Université Paris 8

BCI : Detection et classification des activités cérébrales

Rapport I projet tuteuré

Auteurs :
Nizar ABAK-KALI
11290569
Alexis
ZURAWSKA
12310497

Tuteur :
M. Larbi BOUBCHIR

2 juin 2016

Table des matières

Table des figures

Introduction

Durant notre première année de master en informatique à l'université Paris 8 Vincennes - Saint-Denis, nous devions développer des applications au cours d'un module intitulé "Projets tuteurés". Ce travail devait être effectué sur toute l'année et faire l'objet d'une présentation en plus de ce rapport-ci. Nous devions choisir une thématique parmi celles proposées. Ensuite, notre tuteur attitré nous attribuait un projet en nous expliquant ce qu'il attendait de nous. En ce qui nous concerne, nous devions développer un réseau de neurones capable de reconnaître différents d'activité d'une personne de par son activité cérébrale. C'est après vous avoir présenté certaines généralités touchant au BCI, puis sur l'utilité et l'implémentation des réseaux de neurones et des interfaces cerveaux-machines que nous exposerons l'état de l'art en ce qui concerne la détection et la classification d'activités cérébrales. Puis, nous expliquerons le fonctionnement du système. Ensuite, nous détaillerons l'architecture de l'application. Enfin nous conclurons sur les objectifs atteints et les prévisions futures .

Première partie

Introdution : généralités et contexte

1 BCI

BCI : Brain Computer Interface (ou Interface Cerveau-Machine en français). Aussi appelée Interface Neuronale Directe (abrégée IND), il s'agit d'une interface de communication directe entre un cerveau et un composant externe (généralement un ordinateur).

2 EEG

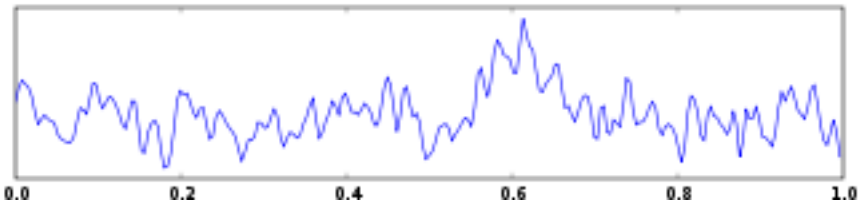


FIGURE 1 – ci-dessus une seconde de signal EEG

EEG ou électroencéphalographie est une technique de mesure de l'activité électrique du cerveau mesurée à l'aide d'électrodes placées sur le cuir chevelu. L'EEG est un examen indolore est non-invasif comparativement à l'iEEG(électroencéphalographie intracrânienne) qui, elle, place les électrodes sous la surface du crâne. le signal EEG obtenu est la résultante de la sommation d'un potentiel d'action post-synaptique synchrone issu d'un grand nombre de neurones.

3 Généralités

Ici, certains concepts clés doivent être définis pour comprendre en quoi consiste notre projet :

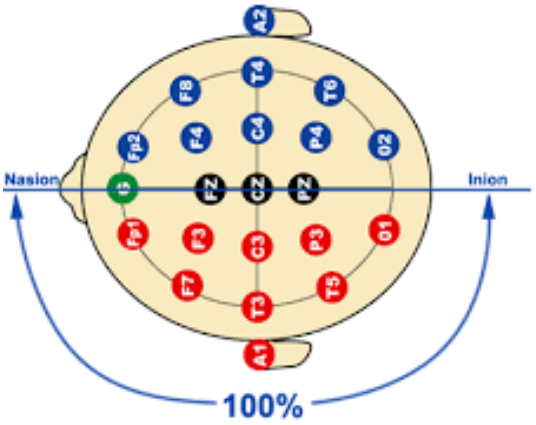


FIGURE 2 – position des capteurs

3.1 Les réseaux de neurones

Les réseaux de neurones sont un modèle de calcul représentant schématiquement les réseaux de neurones biologiques. C'est à celui-ci que l'on va imposer l'apprentissage d'un échantillon de données. Pour cela, nous allons utiliser l'algorithme du gradient, le gradient étant, en mathématiques, un vecteur qui représente la variation d'une fonction en fonction de la variation de ses paramètres. Tout comme nous, le réseau de neurones n'est pas parfait et fera des erreurs lors de son apprentissage. Il faudra donc utiliser la rétropropagation du gradient qui est un algorithme servant à corriger les erreurs du réseau de neurones pour qu'il ne les reproduise pas.

3.2 Notions en lien avec BCI

Certaines notions sont importantes à connaître lorsque l'on s'intéresse au BCI, entre autres le SSVEP et le P300. Ce sont tous les deux des réponses naturelles à une stimulation visuelle. Pour le SSVEP, la réponse est détectée lors d'une stimulation visuelle lorsque la rétine est excitée par un stimulus visuel dont la fréquence d'affichage est entre 3.5Hz et 75Hz. Le P300, par contre, est une réponse qui apparaît 300ms après stimulation visuelle.

4 Contexte

Les réseaux de neurones sont des composants logiciels informatiques permettant de représenter de manière aproximative les réseaux de nehruones humains.

Ceux-ci permettent, via une méthode d'apprentissage basée sur le gradient d'erreur que l'on modifie jusqu'à ce que la réponse donnée par le réseau de neurones soit la bonne.

Dans le cadre de ce projet, nous devions développer un réseau de neurones afin de contrôler automatiquement les activités cérébrales d'un sujet. Ainsi, le système pourrait nous dire grâce au signal EEG du patient quelle activité il est en train de faire ou bien quel est son état. Notre système pourrait donc par exemple connaître l'activité d'un malade d'Alzheimer telle que :

- la lecture ;
- l'attention sur un film ;
- l'écriture ;
- la discussion ;
- le repos.

Ceci est un exemple d'utilisation de ces réseaux de neurones. Or, il se trouve que nous pouvons les utiliser pour des choses plus sofistiquées telles que :

- la détermination de facteurs dominants dans le déclanchement d'une maladie afin de mieux la comprendre et de cibler la recherche afin d'essayer de trouver un traitement.
- l'amélioration de la qualité de vie de personnes lourdement handicapées : traitement des signaux représentant les pensées de la personne pour qu'elle puisse bouger ses bras ou communiquer (si elle ne le peut pas verbalement)
- la possibilité, pour des personnes souffrant du syndrôme de l'enfermement, de pouvoir dialoguer avec les autres par la pensée grâce à un écran où l'on voit toutes les lettres et où le mot et/ou la phrase que pense le malade s'affiche.
- dans notre cas, peut-être que ce projet permettrait de davantage retarder l'évolution de la maladie d'Alzheimer en essayant de stimuler la pensée des patients afin qu'ils puissent conserver leur autonomie un maximum par le procédé du traitement du signal.

Deuxième partie

Etat de l'art

5 SSVEP

SSVEP est l'abréviation de Steady State Visually Evoked Potential, traduite en français par l'état potentiel stable visuellement évoqué. Cela désigne, en neurologie, un ensemble de signaux représentant des réponses naturelles à des stimulus visuels à des fréquences spécifiques. La fréquence dépend de la puissance en Hz du stimulus visuel reçu par la rétine, celui-ci variant entre 3,5 et 75 Hz. La réponse se traduit par une activité électrique générée par le cerveau de même fréquence. Cette réponse est directe car elle se produit dés que le stimulus visuel est présenté. Cette technique est très utilisée en même temps que l'électroencéphalographie en ce qui concerne la vision. Elle est beaucoup appréciée pour son absence d'artefacts qui sont des effets artificiels parfois indésirables en plus du fait que les bruits alentours ne soient pas pris en compte.

6 P300

Le P300 est un potentiel évoqué (modification du potentiel électrique produite par le système nerveux en réponse à une simulation externe (son, image, etc...) ou interne attention, préparation motrice, etc...), et plus généralement une onde positive et d'amplitude 300 ms. Cela signifie que la réponse n'est pas directe et qu'elle n'intervient que 300 ms après que la simulation ait eu lieu. Cette technique est notamment beaucoup utilisée dans l'électroencéphalographie. On distingue deux sous-types de P300 au sein de cette onde P300 aussi notée P3 :

- la P3a, généralement provoquée par un stimulus lors d'un effet de surprise ;
- la P3b où le sujet se trouve face à un stimulus non prévisible où il doit apporter une réponse (prise de décision, action réalisée grâce à la mémoire, etc...).

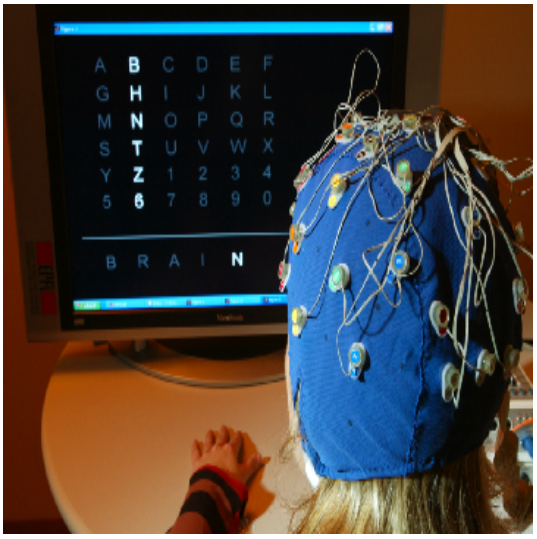


FIGURE 3 – Photo d'un sujet en face d'une interface BCI

7 BCI

Ce procédé consiste en fait à restaurer, au moins partiellement des facultés perdues, voire non connues (dans le cas d'un handicap de naissance). C'est le cas notamment d'une personne ayant perdu totalement la vue à qui on a implanté un tel procédé au niveau de son cortex visuel, ce qui lui a permis de percevoir de nouveau la lumière. Si ces procédés ne sont pas miraculeux pour le moment, ils peuvent également permettre d'effectuer des actions par l'intermédiaire de la pensée. Ainsi, des personnes ont pu écrire sur un écran d'ordinateur ou déplacer le curseur d'une souris en imaginant simplement l'action de le faire. D'autres ont pu déplacer un bras robotisé pour qu'il leur ramène quelque chose par exemple.

Ce procédé peut être unidirectionnel (la machine envoie des données au cerveau ou le contraire mais pas les deux en même temps) ou bidirectionnel. Ces interfaces neuronales directes présentent toutefois plusieurs limites :

- chacune d'elles ne fait qu'une tâche précise et non plusieurs ;
- elles étaient initialement développées dans un but médical rendant l'accès difficile au grand public ;
- à cela s'ajoute le fait que les populations défavorisées n'auront pas les moyens de se procurer de tels équipements, le prix étant très élevé.

8 Applications Médicales

Les applications du programme que nous avons développé peut servir pour des études très variées sur plusieurs types de pathologies telles que la maladie d'Alzheimer ou encore les crises d'épilepsies que l'on peut prévoir et minimiser notamment, pour ne citer que ces deux exemples. On peut aussi l'utiliser comme moyen ludique pour détecter la concentration des élèves en classe en temps réel. On pourrait aussi, à l'avenir, contrôler notre PC à l'aide de notre seule pensée au moyen d'un casque EEG.

Troisième partie

Classification automatique des activités cérébrales

9 Schéma Général

cf ??, page ??.

10 Pré-Traitement du signal

Cette étape consiste en un traitement sur le signal EEG obtenu afin de réduire les "bruits"(ou artefact) crées par les activités du sujet lors de l'acquisition du signal, telles : la position du sujet(sachant que un EEG standard se fait en position allongée ou assise avec un sujet relaxé), les mouvements des muscles du visage (exemple : clignement des yeux). Ainsi on applique une transformée de Fourrier au signal afin d'éliminer les bruits .

11 Extraction des caractéristiques

Cette étape est très importante car elle permet de déduire des caractéristiques notables du signal, elle se déroule lors de la phase d'apprentissage du réseau de neurones. Ainsi, un signal sera caractérisé par un vecteur de caractéristiques. On peut déduire plusieurs types de caractéristiques : temporelles, fréquentielles, tempo-fréquentielles.

12 Classification par réseaux de neurones

Pour la prise de décision nous avons choisi d'utiliser un réseau de neurones SOM, ainsi le signal sera classifié selon différents types de classes. Ces classes seront les activités du sujet que l'on veut détecter.

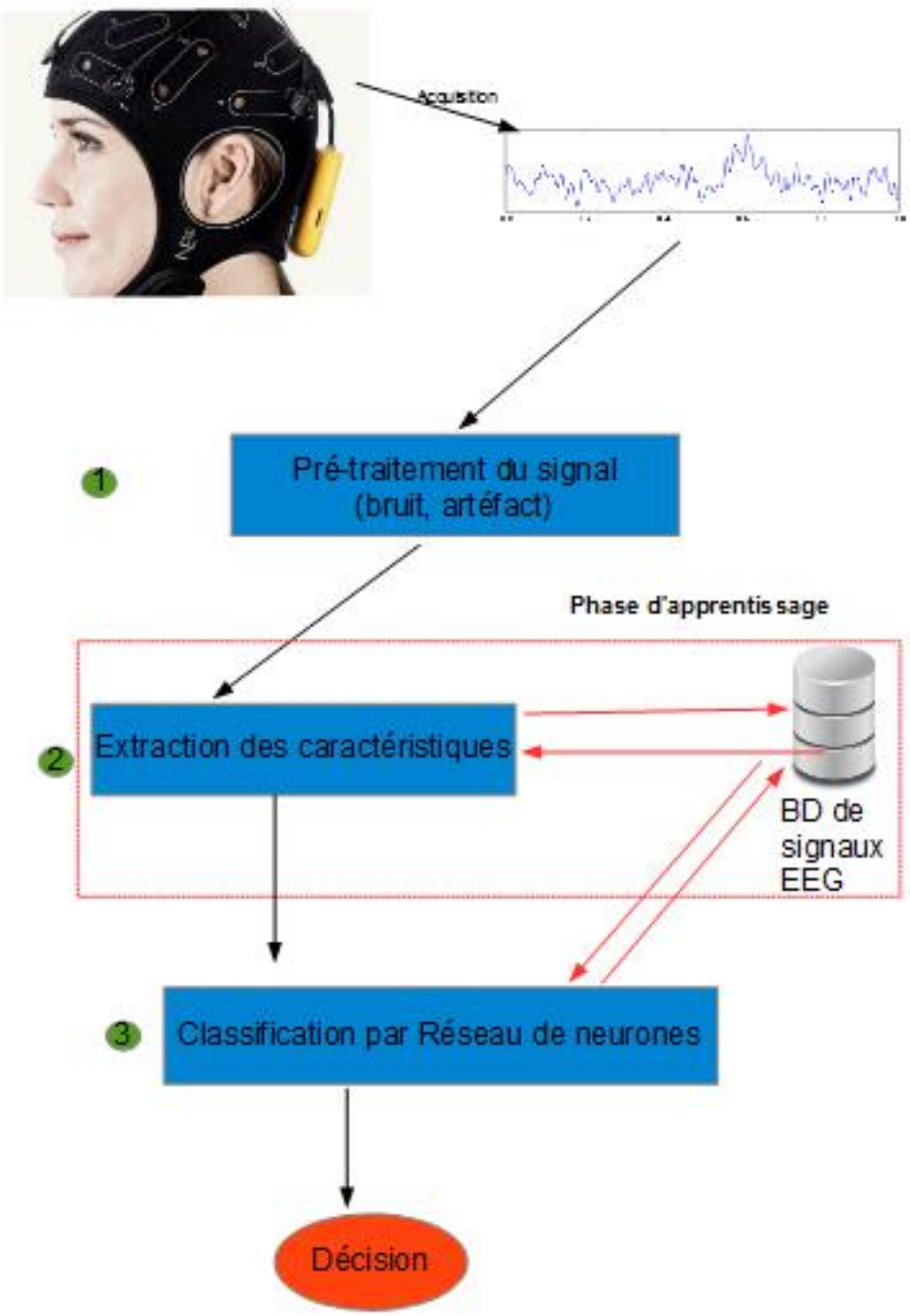


FIGURE 4 – Organigramme du fonctionnement générale de l'application

Quatrième partie

Architecture de l'application

13 Composants de l'application

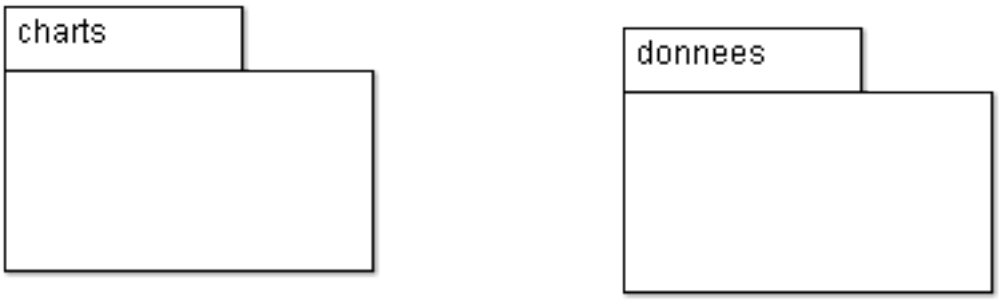


FIGURE 5 – Packages composant l'application

L'application est composée de six packages : eeg, math, neuron, charts, donnees et graphe. Nous allons donc décrire le contenu de chacun de ces packages ainsi que le rôle des classes qui les composent :

- eeg : contient la classe EEG_time_signal qui contiendra les fonctions concernant le traitement du signal en fonction des données reçues et traitées ;
- math : contient toutes les fonctions mathématiques avec la classe ComplexNumber permettant de créer et gérer les nombres complexes, la classe ComplexArray qui permet de stocker les valeurs réelles et imaginaires des nombres complexes, la classe Hilbert qui contient les fonctions concernant les espaces d'Hilbert (extension des espaces euclidiens

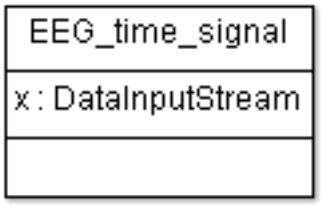


FIGURE 6 – Classe EEG_time_signal

à des dimensions finies quelconques ou infinies) et enfin la classe Fft qui contient les fonctions en rapport avec la Transformée Rapide de Fourrier(ou Fast Fourrier Transform) ;

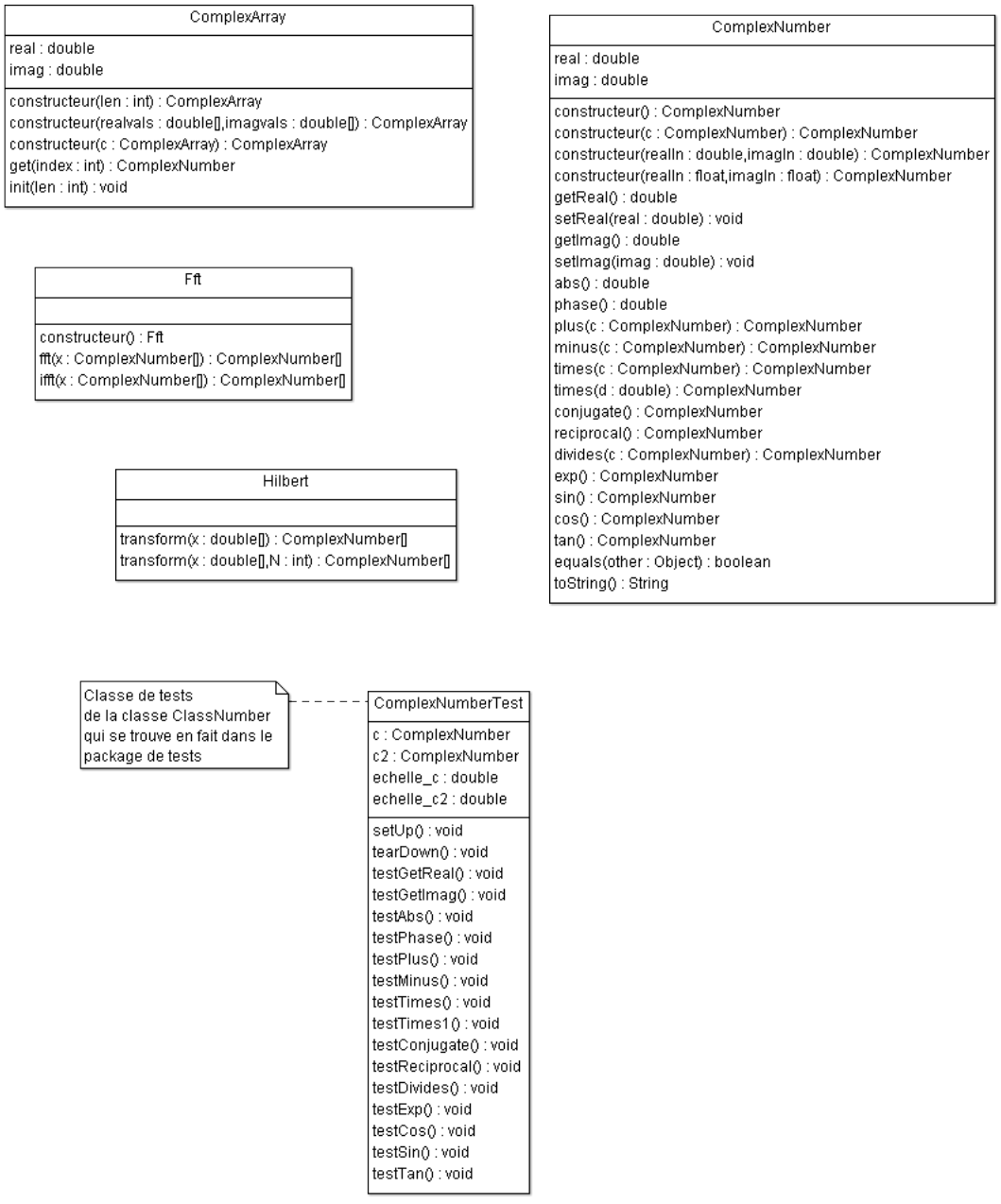
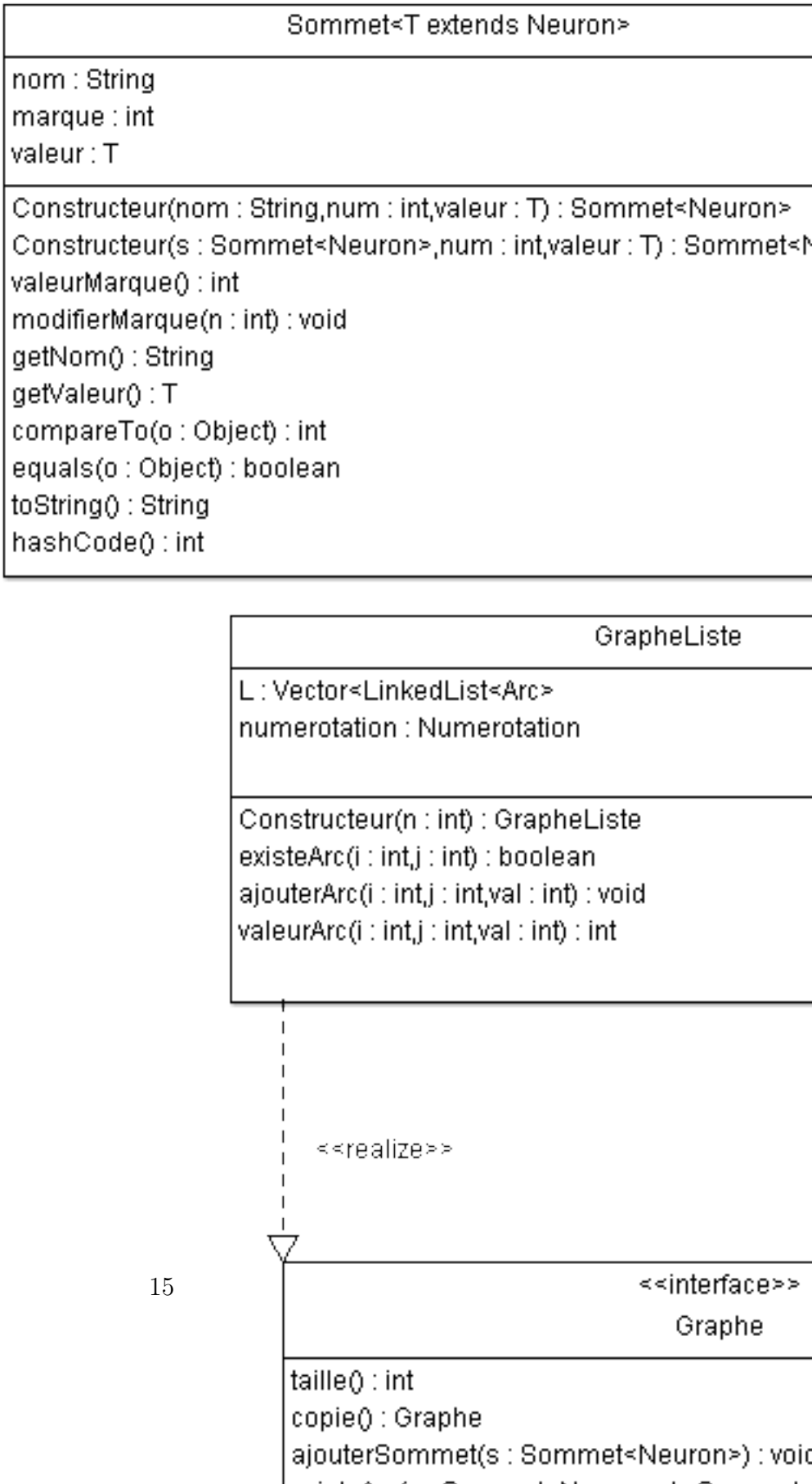


FIGURE 7 – Classes pour les calculs mathématiques

- neuron : contient la classe CaracteristiqueTemporel qui contient toutes les fonctions ayant attrait aux caractéristiques temporelles d'un neurone pour le réseau qui sera par la suite créé. Ces caractéristiques temporelles regroupent notamment la moyenne des données, la variance, etc... ; contient également la classe neuron qui définit les caractéristiques d'un neurone : possède une valeur, une valeur, une donnée des fichiers fournis et un gradient d'erreur qui doit être de 0 lorsque la réponse donnée par le réseau est bonne est bonne ; la classe LongueurDifferenceException permet de gérer les différences de taille des tableaux passés en paramètres concernant les fonctions appelées dans la classe NeuronNetwork ; la classe NeuronNetwork représente la constitution et la forme (graphe) du réseau de neurones ; la classe NeuronNetworkLearnAndResults, quant à elle, permet de tester notre réseau de neurones sur les fichiers de données fournies.
- charts : contient la clase Graphique qui permet de tester l'affichage d'un graphique représentant l'un des signaux d'un des fichiers de données fournies grâce à la librairie Open Source java JFreeCharts.
- donnees : contient la classe LireCVS permettant de récupérer un ou des signaux des fichiers de données fournis pour les stocker en mémoire afin de pouvoir travailler avec.

- graphe :


Sommet<T extends Neuron>
nom : String
marque : int
valeur : T
Constructeur(nom : String,num : int,valeur : T) : Sommet<Neuron>
Constructeur(s : Sommet<Neuron>,num : int,valeur : T) : Sommet<N
valeurMarque() : int
modifierMarque(n : int) : void
getNom() : String
getValeur() : T
compareTo(o : Object) : int
equals(o : Object) : boolean
toString() : String
hashCode() : int
GrapheListe
L : Vector<LinkedList<Arc>
numerotation : Numerotation
Constructeur(n : int) : GrapheListe
existeArc(i : int,j : int) : boolean
ajouterArc(i : int,j : int,val : int) : void
valeurArc(i : int,j : int,val : int) : int
<<realize>>
<<interface>>
Graphe
taille() : int
copie() : Graphe
ajouterSommet(s : Sommet<Neuron>) : voic
existeArc(s : Sommet<Neuron> t : Sommet<

0 – Classes permettant de définir la forme et la constitution du réseau de neurones

contient les classes permettant de définir la structure du réseau de neurones sous forme de graphe (sommet, arc, forme du graphe, etc...)

14 Explication Technique

15 Idéés

Nous avons eu l'idée, pour ce projet, de créer un "classifieur" représenté par un réseau de neurones afin d'obtenir des résultats à partir d'un flux de données que nous stockerons en mémoire dans un tableau. Chaque donnée sera représentée par un vecteur. Pour chaque vecteur, chacune de ses dimensions représentera une caractéristique du signal EEG du sujet que nous chercherons à reconnaître (le tout afin de reconnaître les états tels : lecture, repos, regarder un film, etc...). Nous sommes partis de ces idées là car, lors de nos lectures sur les interfaces cerveau-machine, il s'est avéré que ce mode de traitement du signal est très utilisé .

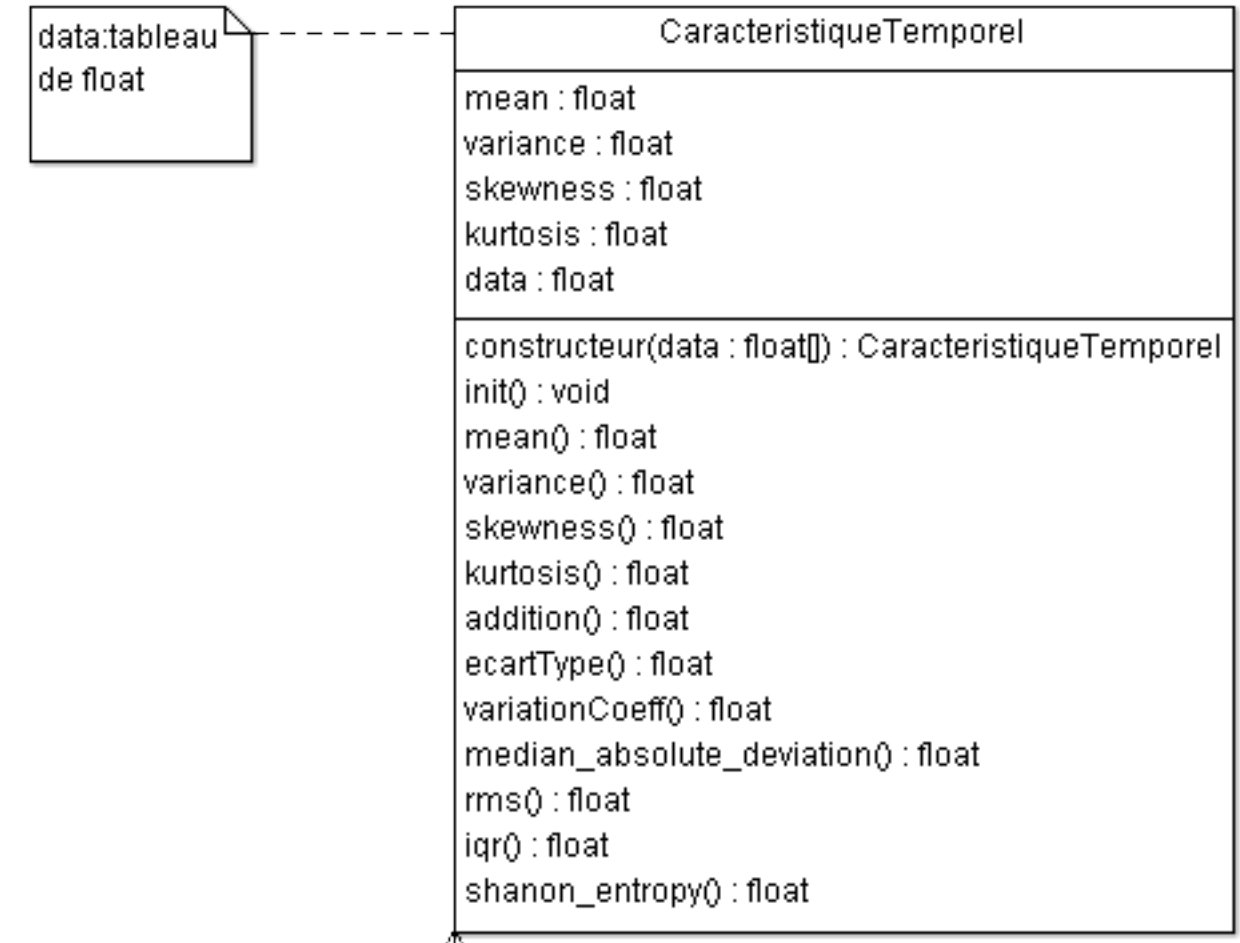
16 Outils

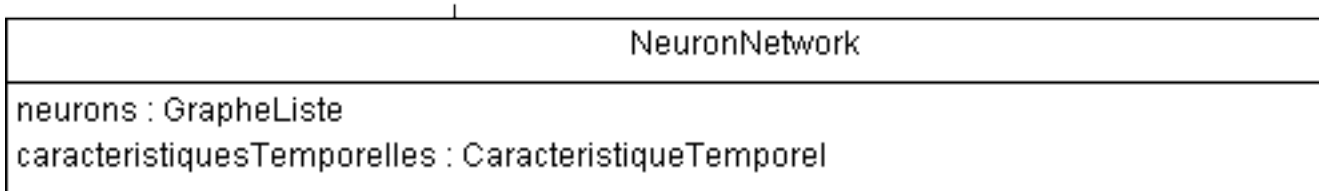
Nous avons développé notre application au moyen du langage Java et nous avons utilisé l'IDE Intellij qui est très pratique car très ergonomique et très intuitif. De plus nous avons utilisé Git comme gestionnaire de versions, le code étant hébergé sur les serveurs de Github. De surcroît, nous avons utilisé JavaDoc pour générer une documentation du code. Par ailleurs, nous avons inclus la librairie JFreeCharts pour réaliser des graphiques sans les contraintes de la bibliothèque Swing pour créer des interfaces.Nous avons LaTex afin de rédiger plus facilement nos rapports . Par ailleurs, nous avons utilisé le logiciel argoUML pour réaliser le diagramme de classes de l'application et OpenOffice Draw pour réaliser l'organigramme.

17 Algorithmes

Par rapport à notre avancement courant, les algorithmes les plus intéressants à présenter sont la fft [1] et la transformée de Hilbert qui utilise la fft .
La fft est un algorithme de calcul de la transformation de Fourrier discrète il nous sert lors de la partie de post-traitement du signal, mais aussi dans la transformée d'Hilbert.
La transformée d'Hilbert est utilisée dans le traitement du signal pour passer le signal réel dans le domaine complexe d'où la classe ComplexNumber. Cette transformée est utilisée lors des calculs des caractéristiques temporelles .

1. Fast Fourrier Transform


data:tableau
de float
CaracteristiqueTemporel
mean : float
variance : float
skewness : float
kurtosis : float
data : float
constructeur(data : float[]) : CaracteristiqueTemporel
init() : void
mean() : float
variance() : float
skewness() : float
kurtosis() : float
addition() : float
ecartType() : float
variationCoeff() : float
median_absolute_deviation() : float
rms() : float
iqr() : float
shanon_entropy() : float


NeuronNetwork
neurons : GrapheListe
caracteristiquesTemporelles : CaracteristiqueTemporel

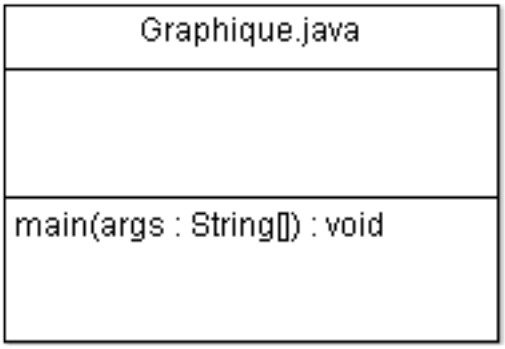
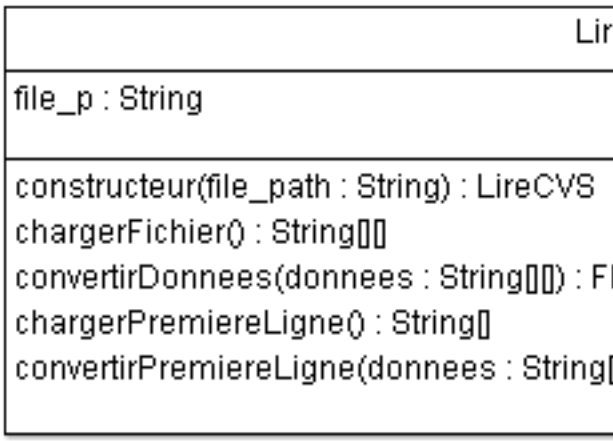


FIGURE 9 – Classe Graphique


Lir
file_p : String
constructeur(file_path : String) : LireCVS
chargerFichier() : String[][]
convertirDonnees(donnees : String[][]) : Fl
chargerPremiereLigne() : String[]
convertirPremiereLigne(donnees : String[

Cinquième partie

Conclusion

18 Objectifs atteints

Toutes les fonctions mathématiques sur les complexes ont été développées ainsi que les fonctions concernant l'espace d'Hilbert et la Fast Fourrier Transform. Par ailleurs, nous avons terminé la classe de récupération du fichier qui sera modifiée une fois en possession de la base de données sur laquelle nous travaillerons. Nous avons également développé une classe permettant d'afficher les signaux pour voir ce qu'ils donnent. Enfin, nous avons développé le réseau de neurones ainsi que sa structure.

19 Objectifs restants et prévisions

Pour conclure, nous pouvons dire que ce projet est très intéressant car il nous a permis de découvrir des domaines concrets insoupçonnés où l'informatique pouvait intervenir afin de faciliter la vie, notamment des personnes lourdement handicapées. Nous n'avons toutefois pas mené à bien nos objectifs car nous nous sommes mal organisés. Du coup, nous avons tout simplement essayé de mettre en oeuvre le réseau de neurones et son fonctionnement sur les données alors que nous aurions du, dans l'idéal, axer notre projet sur la partie purement signal. Nous espérons pouvoir vous fournir un réseau de neurones fonctionnels, quelques petits réglages restant à faire.

Sixième partie

Références

Dans cette partie nous exposons les différentes sources que nous avons trouvées utiles lors de la première partie du projet tuteuré .

- "ON THE USE OF TIME-FREQUENCY FEATURES FOR DETECTING AND CLASSIFYING EPILEPTIC SEIZURE ACTITIES IN NON-STATIONNARY EEG SIGNALS" - Larbi Boubchir, Somaya Al-Maadeed et Ahmed Bourdane.
- https://en.wikipedia.org/wiki/Feedforward_neural_network;
- https://fr.wikipedia.org/wiki/Gradient;
- https://fr.wikipedia.org/wiki/Algorithme_du_gradient;
- https://fr.wikipedia.org/wiki/R%C3%A9tropropagation_du_gradient;
- https://takinginitiative.wordpress.com/2008/04/03/basic-neural-network-tutorial-theory/
- https://takinginitiative.wordpress.com/2008/04/23/basic-neural-network-tutorial-c-implementation-and-source-code/
- https://fr.wikipedia.org/wiki/Interface_neuronale_directe
- http://www.inserm.fr/thematiques/technologies-pour-la-sante/dossiers-d-information/l-interface-cerveau-machine-ou-agir-par-la-pensee
- http://www.internetactu.net/2012/11/21/interfaces-cerveau-machines-defis-et-promesses/
- https://github.com/marytts/marytts/blob/master/marytts-signalproc/src/main/java/marytts/util/math/FFTMixedRadix.java
- http://www.tangentex.com/FFT.htm
- http://www.javaworld.com/article/2074798/build-ci-sdlc/chart-a-new-course-with-jfreechart.html
- http://introcs.cs.princeton.edu/java/97data/FFT.java.html
- https://moodle.umons.ac.be/mod/resource/view.php?id=11577
- http://www.java2s.com/Code/Java/Chart/JFreeChartLineChartDemo1.htm
- http://jcharts.sourceforge.net/samples/scatter.html
- https://en.wikipedia.org/wiki/Steady_state_visually_evoked_potential
- https://fr.wikipedia.org/wiki/P300

- http://www.enseignement.polytechnique.fr/informatique/INF431/GrapheX/docgrapheX.pdf
- http://thierry-leriche-dessirier.developpez.com/tutoriels/java/afficher-graphe-jfreechart-5-min/
- https://openclassrooms.com/forum/sujet/ouvrir-et-lire-un-fichier-csv-56738
- http://thierry-leriche-dessirier.developpez.com/tutoriels/java/charger-donnees-fichier-csv-5-min/