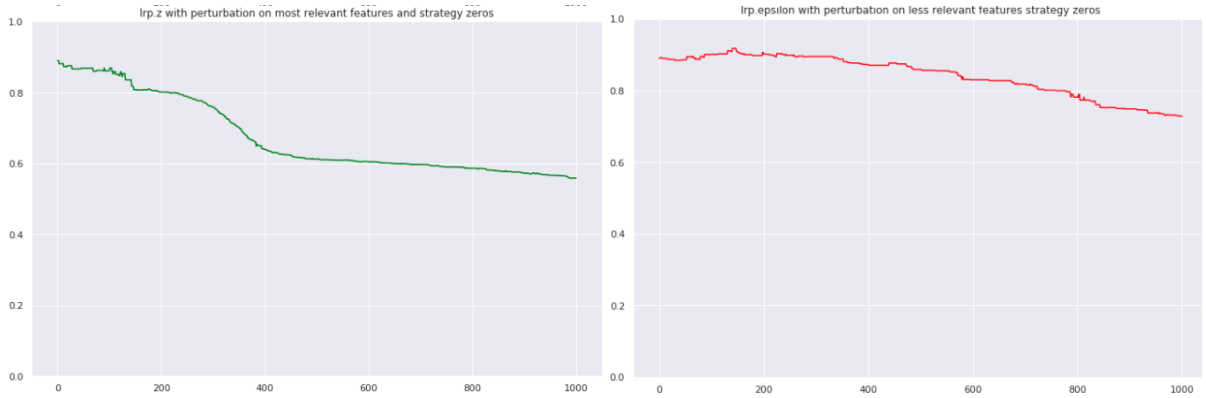
Enfin, les cas qui nous intéressent sont ceux dont la courbe initiale est la plus grande, signifiant que les variables déterminées ont une importance toute particulière dans la réalisation de la prédiction.



(a) LRP.Z (b) LRP.EPSILON

FIGURE 3.4: Méthodes à courbes prononcées

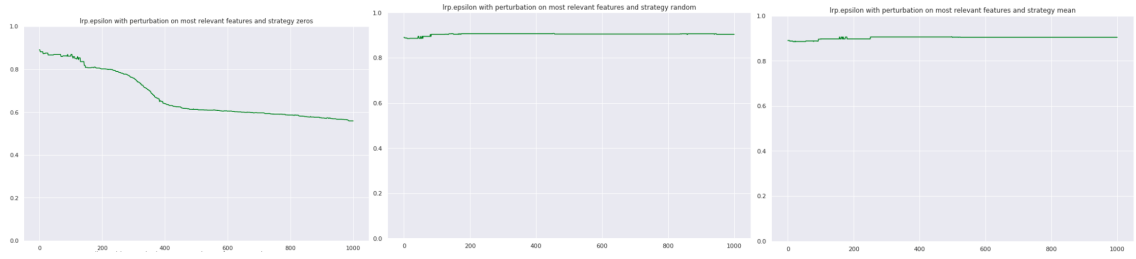
On remarque dans la figure 3.4 que les méthode dont les courbes sont les plus prononcées de toutes sont LRP.Epsilon et LRP.z, on peut donc en conclure que cette méthode a une efficacité plus prononcée sur ce modèle déterminant ainsi les variables les plus décisives ce qui va nous pousser à nous intéresser plus précisement à cette analyseur.
Afin d'appuyer ce propos comparons le processus de perturbation dans un premier temps sur les variables qu'il juge les plus importantes et dans un second celles qu'il considère comme les moins importantes.



(a) LRP.Z plus importantes variables (b) LRP.Z moins importantes variables

FIGURE 3.5: Comparatif des importances de variables

On remarque dans la figure 3.5 une claire différence entre cette perturbation et celle initiale, ici le modèle stagne dans sa perturbation confirmant ainsi que l'analyseur a été efficace dans son évaluation.
Concernant les différentes méthodes de perturbation implémentées analysons leurs impacts sur la précision du modèle.



(a) Perturbation par des zéros (b) Perturbation par des valeurs aléatoires (c) Perturbation avec la moyenne

FIGURE 3.6: Comparatif des méthodes de perturbation

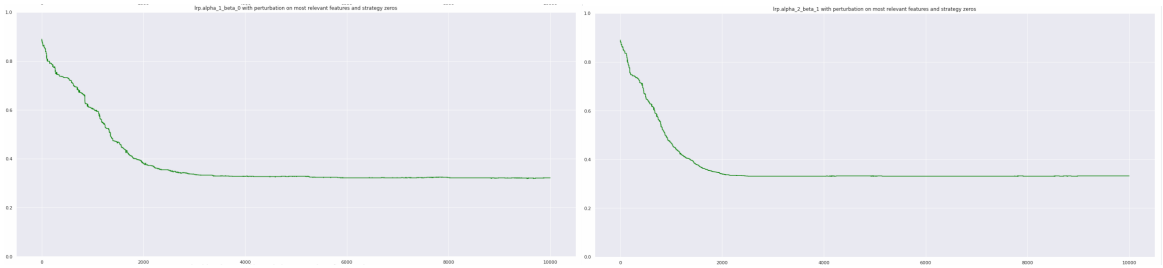
Avec les 2 autres méthodes que sont la perturbation par moyenne et par valeur aléatoire on constate sur la figure 3.6 une différence d'impact qui se retrouve réduit réduit de manière plus que conséquente comparativement à la perturbation avec des zéros.

3.4 Hyperparamètrage de LRP

3.4.1 Perturbation des expressions de gènes

En appliquant une perturbation des données d'expressions de gènes considérées comme importantes par ces variantes de LRP nous avons pu observer les résultats ci-dessous :

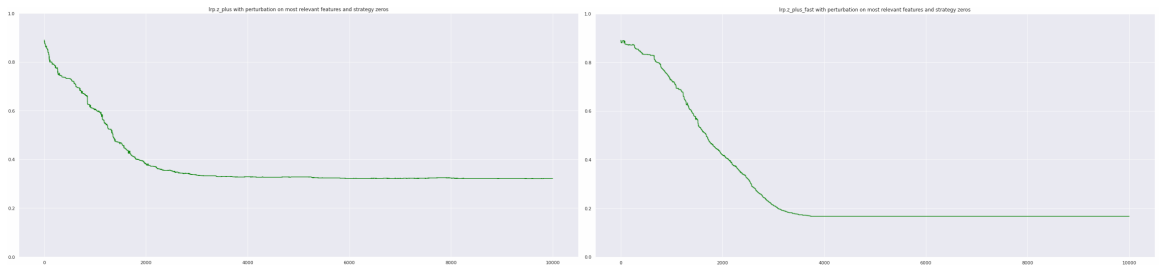
En faisant varier les valeurs de α et β on obtient cependant des résultats plus ou moins similaires (Figure 3.7)



(a) alpha=1 beta=0 (b) alpha=2 beta=1

FIGURE 3.7: Paramètrage d'Alpha et Beta

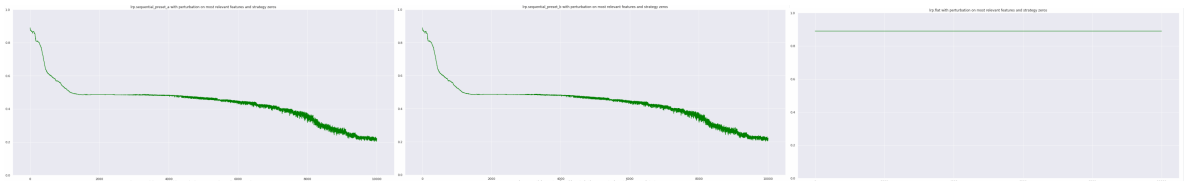
Les versions "plus" et "plus fast" quant à elles de LRP n'ont quant à eux obtenu que des résultats peu satisfaisants (Figure 3.8) comparativement au reste



(a) LRP Z Plus (b) LRP Z Plus Fast

FIGURE 3.8: Méthodes Plus et Plus Fast

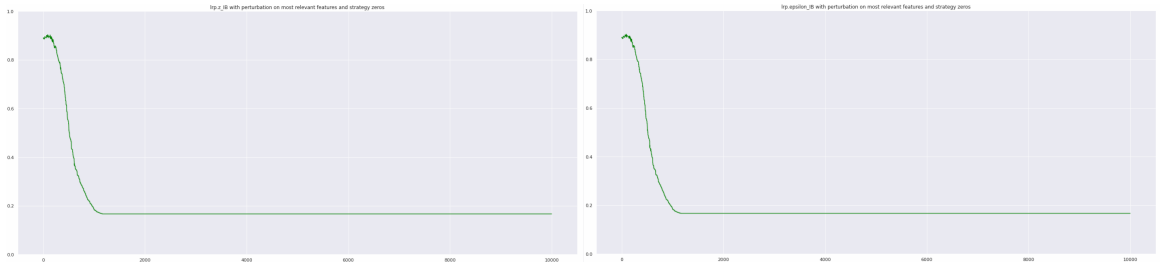
Les pires résultats observés (Figure 3.9) ont été sur les méthodes LRP Sequential avec preset sur a et sur b, ainsi que sur la version Flat de LRP



(a) LRP Sequential Preset A (b) LRP Sequential Present B (c) LRP Flat

FIGURE 3.9: Méthodes d'intérpretation Preset et Flat

Les meilleurs résultats ont été observés sur les méthodes lrp.z.ib et lrp.epsilon.ib et sont plutôt similaires (Figure 3.10)



(a) LRP Z IB (b) LRP Epsilon IB

FIGURE 3.10: Méthodes IB

3.5 Analyse par classe de Leucémie

3.5.1 Expressions de gènes pertinentes

Après application de LRP sur chaque classe séparemment on peut observer des résultats très variés d'une catégorie de Leucémie à une autre, on en déduit que les variables pertinentes à la prédiction peuvent varier d'un type de Leucémie à un autre (Figure 3.11)

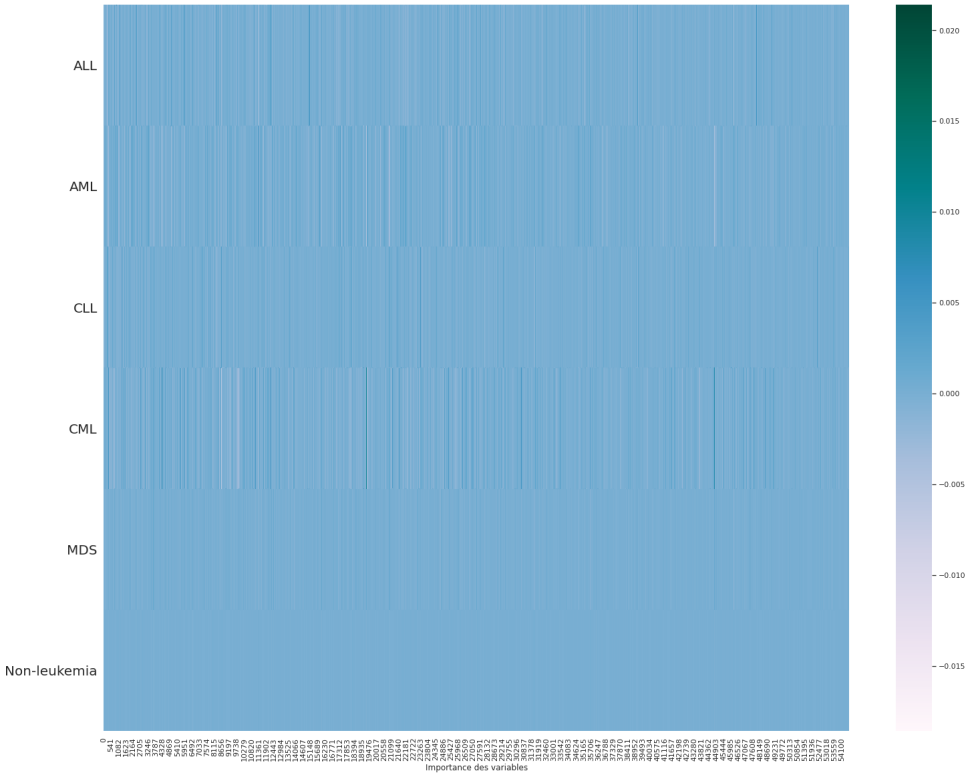
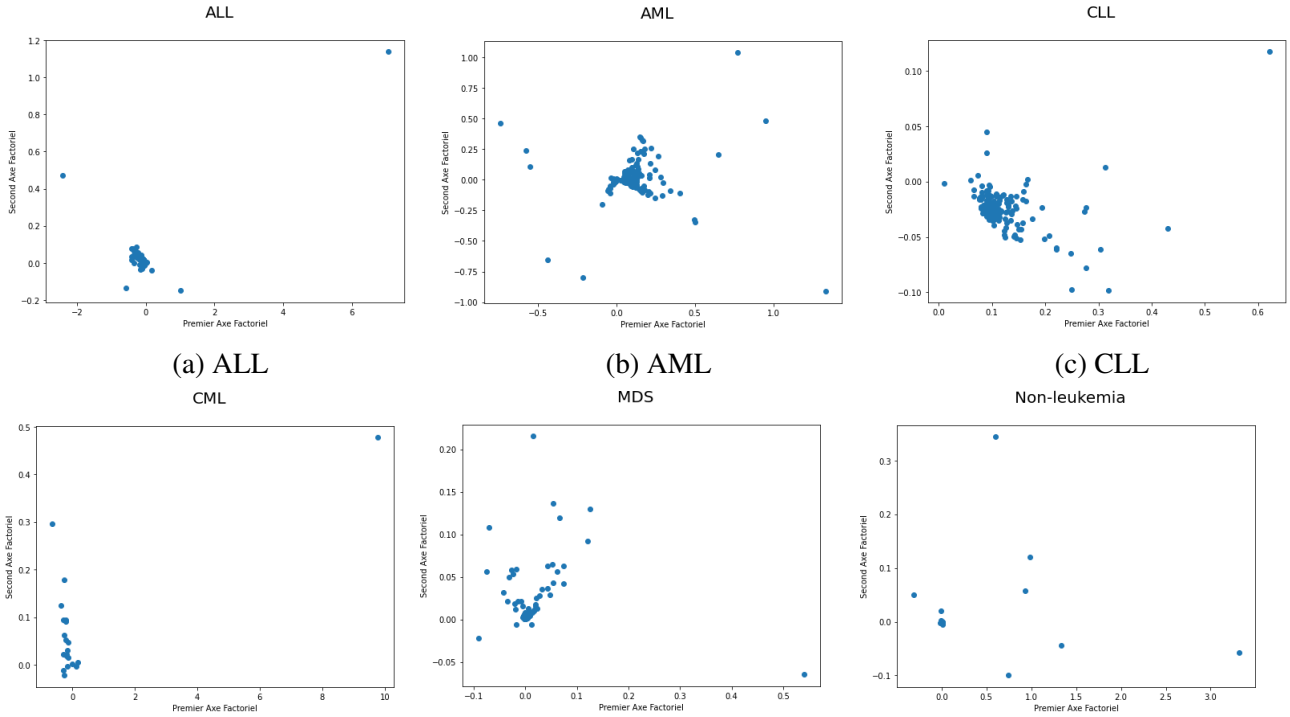


FIGURE 3.11: Pertinences de gènes fournies par LRP pour chaque type de Leucémie

3.5.2 Application de l'Analyse en composantes principales

L'application de l'Analyse en composantes principales a permis d'obtenir deux axes factoriels qui représentent les 53031 dimensions initiales pour chaque pertinence d'expression de géne et grâce à cela nous pouvons afficher ces graphes (Figure 3.12) qui permettent de représenter spatialement pour chaque catégorie de Leucémie : Les individus associés à ce type de Leucémie. Nous pouvons ainsi remarquer, et ce pour tous les types mise à part Non-Leukemia, des concentrations de ces individus dans certains secteurs, on peut donc en déduire que pour un même type de Leucémie les expressions de gènes ayant été pertinentes à sa prédiction pour chacun de ses individus sont très similaires.



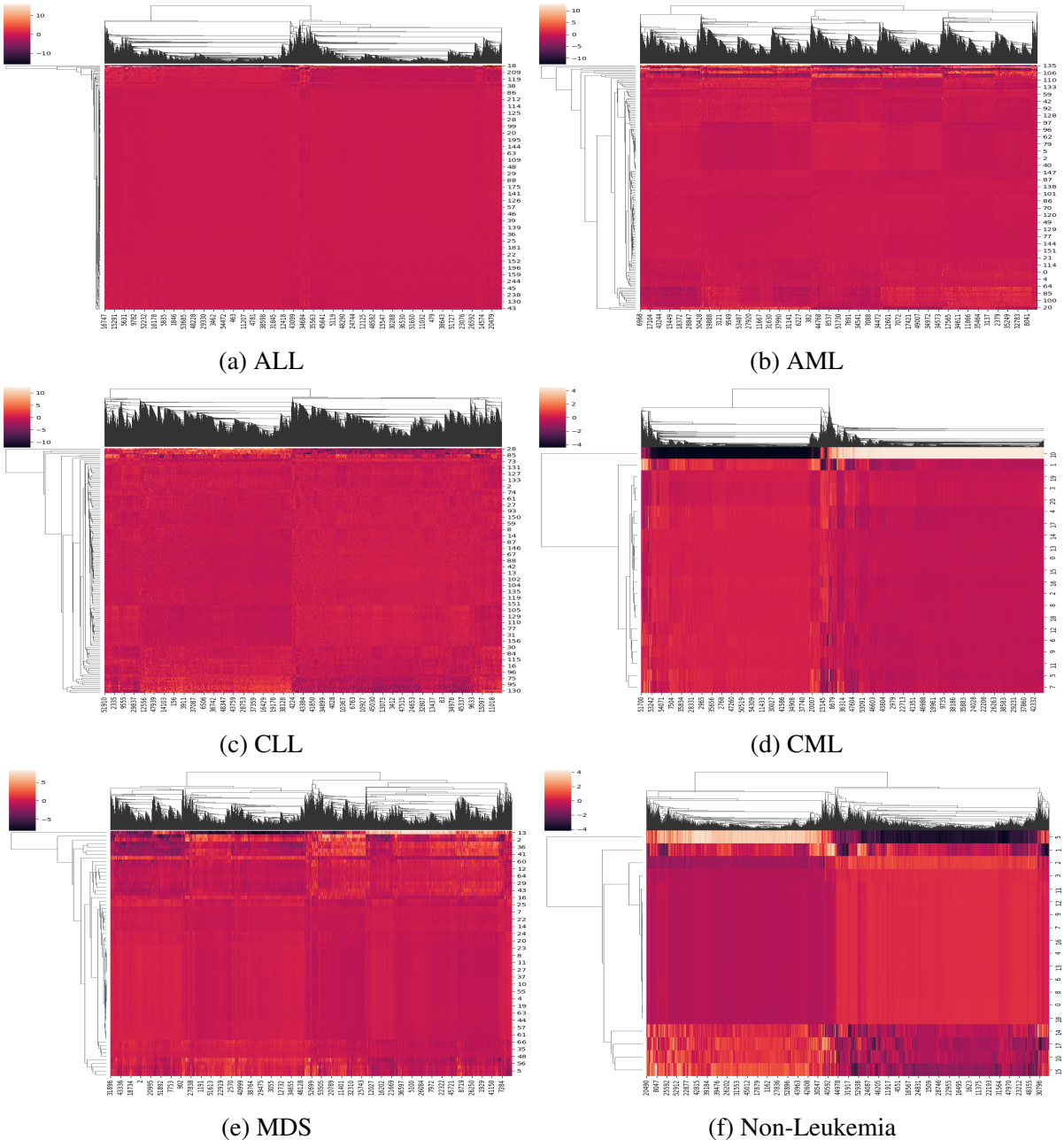
(a) ALL (b) AML (c) CLL

(d) CML (e) MDS (f) Non-Leukemia

FIGURE 3.12: ACP pour chaque type de Leucémie

3.5.3 Partitionnement des Données

Après avoir obtenu une représentation conscise nous effectuons la procédure de partitionnement hiéarchique sur ces classes.



(a) ALL (b) AML

(c) CLL (d) CML

(e) MDS (f) Non-Leukemia

FIGURE 3.13: Partitionnement hiéarchique pour chaque type de Leucémie

Dans la figure 3.13 chaque ligne représente un individu, et chaque colonne la pertinence de l'expression de gène fournie par LRP. Nous pouvons observer pour plusieurs types de Leucémie que des clusters de similarité se créent, ainsi des variables sont regroupées entre elles, ces groupes de variables liées dans le contexte de la prédiction d'un type de Leucémie peuvent être considérées comme biomarqueurs potentiels pour celui-ci et ainsi être un potentiel de piste de recherche médicale.

Conclusion

Le but de ce projet était de répondre à plusieurs questions : quel est le potentiel d'efficacité des méthodes d'interprétation sur des réseaux de neurones traitant des données vectorielles ? Peut-on le représenter de manière visuelle ? Comment évaluer et perturber des variables dans ce contexte ? Peut-on développer de la connaissance par ce procédé ?

A travers ce travail de recherche nous avons pu aboutir à plusieurs conclusions :

- La méthode LRP est très efficace sur les données vectorielles d'expressions de gènes et ce de manière supérieure aux autres algorithmes d'interprétation existants.
- Malgrés les différences existantes entre les principes des méthodologies d'interpretation de réseaux de neurones, on peut retrouver une corrélation entre leurs résultats, qu'elle soit positive (accord commun sur les variables pertinentes) ou négative (contradiction sur les variables pertinentes).
- Nous avons pu observer que les variables pertinentes à une prédiction dans un même jeu de données peuvent varier d'une classe à une autre ainsi la prédiction d'une Leucémie de type ALL dépendait de variables différentes que la prédiction du type AML par exemple.

Ces conclusions pourraient ouvrir de nouvelles pistes dans les domaines scientifiques dont sont issus les jeux de données, notre objectif dans ce projet ayant été d'observer le comportement des algorithmes et techniques d'inteprétations (habituellement utilisées sur des images) sur des données vectorielles. Ainsi les expressions de gènes ayant fait preuve d'importance dans la catégorisation de Leucémie (au travers du processus d'inteprétation puis de perturbation) pourraient nous en apprendre plus sur la maladie. Les regroupements de familles (ou clusters) de types de cette maladie pourraient ouvrir des pistes vers la recherche des similarité effective entre celles-ci.

En conclusion, l'apprentissage profond peut s'avérer bien utile dans la création d'une association ou mapping d'une donnée x vers une prédiction $y = f(x)$, mais il est possible d'aller plus loin par l'inteprétation et validation de cette inteprétation par perturbation de son modèle et ses prédictions afin de mieux comprendre celles-ci ; comprenant ainsi mieux ce qui a été décisif à cela, on peut alors compléter ce procédé par des méthodes d'analyse de données créant ainsi des observations qui elles seront sources de pistes de recherche futures pour tout domaine ayant fourni ces données.

Perspectives

Plusieurs points sont à encore à développer pour aller plus loin dans notre projet :

— Effectuer des tests de la méthode LRP sur d'autres données vectorielles et déterminer si son efficacité est généralisable.

— Développer des méthodes permettant d'exposer de manière plus directe et visuelle la liaison entre un input et l'ensemble des prédictions.

— Appliquer cette méthodologie : interprétation, perturbation, ACP et clustering sur d'autres jeux de données afin d'observer quels types de connaissances pourrait-on en tirer.

Glossaire

Apprentissage Automatique L'apprentissage automatique permet a un systeme d'apprendre à partir des donnéees et non a l'aide d'une programmation explicite.

Apprentissage par Renforcement Type d'apprentissage automatique ne nécessitant pas de base données ou connaissances humaines préalables et se concentrant sur un agent et son interaction avec l'environnement grâce à un système d'actions et récompenses.

Apprentissage Profond (Deep Learning) Se définissant par les multiples variations des réseaux de neurones profonds (CNN, LSTM ect) C'est un type d'apprentissage demandant une bien plus grande quantité de données (d'ou, entre autre, sa démocratisation depuis l'apparition du Big Data).

Données désiquilibrées Jeu de données dont la quantité d'entrées dans une ou plusieurs classes est bien plus conséquent que dans le reste des classes.

Etiquette Représente la sortie associé à chaque entrée d'un jeu de donnée.

Expressions de gènes L'expression d'un gène est un ensemble de processus par lesquels l'information héréditaire stockée dans ce gène est utilisée pour fabriquer des protéines qui jouent un rôle dans le fonctionnement de la cellule L'expression d'un gène passe par un processus de transcription.

Fonction d'erreur Fonction exprimant le taux d'erreur du modèle sur les données entre l'output attendu et celui obtenu.

GPU Processeur graphique permet les calculs de l'affichage, ou des calculs parallèles matriciels.

iNNvestigate Librairie Python permettant permettant l'application de modèles d'interprétation sur des réseaux de neurones.

Intepretation de Réseau de Neurones Procéssus permettant de déterminer les variables pertinentes à la réalisation d'une prédiction par le modèle de réseau de neurones.

Keras Librairie fournissant une intéraction simple (de haut niveau) avec plusieurs architectures ou types de réseaux de neurones.

Leucémie La leucémie est un cancer des tissus responsables de la formation du sang, c'est-à-dire des cellules sanguines immatures se trouvant dans la moelle osseuse (= matière molle et spongieuse située au centre de la plupart des os).

Minimum Local Représente dans une courbe la valeur la plus basse dans une certaine plage, mais qui n'est pas forcément le minimum possible de manière générale .

Modèle Représente les poids ou paramètres résultants d'un processus d'apprentissage automatique.

Modèles Séquentiels C'est un type de réseau de neurones permettant de traiter les données séquentiels à longueur variable (que ce soit en entrée ou en sortie) comme du texte, des séquences audios.

Nvidia CUDA Plateforme de calcul parallèle de la société Nvidia tirant profit de ses cartes graphique.

Pertinence Mesure de l'importance de la variable (ou colonne) dans la réalisation de prédiction par le modèle.

Processus d'apprentissage L'ensemble des méthodologies mise en oeuvre avant, pendant et après l'apprentissage du réseau de neurones afin d'optimiser ses résultats.

Python Langage de programmation polyvalent permettant entre autre de faire : du développement web, de la recherche scientifique, du calcul haute performance, de l'apprentissage automatique.

Régression Un modèle de régression qui cherche à établir une relation linéaire entre une variable, dite expliquée, et une ou plusieurs variables, dites explicatives.

Réseau de Neurones Modèle d'apprentissage automatique inspiré du modèle biologique d'un réseau de neurones.

Réseaux Convolutifs (CNN) Réseaux de neurones tirant parti du concept de filtre, et orientés pour les tâches de classification d'images.

SVM Méthode d'apprentissage automatique basée sur une séparation linéaire entre les données de chaque classe.

Tensorflow Librairie d'apprentissage profond et de calculs haute performance.

Traitement Automatique du Langage Naturel domaine multidisciplinaire impliquant la linguistique, l'informatique et l'intelligence artificielle, qui vise à créer des outils de traitement de la langue naturelle pour diverses applications. Il ne doit pas être confondu avec la linguistique informatique, qui vise à comprendre les langues au moyen d'outils informatiques.

Table des matières

Bibliographie

[1] Nair et G. E. Hinton, “Rectified linear units improve restricted boltzmann machines,” Haifa, 2010, pp. 807–814. [Online]. https ://dl.acm.org/citation.cfm

[2] Kedar Potdar, Taher S. Pardawala, Chinmay D. Pai, ”A Comparative Study of Categorical Variable Encoding Techniques for Neural Network Classifiers”, International Journal of Computer Applications (0975–8887)Volume 175 –No.4, October 2017

[3] Diederik P. Kingma, Jimmy Lei Ba, ”ADAM : A METHOD FOR STOCHASTIC OPTIMIZATION”, ICLR 2015

[4] Wojciech Samek, Alexander Binder, Grégoire Montavon, Sebastian Lapuschkin , Klaus-Robert Müller, ”Evaluating the Visualization of What a Deep Neural Network Has Learned”, IEEE Transactions on Neural Networks and Learning Systems, vol. PP, no. 99, pp. 1-14, 2016.

[5] Gurney Kevin : *an introduction to neural networks*, UCL Press, 1997.

[6] Mathivet Virginie : *L'intelligence artificielle pour les développeurs*, édition ENI, 2015

[7] Maximilian Alber et al : *iNNvestigate neural networks !*, 2018.

[8] Samek et al., perturbation analysis, 2017.

[9] H. Dam et al., *Explainable Software Analytics*, 2018.

[10] Supriyo Chakraborty et al, *Interpretability of Deep Learning Models : A Survey of Results*, 2017

[11] Samantha Krening et al, *Learning from explanations using sentiment and advice in rl. IEEE Transactions on Cognitive and Developmental Systems*, 2017.

[12] Laurens van der Maaten and Geoffrey Hinton. *Visualizing data using t-sne*. Journal of machine learning research, 2008.

[13] Marco Tulio Ribeiro, et al. *Why should i trust you ? : Explaining the predictions of any classifier*. International conference on knowledge discovery and data mining, 2016.

[14] Sebastian Bach, et al. *The LRP Toolbox for Artificial Neural Networks*. Journal of Machine Learning Research 17 (2016).

[15] Dan Shiebler, *Understanding Neural Networks with Layerwise Relevance Propagation and Deep Taylor Series*, 2017

[16] Grégoire Montavon et al. *Methods for interpreting and understanding deep neural networks*, 2017.

[17] Gréegoire Montavon et al. *Layer-Wise Relevance Propagation : An Overview*,2019.

Annexe A

Appendix

Dans ce présent rapport, un état de l'art a été presenté dans le premier chapitre, suivi d'une partie théorique sur l'approche LRP. Le troisième chapitre a été consacré aux résultats obtenus lors de la phase expérimentation. Ce repository Github regroupe l'ensemble des Notebooks et du code source utilisé afin de réaliser ce projet.