Cahier des charges
Sujet 10 : Réseau de Neurones

Thibaut Pepin
Soumia Rezgui
Isaac Szulek
Severine Selaquet
Anthony Montigne
Arezki Slimani

14 mars 2018

Table des matières

1 Présentation

1.1 Introduction

Nous souhaitons analyser des images, un problème compliqué qui a besoin de prendre en entrée une image et dont le but est d'essayer de deviner ce que représente cette image. Pour cela il nous faut une structure qui sera capable de prendre beaucoup de données en entrée et de capturer des relations complexes entres les entrées et les sorties, c'est là qu'interviennent les réseaux de neurones artificiels.

1.1.1 Définition

Les réseaux de neurones artificiels sont des modèles inspirés du fonctionnement du cerveau animal. Ces modèles prennent en compte quelques grands principes :

- *parallélisme* : les neurones sont des entités réalisant une fonction très simples, mais fortement interconnectés ce qui rend le traitement massivement parallèle.
- *poids synaptiques* : les connections entre neurones ont des poids variables, ce qui rend les neurones plus ou moins influents sur d'autres neurones.
- *apprentissage* : ces coefficients synaptiques sont modifiables lors de l'apprentissage pour réaliser au réseau la fonction désirée.

Types Réseaux de neurones :

1. *Le perceptron monocouche :*
   Dans cette première version le perceptron était alors mono-couche et n'avait qu'une seule sortie à laquelle toutes les entrées sont connectées. Ce type de réseaux de neurones étaient limités et ne permettaient pas de résoudre des problèmes non-linéaires et des problèmes complexes.
2. *Perceptron multicouches (MLP) :*
   Les MLP (multi-layer perceptron), ou réseaux à couches, forment la très grande majorité des réseaux. Ils sont intemporels (réseaux statiques et non dynamiques).

1.1.2 Structure

Les neurones sont organisés en couches : chaque neurone est connecté aux neurones de la couche suivante, et y propage sa sortie (ces réseaux sont d'ailleurs qualifiés de feedforward (faire avancer)). La première couche du réseau est appelée couche d'entrée, c'est par cette couche que sont transmise les donnée. La dernière couche est appelée couche de sortie est c'est la qu'est récupérée la solution, chaque neurone de cette couche possède une 'étiquette', il s'agira de la solution dans le cas où l'activation de ce neurone est la plus forte. Chaque liaison entre neurone se voit associer un poids nécessaire au calcul de la solution.

Les neurones compris entre la couche d'entrée et de sortie sont appelées couches cachées.

1.1.3 Applications

Les applications des réseaux MLP sont très diverses et étendues. Elles vont de la reconnaissance de motifs, à la modélisation en passant par l'apprentissage de comportements ou de jeux (alphago ..).

1.1.4 Type D'apprentissage

On appelle apprentissage des réseaux de neurones la procédure qui consiste à raffiner les paramètres des neurones du réseau afin que celui-ci remplisse aux mieux la tâche qui lui est affectée. Le but de l'apprentissage est de permettre au réseau de neurone de généraliser à partir des exemple rencontrés lors de l'apprentissage. Nous distinguons deux types d'apprentissages :

- *Supervisé :* son fournit au réseau le couple (entrée, sortie attendue) et on modifie les poids en fonction de l'erreur entre la sortie désirée et la sortie obtenue.
  Exemple : utile aux chercheurs et aux ingénieurs qui disposent d'un ensemble de variables mesurées et d'un ensemble de mesure relative à un processus quelconque (physique, chimique, économique,financier...), du coup le but est d'établir un modèle du processus étudié à partir des mesures disponible.
- *Non Supervisé :* le réseau doit détecter des points communs aux exemples présentés, et modifier les poids afin de fournir la même sortie pour des entrées aux caractéristiques proches.
  Exemple : utiles aux applications qui permettent de retrouver des informations dont sait qu'elles doivent être présente dans les données mais on ne sait pas comment les extraire.

Pour conclure sur le choix de l'apprentissage : il n'existe pas vraiment de méthode d'apprentissage supérieur à une autre, ce choix s'effectue principalement en fonction des type de données à la disposition de l'utilisateur et du but recherché.

1.1.5 Limites des Réseaux de Neurones

Les réseaux de neurones ne fournissent pas les explications concernant leurs résultats ce qui limite l'analyse des phénomènes existants, ils peuvent être assimilés à une boîte noire qui donne une réponse quand on lui fournit les données mais qui ne délivre pas de justifications simple à analyser, ça se résume à un pouvoir explicatif limité.

Les réseaux de neurones sont donc une alternative qui peut être très efficace pour les problèmes que les algorithmes classiques ne peuvent résoudre. Un de leurs grands avantages est leur capacité à généraliser.

1.2 But du Projet

Afin de réaliser une l'analyse d'image sur de grands ensemble de données, il nous a été demandé de créer une application permettant de facilement créer et d'utiliser un réseau de neurone à l'aide d'une interface graphique. Afin de pouvoir efficacement utiliser les données à la disposition de l'entreprise nécessaire à l'apprentissage du réseau de neurone (qui sont un couple de données d'entrées et de résultats attendus) nous allons mettre un place un réseau de neurone de type MLP utilisant l'apprentissage supervisé.

2 Partie Prenant

2.1 Commanditaire

2.2 Utilisateur

De part la nature assez ouverte de l'application, l'utilisateur devra être capable de définir les paramètres du réseau de neurone en fonction de son problème et donc de posséder une certaine connaissance des réseaux de neurones.

3 Contraintes sur le Projet

3.1 Contraintes sur le Conception de la Solution

— L'application doit pouvoir tourner sous Linux.
— L'application doit utiliser un réseau de neurone de type MLP qui apprendra grâce à l'apprentissage supervisé.
— Disponibilité des données d'apprentissage au format adéquat.
— L'application doit posséder une interface graphique facile à prendre en main.

3.2 Contraintes de planning

— Cahier des charges : 14 mars 2018
— Cahier des spéci ?cations : 18 avril 2018
— Rendu ?nal : 25 mai 2018

4 Glossaire et Conventions de Dénomination

— *Traitement :* Utilisation du système par l'utilisateur pour traiter une image de son choix et obtenir la valeur correspondante attendue.
— *MLP :* perceptron multi-couches ("‘multi-layer perceptron"’).
— *Activation :* Valeur de chaque neurone (entre 0 et 1).
— *Poids :* Valeur des liens entre les neurones.

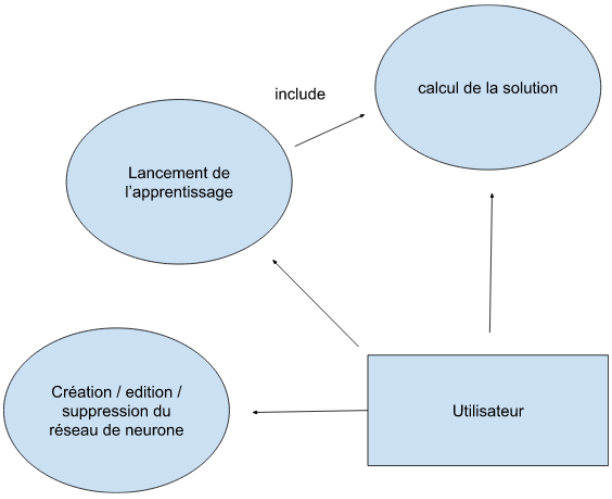
— *Biais :* Valeur définissant le seuil d'activation de chaque neurone (propre à chaque neurone).
— *Étiquette :* Signification des neurones de la couche de sortie.
— *Sigmoïde :* Fonction ramenant la valeur d'entrée a une valeur entre 0 et 1 (comme l'activation) defini tel quel :

$$\sigma(x) = \frac{1}{1 + e^{-x}}$$

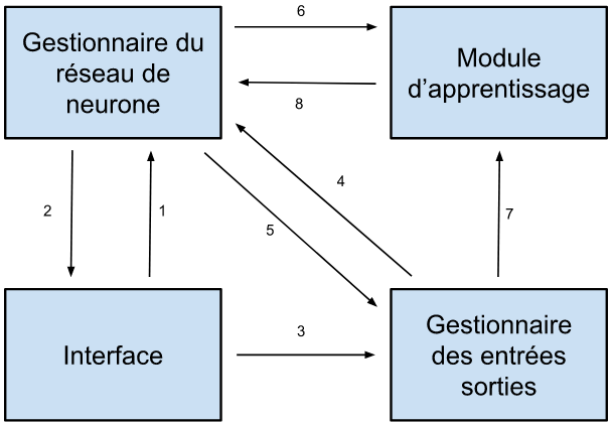
— *Gradient :* Le gradient d'une fonction (noté ∇f) est la direction de la plus forte pente par rapport à un point donné.

5 Exigences Fonctionnelles

5.1 Diagramme de cas D'utilisation



5.2 Organigramme et Liens entre Modules



Connections entre les modules :

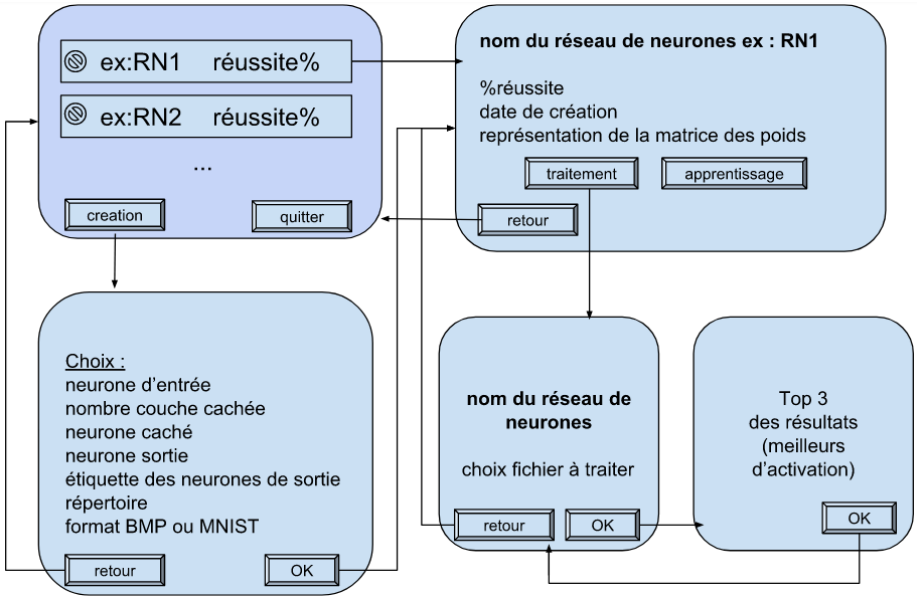
1. Lors de la création d'un réseau de neurone, l'utilisateur choisit les différents paramètres du réseau de neurone, ces informations sont donc envoyées au gestionnaire du réseau de neurone qui va créer le réseau. Pendant la phase d'apprentissage, il faut également spécifier au réseau de neurone qu'il faut envoyer la solution calculée au module d'apprentissage.
2. Après l'estimation d'une solution par le réseau de neurone, la solution est envoyée à l'interface. De plus, pour la visualisation de la méthode de fonctionnement du réseau, les matrices contenants les poids doivent également être envoyées à l'interface.
3. Lors de l'apprentissage ou du traitement d'image, l'interface demande au gestionnaire d'entrée/sortie de lire un fichier à un emplacement en particulier.
4. Après lecture d'un fichier, le contenu est envoyé au gestionnaire du réseau de neurone qui va en effectuer le traitement.
5. Afin de sauvegarder un réseau de neurone, les différentes matrices et vecteurs contenants les poids et les biais du réseau de neurone doivent être envoyés au gestionnaire d'entrées/sorties.
6. Lors de l'apprentissage, la solution calculée est envoyée au module d'apprentissage.
7. Afin de permettre l'apprentissage du réseau de neurone, la solution attendue doit être récupérée puis envoyée au module d'apprentissage.
8. Après avoir calculé les modifications à apporter aux poids et biais du réseau de neurone, celles-ci doivent être envoyé au gestionnaire du réseau de neurone afin que celui-ci les applique.

Tableau des fonctionnalités + estimation en coût en ligne de code (690 lignes) :

Gestionnaire du Réseau de Neurones (2 personnes) (155)	Apprentissage (2 personnes) (150)	Gestionnaire d'entrées sorties (1 personne) (75)	Interface (1 personne) (310)
-Formatage des informations reçu par	-Algorithme de rétropropagation (50)	-Récupération d'une image au format	-Définition de la taille des couches cachéés du

6 Exigences Non Fonctionnelles

6.1 Interface



6.2 Facilité D'utilisation et Facteur Humain

Le produit nécessitera une compréhension de la langue française. L'application aidera l'utilisateur à ne pas faire d'erreur en lui proposant des conseil et en utilisant des message de confirmation pour les actions tel que la suppression.

6.3 Facilité D'apprentissage

La prise en main par un utilisateur apte à l'utilisation de l'application sera quasi immédiate.

6.4 Rapidité D'exécution et Temps de Latence

6.4.1 Algorithme de Propagation

Fonctionnement :

1. Mettre les données d'entrées dans la première couche du réseau de neurone.
2. Calculer les activations des neurones de la couche suivante en suivant la formule :
$$A_j = \sigma(A^{j-1} * W^j + B^j)$$
avec A^j les activations de la couche 'j', σ la fonction sigmoïde, W^j les poids des liaisons entre les neurone de la couche 'j-1' et 'j' et B^j les biais des neurones de la couche 'j'.
3. Récupérer l'étiquette du neurone le plus actif de la couche de sortie.

Complexité : C = O(n) avec n le nombre de liaison entre neurone dans le réseau, ce qui est linéaire, mais le nombre de liaisons croît exponentiellement par rapport au nombre de neurone.

6.4.2 Algorithme de Rétro-Propagation

Fonctionnement :

1. Présentation d'un motif d'entraînement au réseau.
2. Comparaison de la sortie du réseau avec la sortie ciblée.
3. Calcul de l'erreur en sortie de chacun des neurones du réseau.
4. Calcul, pour chacun des neurones, de la valeur de sortie qui aurait été correcte.
5. Définition de l'augmentation ou de la diminution nécessaire pour obtenir cette valeur (erreur locale).
6. Ajustement du poids de chaque connexion vers l'erreur locale la plus faible.
7. Attribution d'un blâme à tous les neurones précédents.
8. Recommencer à partir de l'étape 4, sur les neurones précédents en utilisant le blâme comme erreur.

Complexité : Égale à celle de l'algorithme de propagation à un facteur près.

6.5 Fiabilité

Due à la nature même du réseau de neurone, la fiabilité de la prédiction sur la solution d'un problème après l'apprentissage du réseau de neurone ne pourra être de 100%. La fiabilité ne dépendra cependant pas de l'application mais de la complexité du problème à résoudre ainsi que des paramètres du réseau de neurone choisis par l'utilisateur.

7 Estimation des Coûts

7.1 Coût Humain

Minimum : 1/2 journée de travail seul par semaine 1/2 journée de travail en équipe par semaine

7.2 Répartition des tâches :

Réseau de Neurones	Apprentissage	Gestionnaire d'entrées sorties	Interface
Severine Selaquet Isaac Szulek	Thibaut Pepin Soumia Rezgui	Anthony Montigne	Arezki Slimani

8 Implémentations Supplémentaires Possibles

Afin d'améliorer les performance de l'application, il serait possible d'opter pour la version stochastique de la descente de gradient où la rétro-propagation est effectuée sur la moyenne de l'erreur de plusieur images et non plus sur l'erreur de chaque image. Il serait également possible d'implémenter un algorithme de multiplication matriciel (type Strassen) afin de rendre la calcul le plus commun de cette application plus rapide.

9 Idées pour les Futures Versions

Implémentation d'un réseau de convolutions en amont du réseaux de neurones et de l'apprentissage, qu'on appelle le CNN. Il s'applique à des techniques de filtre avec l'algorithme de Sliding Window afin qu'il découvre les objets dans une image complexe. Ce type de réseaux fait l'extraction des caractéristiques des images.

10 Portabilité

Le produit devra tourner de base sur un environnement Linux (debian) et doit pouvoir y être installable.

11 Conclusion

Il serait plus adéquat dutiliser le langage C plutôt qu'un autre langage pour la nature elle-même du langage qui est procédurale : pas d'interdépendance des différents éléments de notre programme, un seul algorithme complexe à développer ; utilisation des structures de données bien spécifiques (tableaux, matrices), mais aussi la conception de notre application est plutôt linéaire.

Nous citons aussi dautres points forts du langage :

- La simplicité : simple à utiliser (assimiler l'ensemble de ses mécanismes)
- La puissance : langage universel, convenant aussi bien à la programmation système qu'au codage dalgorithmes et à la représentation de structure de données complexes, au développement dinterface ou au calcul numérique.
- La portabilité : portable facilement sur des systèmes différents.
- La souplesse : nimpose pas de cycle particulier.
- l'efficacité : grâce aux variétés de ses expressions ; il permet d'optimiser à la main les sources de programmes.

Pour conclure, notre application permettra à l'utilisateur de créer et de choisir un ou plusieurs réseau de neurones et d'y traiter une image ou d'y réaliser un apprentissage en utilisant un répertoire d'images variées que l'utilisateur aura choisi préalablement.