Rapport Final Travaux Personnel Encadré

Thème : " Compréhension par l'exploration et la visualisation des réseaux de neurones "

Par : Mezatio Eric Papain
Encadré par : Serge Stinckwich
Spécialité SIM(Système Intelligent et Multimédias)
Année Académique 2018/2019

Plan de travail

Chapitre 1: Analyse

Introduction

Domaine d'application et problème visé par le sujet

Chapitre 2: Recherches Bibliographiques

Introduction

I- Les Réseaux de Neurones

1. **Perceptron vs Neuron Artificielle**
2. **Réseaux de Neurones**

II- Visualisation comme aide à la compréhension des réseaux de neurones

1. **Pourquoi?**
2. **Qui?**
3. **Qu'est ce qu'on visualise?**

RÉFÉRENCES

Chapitre 3: Solution proposée

Introduction

Problème

1- Qu'allons-nous faire ?

2- Pourquoi allons-nous faire cela ?

3- Comment allons-nous faire cela ?

a- Les paramètres appris par notre modèle

b- Les neurones

c- Les différentes valeurs des activations

d- Les variations des poids pendant l'entrainement

4- Qu'est-ce qui existe pour nous aider ?

RÉFÉRENCES

Chapitre 4: Réalisation

Introduction

I. Environnement de travail

II. Expérimentations

III. Comparaison

Conclusion & Perspectives

Chapitre 1: Analyse

Thème: Compréhension par l'exploration et la visualisation des réseaux de neurones

Ce présent document explique en quelques lignes l'analyse faite sur le thème intitulé :
Compréhension par l'exploration et la visualisation des réseaux de neurones

Introduction

L'Informatique est définie comme étant la science du traitement automatique et rationnelle de l'information. Cette définition n'a pas cessé d'évoluer et on pense aujourd'hui à la science du traitement non seulement automatique, mais du traitement basé sur l'apprentissage et la prise de décision autonome par des ordinateurs. À ce sujet, la construction logicielle s'appuie sur plusieurs approches dont les plus importantes sont les suivantes : l'approche algorithmique et l'approche basée sur la connaissance. La seconde est celle de l'intelligence artificielle donc l'une des branches très en croissance est le Deep learning ou en d'autres termes apprentissages profond qui est celle qui nous intéresse tout le long de notre document. En effet, l'apprentissage profond est un ensemble spécifique de techniques issues du domaine plus large de l'apprentissage automatique ou encore machine learning (ML) qui se concentre sur l'étude et l'utilisation de réseaux de neurones artificiels profonds pour apprendre des représentations structurées de données analysées. En effet, cette représentation utilise dans la majorité des cas les réseaux de neurones pour représenter les données à analyser, ces réseaux de neurones ont connu un très grand développement au cours des dernières décennies. Cette technique abordant plusieurs domaines différents tels que la finance, la médecine, la production industrielle, la géologie, la physique et bien d'autres domaines encore. Ainsi donc, l'intérêt accordé à ces réseaux de neurones vient du fait de leurs grands succès connu dans la résolution de gros problèmes au détriment des autres techniques d'analyse comme des analyses statistiques simples qui se font manuellement par l'Homme. Ceci est dû à son caractère très polyvalent, à sa simplicité d'utilisation.

Malheureusement, nous ne comprenons pas pourquoi ils fonctionnent si bien et comment ils parviennent à représenter des informations et à prendre des décisions, ni comment ils pourraient être améliorés. De plus, la construction de ces réseaux de neurones est une activité complexe, car elle nécessite la construction d'une architecture très complexe à évaluer.

Pour remédier partiellement à ce problème de compréhension, de complexité des réseaux de neurones, plusieurs travaux ont été réalisés auxquels nous pouvons mentionner l'utilisation des techniques et outils de visualisation (Visual Analytics) qui permettent de comprendre leurs fonctionnement afin d'améliorer leurs performances.

Ainsi donc, en s'intéressant particulièrement aux travaux proposés pour la compréhension des réseaux de neurones par la visualisation, nous constatons qu'ils font apparaîtrent plusieurs catégories d'utilisateurs dans leur cycle de vie (développeurs, end-users, ...), et face à cela notre préoccupation est celle de savoir : **<<Comment pouvons-nous concilier ces différents points de vues afin de permettre la compréhension d'un réseau de neurones dans ses différentes dimensions ?>>**

Domaine d'application et problème visé par le sujet

Notre analyse est orientée dans le domaine de l'intelligence artificielle plus précisément celui de l'apprentissage en profondeur, particulièrement celui de la **compréhension des réseaux de neurones par l'exploration et la visualisation.**

Ainsi donc, les prochaines parties seront constitué comme suit:

- Pour l'état de l'art, nous nous intéresserons à l'article de : ***Hohman, Fred Matthew and Kahng, Minsuk and Pienta, Robert and Chau, Duen Horng, « Visual Analytics in Deep Learning: An Interrogative Survey for the Next Frontiers*** » qui expose particulièrement aux questionnements sur la visualisation des réseaux de neurones, elle nous permettra de mieux comprendre de quoi il s'agit non seulement, mais aussi de comment aborder ce thème sur la

visualisation des réseaux de neurones;

De même, nous ferons une étude connexe sur des articles en référence de ce dernier.

- Afin de mettre à jour les informations concernant le domaine, nous serons amenés à compléter cet état de l'art à l'aide des derniers travaux publiés.

- Nous étudierons par la suite quelques concepts des réseaux de neurones afin de comprendre leurs implémentations afin d'y mener une étude expérimentale optimale.

Chapitre 2: Recherches Bibliographiques

Introduction

Le deep learning a récemment connu un développement rapide et a fait l'objet d'une attention considérable en raison de ses performances sur des problèmes difficiles précédemment penser. En effet, le défi se trouve au niveau de traduire un problème facilement compréhensible par l'homme à un problème compréhensible par la machine. Par exemple, un Homme qui voit un chat saurait bien le décrire même si celui-ci manque des parties de son corps, de même, un Homme est également capable de faire la différence entre un enfant, un bébé, une vieille personne, mais comment amener la machine à savoir faire des distinctions pareilles ? Pour résoudre ce problème, ou pour répondre en partie à cette question, nous avons eu en 1945 les recherches d'Alan Turing qui portait sur l'apprentissage automatique qui énonçait déjà quelques voix de solutions pour se type de problèmes conciliant le rapprochement du raisonnement Homme-Machine.

Depuis quelques années, déjà, on assiste à une très grande avancé scientifiques en informatique plus précisément dans le domaine de l'Intelligence Artificielle donc le point fort repose sur le deep learning qui tire toute son énergie et sa puissance dans les réseaux de neurones artificiels qui sont inspirés totalement du fonctionnement du neurone humain.

Ces réseaux de neurones réussissent à atteindre de très grandes performances dans des problèmes qui trouvaient jusqu'ici difficilement des solutions ce avec un temps très rapide et optimal. Mais la difficulté réside au niveau de la compréhension de leur fonctionnement, car l'Homme dans un souci de sécurité et même aussi de s'assurer des bonnes décision prise par la machine se doit de comprendre comment fonctionne celui-ci. Plein de travaux, on été réalisé dans ce sens et dans cette partie de notre travail nous présenterons premièrement quelques terminologies sur les réseaux de neurones, ensuite, nous présenterons quelques travaux existant dans le domaine de compréhension du fonctionnement des réseaux de neurones ensuite en ce qui nous concerne, quelques travaux existant pour la compréhension des réseaux de neurones, ensuite proposer des solutions allant dans le même sens, et enfin donner quelques avantages de notres solutions proposées.

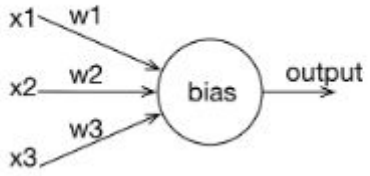
I- Les Réseaux de Neurones

Les réseaux de neurones, fabriqués de structures cellulaires artificielles, constituent une approche permettant d'aborder sous des angles nouveaux les problèmes de perception, de mémoire, d'apprentissage et de raisonnement. Grâce à leur traitement parallèle de l'information et à leurs mécanismes inspirés des cellules nerveuses (neurones), ils infèrent des propriétés émergente permettant de solutionner des problèmes jadis qualifiés de complexes. ***Marc Parizeau, "RÉSEAUX DE NEURONES GIF-21140 et GIF-64326 "Automne 2004, pp. 1.***
Les réseaux de neurones servent aujourd'hui à toutes sortes d'applications dans divers domaines. Par exemple, on a développé un auto-pilote pour avion, ou encore un système de guidage pour automobile, on a conçu des systèmes de lecture automatique de chèques bancaires et d'adresses postales, on produit des systèmes de traitement du signal pour différentes applications militaires, un système pour la synth ` ese de la parole, des réseaux sont aussi utilisés pour bâtir des systèmes de vision par ordinateur, pour faire des prévisions sur les marchés monétaires, pour évaluer le risque financier ou en assurance, pour diff ´ erents processus manufacturiers, pour le diagnostic médical, pour l'exploration pétrolière ou gazière, en robotique, en télécommunication, et j'en passe ! Bref, les réseaux de neurones ont aujourd'hui un impact considérable et, il y a fort à parier, que leur importance ira grandissant dans le futur. ***Marc Parizeau, "RÉSEAUX DE NEURONES GIF-21140 et GIF-64326 "Automne 2004, pp. 4.***

Pour y arriver à ces réseaux de neurones, plusieurs étapes, on été réalisé pour rapprocher au maximum la construction des réseaux de neurones artificiels à celui du cerveau humain notamment le perceptron, le neurone et le réseau de neurones.

1) Perceptron vs Neuron Artificielle

Dans son livre intitulé Artificial Intelligence, Alexandre Bergel définit le perceptron comme étant une sorte de neurone artificiel qui modélise le comportement d'un neurone biologique et qui produit une sortie pour un ensemble fourni de valeurs d'entrée qu'il représente comme le montre la figure ci-dessous:



Dans cet ouvrage il précise que le perceptron accepte 1 ou plusieurs valeurs numériques en tant qu

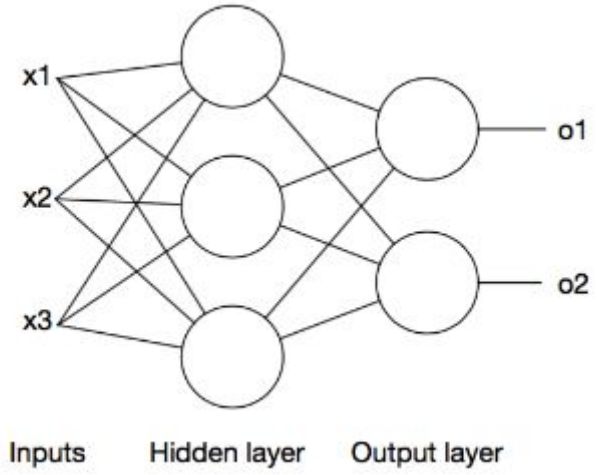
entrées et produit une valeur numérique en sortie ainsi donc, le perceptron fonctionne sur des nombres, ce qui signifie que les entrées et les sorties sont des valeurs numériques.

Sur la figure représentée plus haut issu du même article, Les entrées sont représentées par des flèches entrantes situées à gauche du cercle central et la sortie par une flèche sortante à droite de celui-ci. Dans la figure, notre perceptron à trois entrées, notées *x1* , *x2* et *x3* . Toutes les entrées n'ont pas la même importance pour le perceptron. Par exemple, une entrée peut être plus importante que d'autres. La pertinence d'une entrée est exprimée à l'aide d'un poids (également une valeur numérique) associé à cette entrée. Dans notre figure, l'entrée *x1* est associée au poids w1 , x2 au poids w2 et x3 à w3.
En plus de la valeur d'entrée pondérée, un perceptron nécessite un biais , une valeur numérique faisant office de seuil. Nous notons le biais comme b .

Cependant, le perceptron a de sérieuses limitations, qui inciteront à formuler un neurone artificiel plus robuste, appelé neurone sigmoïde, Une chaîne de perceptrons ne peut pas apprendre ce qui est possible avec le neurone sigmoïde qui est défini comme un neurone largement accepté en tant que représentation mathématique du comportement d'un neurone biologique.

2) Réseaux de Neurones

Un réseau de neurones artificiels est un ensemble de neurones artificiels connectés. Chaque connexion entre neurones artificiels peut transmettre un signal de l'un à l'autre. Le neurone artificiel qui reçoit le signal peut le traiter, puis signaler les neurones qui y sont connectés. Les réseaux de neurones artificiels sont couramment utilisés pour effectuer des tâches particulières, notamment le regroupement, la classification, la prédiction et la reconnaissance de formes. De la même manière qu'avec le neurone perceptron et sigmoïde, un réseau de neurones acquiert des connaissances par l'apprentissage.***(Alexandre Bergel, "agileartificialintelligence.github.io/book/build/04-NeuralNetwork.html").***
Nous pouvons voir une simple représentation dans la figure ci-dessous:



Dans cette figure , Hidden layer représentent les différentes couches cachée du réseau qui contient 3

neurones sur cette couche et de même Output layer représentent la couche de sortir du réseau de neurones.
Plusieurs articles ont été produits dans le sens de comprendre ces réseaux de neurones, car plus la structure est compacte plus elle devient difficile à comprendre. Pour rester dans le même contexte que nous avons énoncé plus haut dans la visualisation des réseaux de neurones, nous présenterons dans la suite quelques travaux effectué dans ce sens.

II- Visualisation comme aide à la compréhension des réseaux de neurones

Pour une meilleure compréhension dans cette partie, nous nous attarderons sur l'article de *Fred Hohman, Minsuk Kahng, Robert Pienta, et Duen Horng Chau, "Visual Analytics in Deep Learning: An Interrogative Survey for the Next Frontiers" arXiv:1801.06889v3 [cs.HC] 14 May 2018.* ou ils abordent une approche très méthodique basé sur le questionnement donc les différentes réponses ont été les suivantes :

1. Pourquoi?

Dans l'article, 4 raisons fondamentale sont donnés à partir de sondages fait par 16 auteurs et donc il en découle que, visualiser le réseau de neurone permet d'interpréter, d'expliquer, de déboguer et d'améliorer la conception des modèles(algorithmes d'apprentissage développé par les concepteurs de réseaux de neurones).
En effet, les auteurs affirment que la visualisation se situe au niveau de la compréhension dans la prise de décision par les modèles non seulement, mais aussi de comment est-ce qu'ils apprennent pendant la formation afin de pouvoir faire confiance aux modèles formés.
Les auteurs mentionnent aussi la relation avec le débogage et l'amélioration des paramètres, car ici, les modèles sont déjà entraînés, ils ont déjà appris, mais il est nécessaire de visualiser les résultats et les métriques de chaque modèle pour voir qui a la plus grande précision qui a la plus faible précision... Tout en évitant le sur-apprentissage qui provoque de faux résultats attendue sur des jeux de donnée différents.

2. Qui?

En effet, pour répondre à cette question, trois catégories d'acteur sont cernées et classer suivant le degré de connaissance de chaque groupe de la notion de deep learning. Il s'agit de :

➢ les concepteurs de modèles,

La première catégorie de personnes utilisant le deep learning, c'est ceux qui développe, expérimenté et déploie les modèles d'apprentissage de réseaux de neurones. La connaissance et intuition en deep learning accélère la classification des modèles par exemple en fonction du

type de donner la bonne pratique exemple à savoir comment faire varier des hyper paramètres pour atteindre des meilleures performances.
Les outils de visualisation dans ce domaine est beaucoup plus technique, nous pouvons citer :

- TensorBoard,

Qui est une des plate-forme de visualisation open source incluse dans Google pour les bibliothèques graphiques de dataflow de TensorFlow qui est une interface permettant d'exprimer des algorithmes d'apprentissage automatique et une implémentation permettant d'exécuter de tels algorithmes.
Dans *M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin et al., "TensorFlow: Large-scale machine learning on heterogeneous distributed systems," arXiv:1603.04467, 2016*. Ces auteurs font une description de cet outil et il faut retenir que TensorBoard comprend un certain nombre de composants intégrés pour aider les développeurs de modèles de comprendre, déboguer et optimiser les programmes TensorFlow. Il inclut le traçage en temps réel des métriques de modèle quantitatives pendant la formation, la prédiction au niveau des instances et la visualisation du graphe de calcul.
Une capture du graphe de visualisation d'un CNN(Convolutional Neural Network) dans le même article nous permet d'avoir la capture ci-dessous :

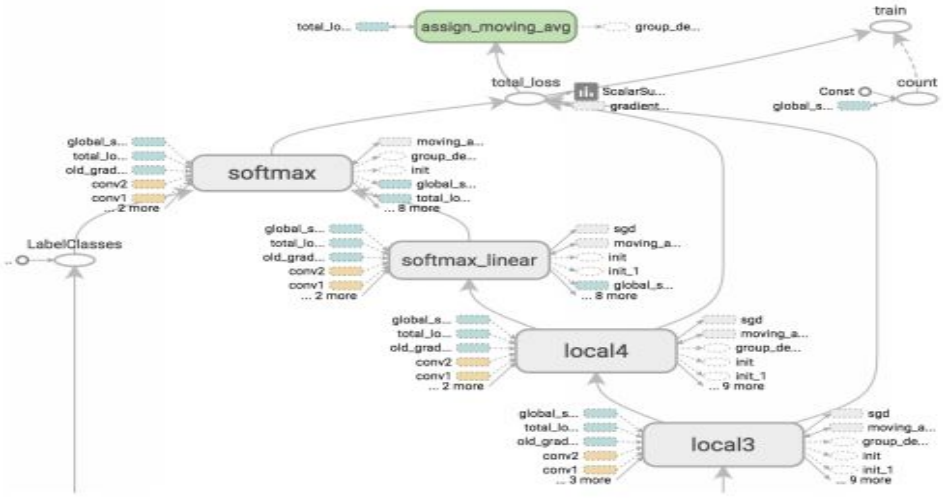


Figure 10: TensorBoard graph visualization of a convolutional neural network model

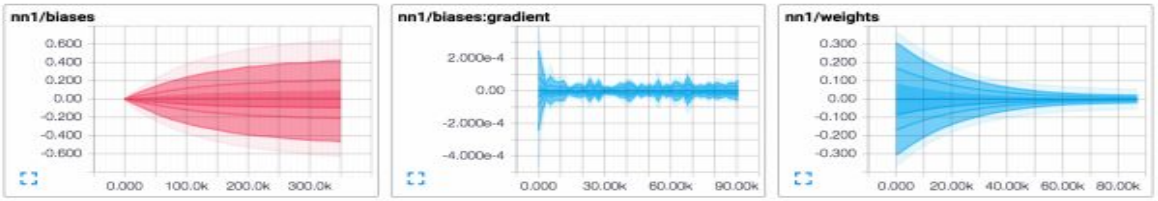


Figure 11: TensorBoard graphical display of model summary statistics time series data

- DeppEyes

Cet outil est présenté comme étant un système d'analyse visuelle progressive destiné à l'analyse des réseaux de neurones profonds pendant la phase de formation facilitant ainsi plusieurs tâches de création de modèles, par exemple l'identification de couches stables dans la conception des réseaux de neurones. Pendant le processus de formation, l'identification des couches inutiles, des filtres dégénérés qui ne contribuent pas à la performance d'un modèle décisions, l'élagage de telles entités et identification de modèles non détecté par le réseau, indique davantage que des filtres ou des couches supplémentaire peuvent être nécessaires. *(N. Pezzotti, T. Hollt, J. Van Gemert, B. P. Lelieveldt, E. Eisemann, and A. Vilanova, "DeepEyes: Progressive visual analytics for designing deep neural networks," IEEE TVCG, vol. 24, no. 1, 2018*).
Nous pouvons voir un petit affichage pendant l'entraînement tirer de cet article dans la capture ci-dessous :

(a) Aperçu sur la formation
(b) Histogrammes de perplexité
(c) Nouvelle visualisation cela permet la détection de couches stables
(d) L'activation de filtre est visualisée sur la carte en entrée.
(e) Visualisation et relations entrent les filtres.

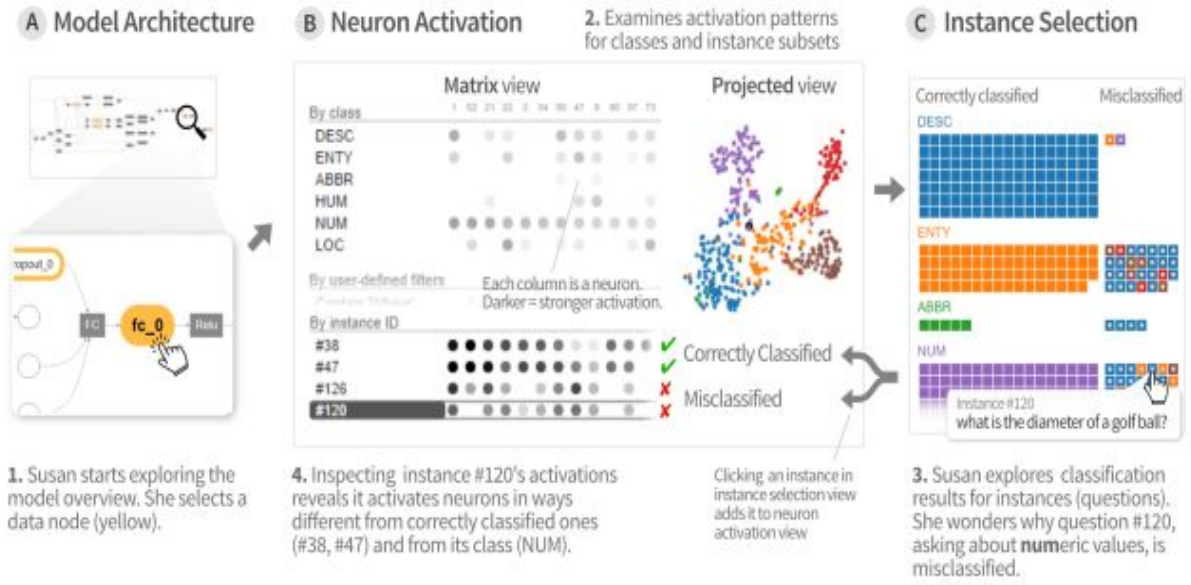
Plusieurs autres outils existent comme BLook, ML-o-scope, DGMTracker, mais nous nous limiterons juste à l'exploration de ces deux outils.

➢ Les utilisateurs de modèle,

Un exemple de système d'analyse visuelle pour cette catégorie d'utilisateurs est :

- ActiVis

C’est un système d'analyse visuelle permettant d'interpréter les résultats de réseaux de neurones à l'aide de nouvelle représentation visuelle qui unifie les niveaux d'instance, des sous-ensemble d’inspections des activations de neurones.
Les utilisateurs de modèle peuvent de manière flexible spécifiez des sous-ensembles à l'aide d'entités en entrée, d'étiquettes ou de tout résultat intermédiaire d'un pipeline d'apprentissage automatique. Dans la capture ci-dessous, tirer de l’article de *M. Kahng, P. Andrews, A. Kalro, and D. H. Chau, “ActiVis: Visual exploration of industry-scale deep neural network models,” IEEE TVCG, vol. 24, no. 1, 2018*. Ils présentent un peu quelques visualisations que nous pouvons réaliser avec cet outil.



1. Susan réalise une exploration de l'architecture du modèle, à travers sa vue d'ensemble du graphe de calcul (en A). Sélection d'un nœud de données (en jaune) affichage des activations neuronales (en B).

2. Matrice d’activation de neurones, elle montre les activations pour les instances et sous-ensembles d'instances ; la vue projetée affiche la projection 2D d'activations d'instances.

3. Dans le panneau de sélection d’instances (en C), elle explore les instances individuelles et leurs résultats de classification.

4. L’ajout d’occurrences à la matrice permet de comparer les modèles d'activation sur les instances, les sous-ensembles et les classes, révélant les causes des erreurs de classification.

Pour cette catégorie d’acteurs, plusieurs outils existent auxquels nous pouvons citer aussi LSTMVis, Embedding Projector et bien d’autres outils encore.

➢ Et les non expert en deep learning

La troisième catégorie d'acteurs correspond à ceux qui pourraient utiliser la visualisation comme aide à l'apprentissage des réseaux de neurones. C'est des individus qui n'ont généralement aucune connaissance préalable de l'apprentissage en profondeur, et peut ou peut ne pas avoir une formation technique. Une grande partie de la recherche ciblée sur ce groupe a un but éducatif, essayer d'expliquer ce qu'est un réseau de neurones et comment cela fonctionne à un niveau élevé, parfois sans révéler un apprentissage en profondeur est présent. Ce groupe comprend également les personnes qui utilisent simplement des dispositifs alimentés par l'IA et applications grand public. Plusieurs outils existent pour cette catégorie d'acteurs auxquels nous pouvons citer : Teachable, Deep Visualization Toolbox, TensorFlow Playground et bien d'autre application encore.

3. Qu'est ce qu'on visualise?

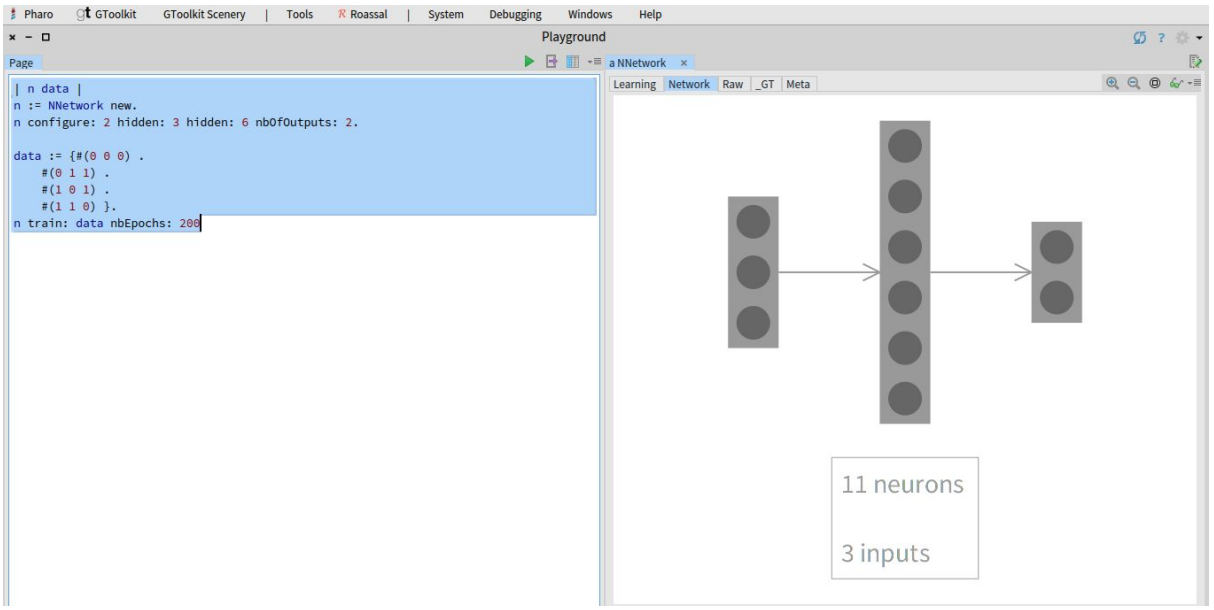
Dans l'article *Fred Hohman, Minsuk Kahng, Robert Pienta, et Duen Horng Chau, "Visual Analytics in Deep Learning: An Interrogative Survey for the Next Frontiers" arXiv:1801.06889v3 [cs.HC] 14 May 2018*, plusieurs entités peuvent être visualisé auxquels nous pouvons citer :

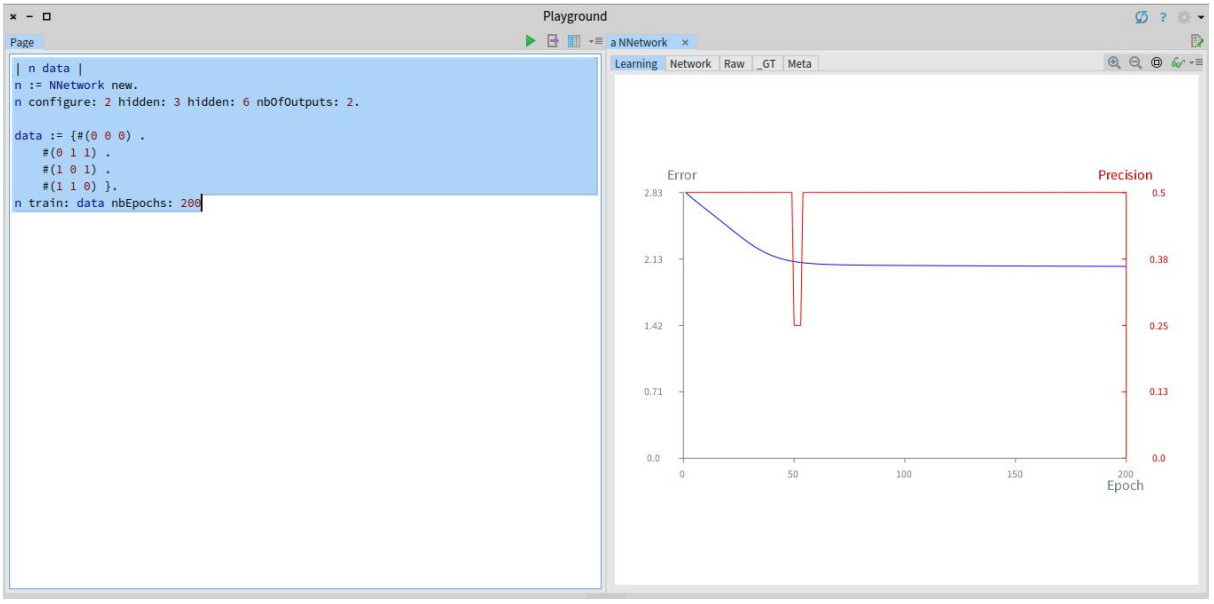
- Les graphes de calcul et aussi l'architecture des réseaux de neurones
- Les paramètres appris par notre modèle
- Les neurones
- Les différentes valeurs des activations
- L'erreur propagé après chaque entraînement
- Les métriques du modèle conçu
- Les variations des poids pendant l'entrainement

Parallèlement, plusieurs travaux sont effectués dans le même sens en Pharo, ou nous avons plusieurs outils de visualisations ont été conçus dans le sens d'améliorer la perception des visualisations notamment Roassal qui est un moteur de visualisation intégré dans Pharo, mais aussi GToolkit qui est un framework qui intègre plusieurs outils donc un moteur de visualisation de données d'une part, mais aussi un éditeur de code très interactif qui permet de concilier texte, code et puis visualisation et plein d'autre outils intéressant dans le même sens.
Tous ses outils fournissent des possibilités de visualisations des réseaux de neurones, mais n'offre malheureusement pas d'interprétation, d'une part et la documentation d'autre part des résultats obtenue qui pourront nécessairement servir à la compréhension plus en profondeur des réseaux de neurones. En d'autres termes, comment pouvons nous montrer que par la visualisation et en utilisant une approche exploratoire, nous pouvons arriver à interpréter et plus encore à comprendre le fonctionnement et interpréter les résultats renvoyés par les réseaux de neurones.

Nous pouvons les voirs dans quelques représentations ci dessous une combinaison de Roassal et

GToollkit qui montre une représentation architecturale des réseaux de neurone et ainsi que l'affichage de l'erreur propagé pendant chaque entraînement :





RÉFÉRENCES

1) *Marc Parizeau, "RÉSEAUX DE NEURONES GIF-21140 et GIF-64326 "Automne 2004, pp. 1.*

2) *Alexandre Bergel, "agileartificialintelligence.github.io/book/build/04-NeuralNetwork.html"*

3) *Fred Hohman, Minsuk Kahng, Robert Pienta, et Duen Horng Chau, "Visual Analytics in Deep Learning: An Interrogative Survey for the Next Frontiers" arXiv:1801.06889v3 [cs.HC] 14 May 2018.*

4) M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin et al., "TensorFlow: Large-scale machine learning on heterogeneous distributed systems," arXiv:1603.04467, 2016.

5) *N. Pezzotti, T. Hollt, J. Van Gemert, B. P. Lelieveldt, E. Eisemann, and A. Vilanova, "DeepEyes: Progressive visual analytics for designing deep neural networks," IEEE TVCG, vol. 24, no. 1, 2018*

6) M. Kahng, P. Andrews, A. Kalro, and D. H. Chau, "ActiVis: Visual exploration of industry-scale deep neural network models," IEEE TVCG, vol. 24, no. 1, 2018

Chapitre 3: Solution proposée

Introduction

Nous avons présenté dans le premier chapitre une analyse suffisante de notre sujet qui portait sur “Compréhension par l’exploration et la visualisation des réseaux de neurones” ensuite nous avons présenté dans le chapitre suivant quelques recherches et travaux existant à notre sujet.

Ainsi donc, en revenant a la question des réseaux de neurones artificielles qui font apparaître plusieurs acteurs dans leur cycle de vie, notre interrogation était de savoir comment nous pouvons concilier les différents points de vues de différents acteurs afin de faciliter la compréhension d’un réseau de neurones dans ses différentes dimensions.

Dans ce chapitre, nous proposons une solution à cette intérogation en s'intéressant aux travaux existant et en apportant notre contribution personnel. Pour montrer le bien-fondé de cette étude et de notre solution, nous évaluerons les réponses sur un ensemble de questions présenté plus bas.

Problème

Petit rappel sur notre question de recherche :

<<Comment pouvons-nous concilier les différents points de vues des différents acteurs utilisants les réseaux de neurones afin de permettre la compréhension d'un réseau de neurones dans ses différentes dimensions ?>>.

Nous avons présenté dans le chapitre précédent des travaux qui ont été réalisés dans ce domaine que nous ne rappellerons plus ici.

Ainsi, nous pouvons au vu de tout ce qui existe une grande proposition d’élaboration d’outil basé sur deux grandes approches :

- ➢ La première est l’Utilisation d'outils adaptables (Moldable tools) qui permettront de faciliter aux différentes catégories d'utilisateurs la construction et l'adaptation de moyens de visualisation et de compréhension des réseaux de neurones.
- ➢ La seconde consisterait à introduire l’Utilisation d'une démarche exploratoire proche du "*litterate programming*" qui permettrait dynamiquement de lier du texte narratif, du code et puis la visualisation.

Comme nous l’avons présenté plus loin dans le chapitre précédent, ces deux approches de solutions, on été abordé dans plusieurs plateformes comme Activis, DeepEyes, TensorBoard, et sans oublier GToolkit basé sur Pharo.

1- Qu'allons-nous faire ?

Notre travail consisterait à utiliser les réseaux de neurones comme exemple pour la plateforme GToolkit et le moteur de visualisation Roassal basé sur Pharo afin de valider nos solutions énoncé dans le paragraphe précédent.

A cet effet, nous nous inspirerons de la question du What? introduit et bien expliqué dans l'article de ***Hohman, Fred Matthew and Kahng, Minsuk and Pienta, Robert and Chau, Duen Horng, « Visual Analytics in Deep Learning: An Interrogative Survey for the Next Frontiers*** » qui traite des composants techniques de neurones réseaux pouvant être visualisés. En décrivant d'abord ce qui peut être visualisé, nous pouvons plus facilement fonder notre discussion sur comment les visualiser .

Plusieurs entités été décrites dans le chapitre précédent nous nous occuperons dans Pharo de visualiser :

- Les paramètres appris par notre modèle
- Les neurones
- Les différentes valeurs des activations
- Les variations des poids pendant l'entrainement

Deuxièmement, nous utiliserons GToolkit pour proposer un tutoriel d'apprentissage et conciliation code et visualisation pour les acteurs non-experts en intelligence artificielle.

Ces deux approche nous aideront à fournir des raisons suffisante de réaliser des visualisation des réseaux de neurones pour améliorer leur compréhension, l'explication, l'interprétation, le débogage, et l'amélioration des modèle pour les concepteurs et les utilisateurs de modèles conçus.

2- Pourquoi allons-nous faire cela ?

Nous visualisons ces paramètres dans le but de montrer une compréhension des réseaux de neurones par les visualisation pendant différentes phases d'entraînement, d'apprentissage, de transmission de paramètre.

Cette expérimentation nous aidera peut-être pour tirer des conclusions des visualisation que nous obtenons afin de tirer des informations du comportement de ces réseaux de neurones artificiels.

Nous concevons un tutoriel interactif conciliant code, texte et visualisation afin de rejoindre la deuxième de notre solution qui était beaucoup plus orienté vers la troisième catégorie d'acteur les non-experts en intelligence artificielle.

Cela permettra non seulement de faciliter la compréhension des réseaux de neurones par différentes catégories d'utilisateur, mais aussi une très grande interaction et une facilité au

niveau de la manipulation de ces derniers.

3- Comment allons-nous faire cela ?

Dans la première approches ou nous visualisons les éléments suivants :

a- Les paramètres appris par notre modèle

Pour visualiser les paramètres appris, nous nous situons exactement pendant les différentes époques d'apprentissage ou nous récupérons successivement le bias, le poids et les valeurs renvoyées par le modèle (afin de les comparer à la fonction de sortie attendue et aux valeurs de l'époque précédente). Cela permettra aux concepteurs de ses modèles d'avoir une intuition sur l'ajustement et l'initialisation des paramètres.

Elle permettra également aux utilisateurs de modèles de comparer les modèles afin de sélectionner le modèle le plus prédictif non seulement permettra de suivre l'allure des variable afin d'anticiper sur d'éventuels sur-apprentissages du modèle.

Pour leurs représentations, nous utilisons une courbe représentant les différentes valeurs observer pendant la phase de formation du modèle.

b- Les neurones

Nous visualisons pour chaque couche du réseau de neurones les neurones comportant ses couches.

Pour ce faire nous récupérerons des neurones sur les couches caché de notre réseau et nous représenterons sur un graphe noeud-lien.

Nous visualisons plus particulièrement les différents paramètres de chaque neurone toujours pendant toutes la phase d'entraînement du modèle pour voir la transmission de nouveaux paramètres au niveau de chaque neurone et cette connaissance pourrait peut-être aider plus particulièrement les concepteurs de modèles dans la conception architecturale des réseaux.

c- Les différentes valeurs des activations

Nous les représenterons dans une courbe pour mieux suivre leur évolution qui nous aiderons de déduire un seuil à partir desquels le modèle produirait des valeurs avec une probabilité très stable et cela pourrait aider les concepteurs et les utilisateurs sur le nombre d'entraînements à effectuer pour une formation normale du modèle.

d- Les variations des poids pendant l'entrainement

Ces paramètres nous les représenterons également sur une courbe pour étudier leurs variations après chaque époques d'entraînement afin de voir à partir du quels valeurs des variables commence a avoir des valeurs constantes et cela pourrait nous aider à comprendre peut etre a partir de quel seuil le réseau de neurone fini son entrainement.

Pour la deuxième approche de solution nous allons à partir de GToolkit, nous allons concevoir un exemple de tutoriel expérimentant basé sur notre deuxième solutions proposées.

4- Qu'est-ce qui existe pour nous aider ?

Ainsi pour mener à bien notre étude, plusieurs outils existe déjà et nous à été fourni dans Pharo plus précisément :

- Roassal, qui est un moteur de visualisation basé sur Pharo, nous l'utiliserons spécialement pour la représentation graphique de nos courbes, car elle incarne plusieurs outils graphiques qui nous faciliteront l'affichage des différents paramètres récupérer et énumérés plus haut.(http://scg.unibe.ch/download/merino/roassal-tutorial/tutorial.html)
- GToolkit, qui est un framework Pharo, qui dispose de plusieurs outils de visualisation agile permettant d'avoir un retour immédiat d'information à partir du code non seulement, mais aussi un éditeur de code très interactif qui permet de concilier texte, code et puis visualisation.(https://gtoolkit.com/#install)
- Enfin, nous mènerons une expérimentation basée sur une analyse visuelle et explicative des réseaux de neurones en s'inspirant plus particulièrement des travaux d'Alexandre Bergel (https://agileartificialintelligence.github.io/) sur l'implémentation des réseaux de neurones en Pharo.

Dans la suite de notre travail, nous présenterons le dernier chapitre sur la partie pratiques de tout ce que nous avons énuméré plus haut.

RÉFÉRENCES

1) *https://gtoolkit.com/#install*

2) *Alexandre Bergel, "agileartificialintelligence.github.io/book/build/04-NeuralNetwork.html"*

3) *Fred Hohman, Minsuk Kahng, Robert Pienta, et Duen Horng Chau, "Visual Analytics in Deep Learning: An Interrogative Survey for the Next Frontiers" arXiv:1801.06889v3 [cs.HC] 14 May 2018.*

4) http://scg.unibe.ch/download/merino/roassal-tutorial/tutorial.html

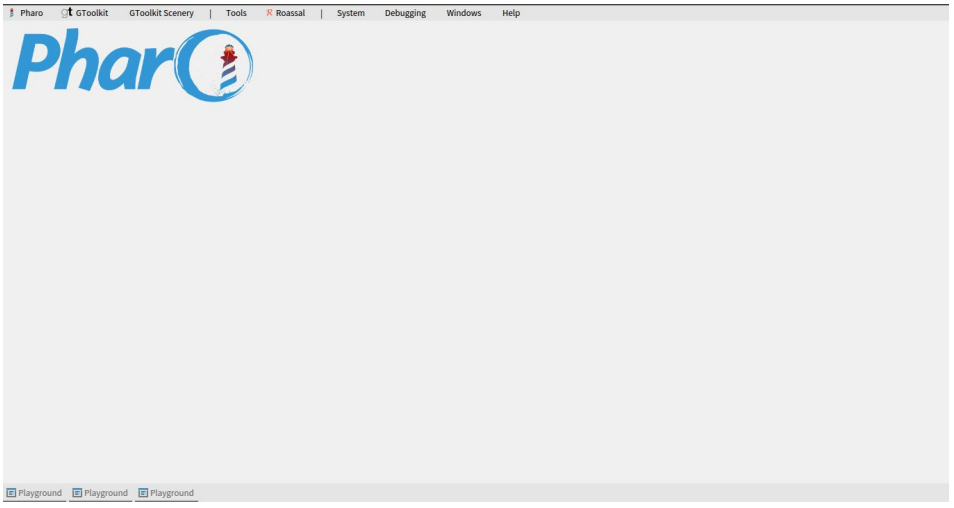
Chapitre 4: Réalisation

Introduction

Pour fournir des résultats de notre travail dans cette partie pratiques nous ferons une petite présentation de notre environnement de travail ensuite les fonctions que nous avons eu à mettre en œuvre, ensuite une phase d'expérimentations et ensuite d'une analyse de nos résultats obtenue et enfin quelques perspectives.

I- Environnement de travail

Pour mener à bien notre démarche Exploratoire et Expérimentale, nous utiliserons le langage de programmation Pharo qui est un langage de programmation immersif et qui propose déjà une très bonne base de départ avec non seulement une implémentation de réseau de neurones mais aussi des outils très puissant de visualisation et d'interaction directe avec le code.



Après importation du projet ***NeuralNetwork*** de Alexandre Bergel ou il réalise une implémentation des réseaux de neurone en Pharo, nous avons ajouté plusieurs autres classes et méthodes pour la représentation de notre expérimentation.

- la ***classe VisualisationNNeuralExample***

comprends 5 méthodes

  - ***exampleVisuUtilisateurModele***

Cette fonction comporte une configuration spéciale pour les utilisateur de modèle qui

leurs permettrait d'ajuster juste les paramètre de formation du modèle comme par exemple le nombre de couche du réseau, la position du neurone a visualiser, le nombre d'époque de formation, nous verrons cela plus bas sur des cas pratique.

- ***exampleVisualisationBias***

Cette fonction permet d'afficher en fonction des paramètre introduit d'afficher la courbe de variation du bias pour un neurone donné nous le verrons dans les captures.

- ***exampleVisualisationEntrainement***

Cette fonction permet d'afficher en fonction des paramètre introduit d'afficher la courbe de variation des valeurs de sortie qui seront comparer avec la valeur attendu pour tous le réseau de neurones.

- ***exampleVisualisationErrorPrecision***

Cette fonction permet d'afficher en fonction des paramètre introduit d'afficher la courbe de variation de l'erreur après une époque d'entraînement.

- ***exampleVisualisationWeights***

Cette fonction permet d'afficher en fonction des paramètre introduit d'afficher la courbe de variation des poids pour un neurone donné nous le verrons dans les captures.

- la ***classe DocumentationVisualisationNNeuralExample***

Elle comprend des méthode permettant l'expérimentation de la formation interactive (surtout pour les non-experts).

- la ***classe NNetwork***
  - ***VisualisationparamWeights***

  Pour la visualisation des poids transmis pendant l'entrainement
  - ***constructionGrapheDePoint***

  pour la récupération successive de tous les paramètres pendant l'entrainement
  - ***firstNeuronLayer***

  Pour récupérer la position de la couche courante
  - ***grapheBias***

  Pour afficher la courbe représentative de la variation de bias
  - ***grapheEntrainement***

  Pour afficher la courbe représentative de la variation des output durant tout l'entraînement
  - ***grapheWeights***

- ***nombreEpoqueFormation***
- ***positionNeuron***
- ***positionNeuroneaVisualiser***

- la ***classe NeuronLayer***

Cette classe déjà prédéfinie par Alexandre Bergel contient déjà certaine méthode de traitement de réseau neurones donc nous nous avons utilisé et ajouté deux autres méthode supplémentaire qui nous aideront dans nos travaux de visualisation:

- ***positionNeuron***

cette méthode permet de récupérer et ou de se positionner a une position donnée d'un neurone sur la couche courante.

- ***neurons***

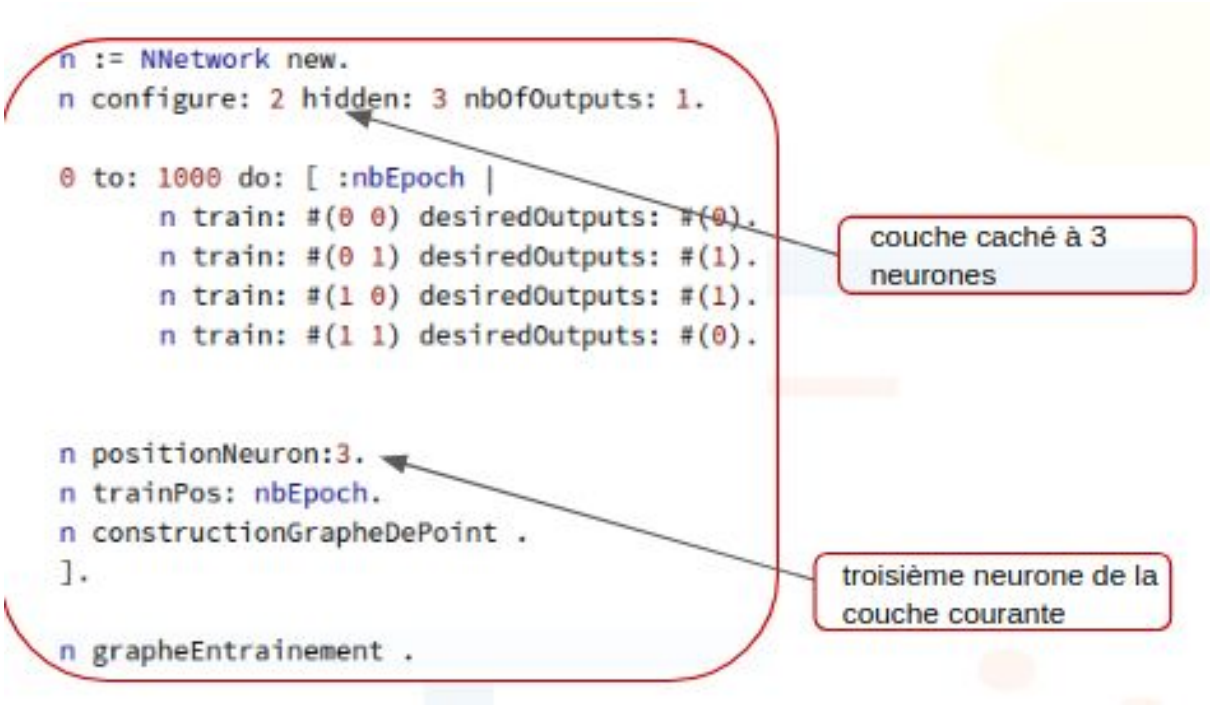
Elle permet de récupérer un neurone.

- ***la classe TPEtest***

Cette classe comprend un certain nombres de tests qui nous ont permis de valider nos fonctions écrites dans le cadre de notre expérimentation.

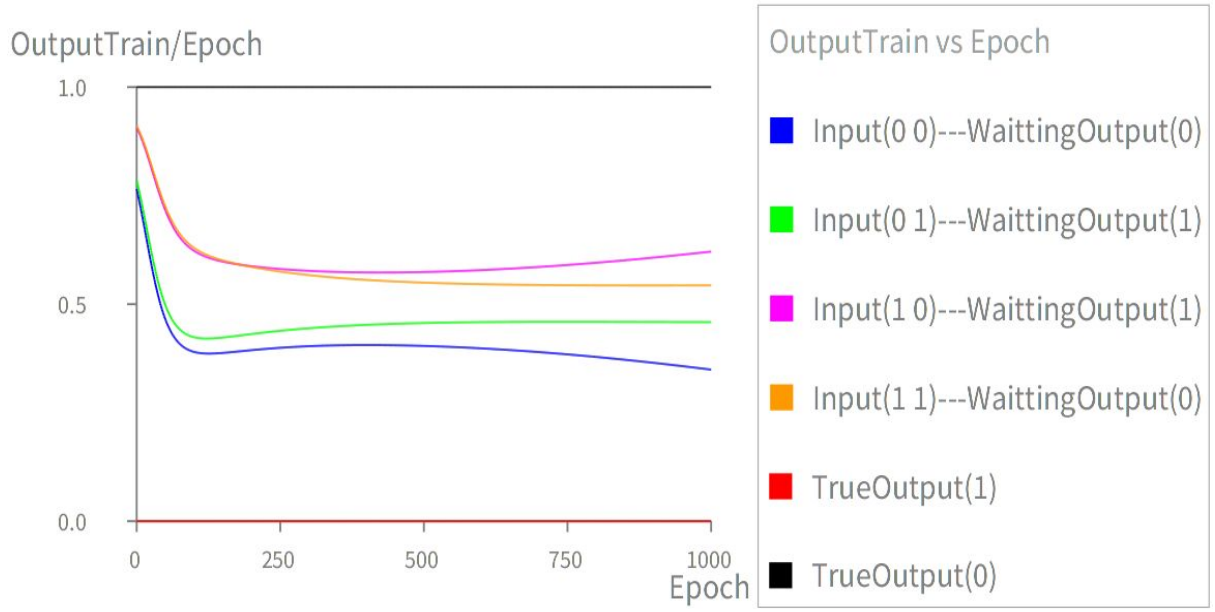
II- Expérimentations

Ici nous présenterons quelques capture expliquez de notre étude expérimentale.

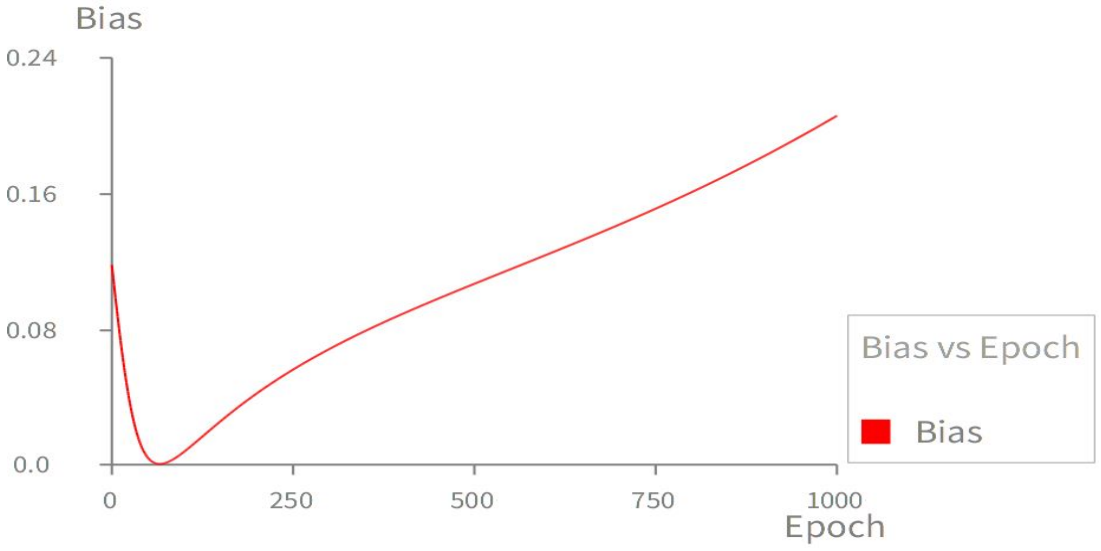


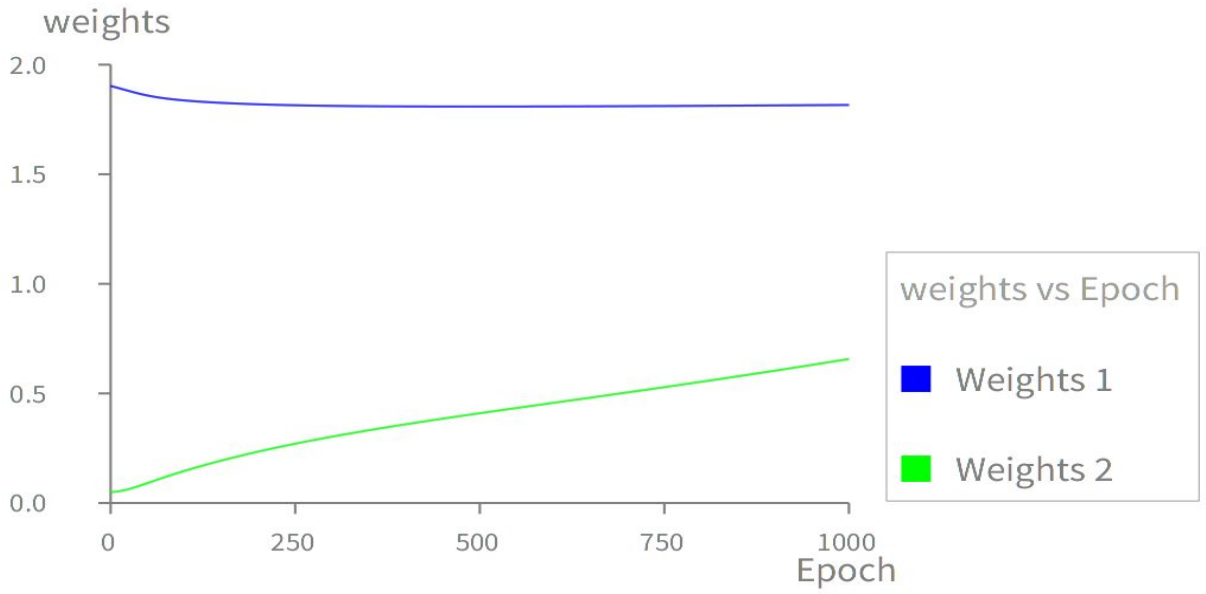
Dans le code ci dessus nous avons une configure de réseau à 2 couche cachés et composé de 3 neurones et nous visualisons la position du neurone a la position 3 de la couche

courante les résultats obtenus donne :



Avec cette courbe de sortir qui représentent les différents output après chaque époque d'entraînement, nous pouvons en fonction de l'allure de ces courbes nous rendre compte que a partir de 1000 epoque d'entraînement, le réseau n'a pas encore suffisamment appris pour pouvoir deviner avec une très grande probabilité le résultat attendue car nous pouvons voir sur le graphe les sortie attendue pour chaque couleur qui représente chaque entraînement. nous pouvons voir ses explication dans le tableau de bord a droite de la courbes de sortie après chaque entraînement.

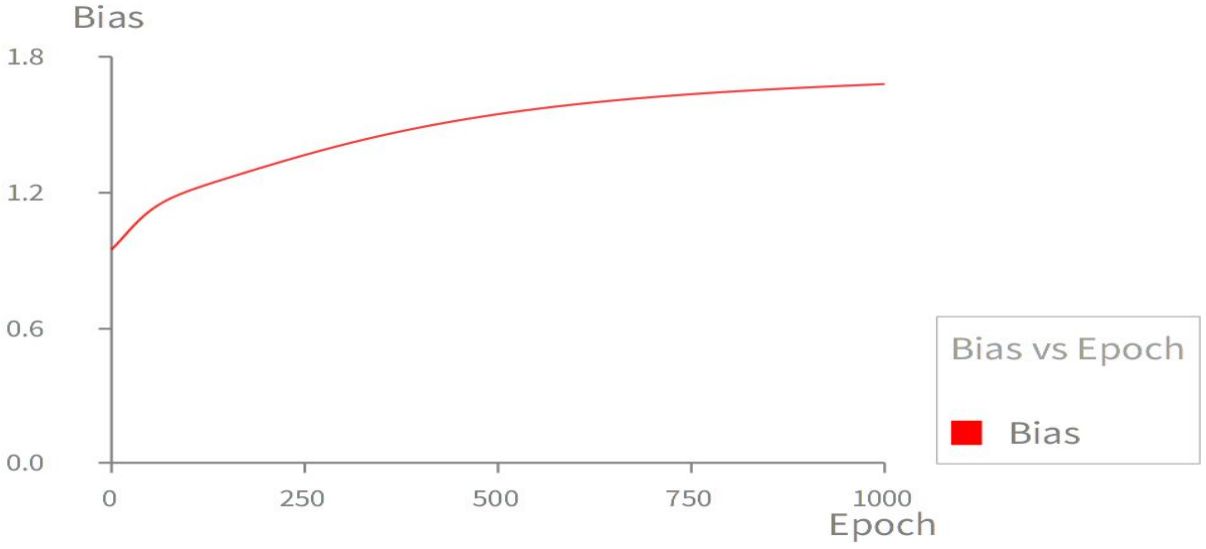


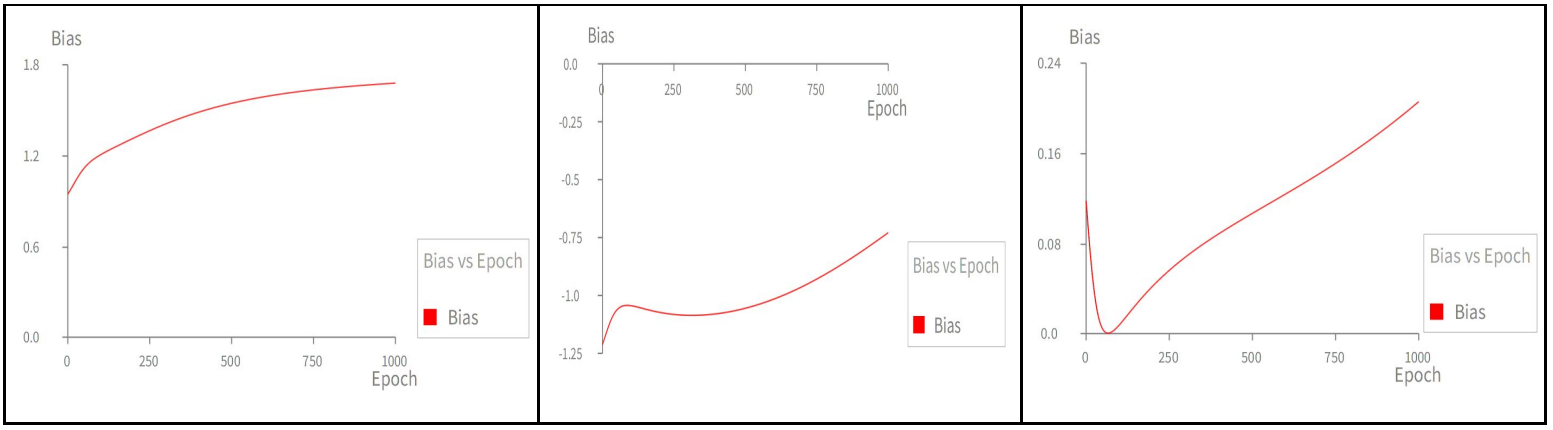


Nous avons également dans la représentation plus les courbes de variation des poids et bias après chaque époque d'entraînement.

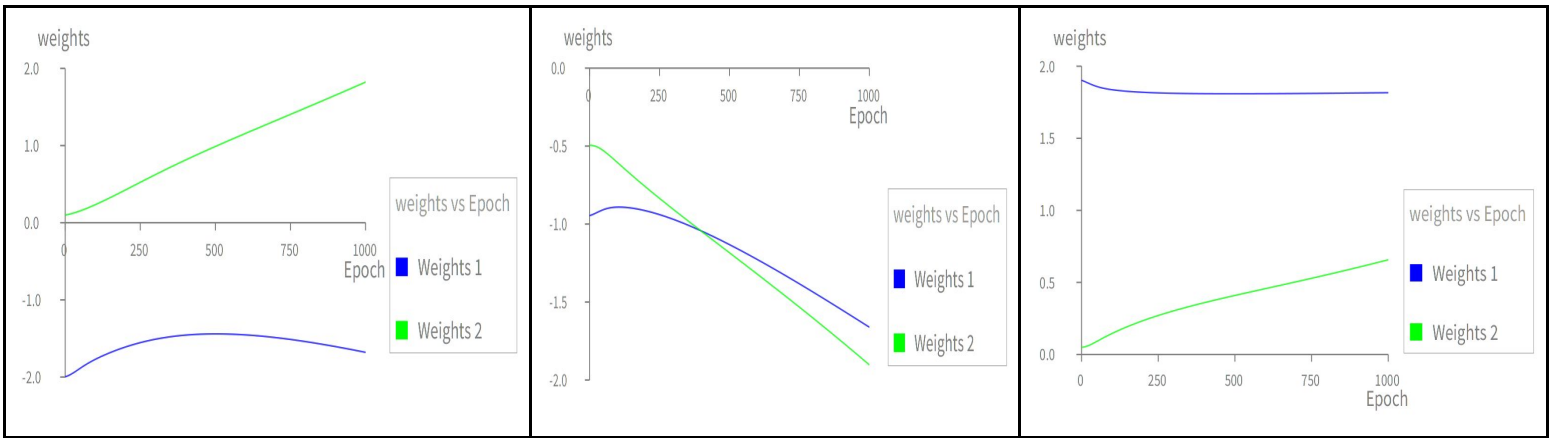
Nous constatons également qu'après 1000 époque les variation des poids et bias son différent et la variation est beaucoup plus constater au niveau de la courbe de bias cela peut nous aider a comprendre que pendant la phase d'entraînement le bias est le paramètre qui influence le plus dans cette phase et cela pourra aider pendant la phase d'initialisation de paramètre.

De même , sur la courbe de bias pour le neurone numéro 1 de la couche courante, nous constatons que la variation du bias est très différent de celui du neurone numéro 3 de la couche courante que nous avons visualiser plus haut. ces information nous permettre également de comprendre que au cours de l'entraînement, chaque neurone apprend des paramètres différents des autres neurons.





Courbe représentative visualisation bias pour 3 neurone pendant 1000 époques d'entrainements



Courbe représentative visualisation poids pour 3 neurone pendant 1000 époques d'entrainements

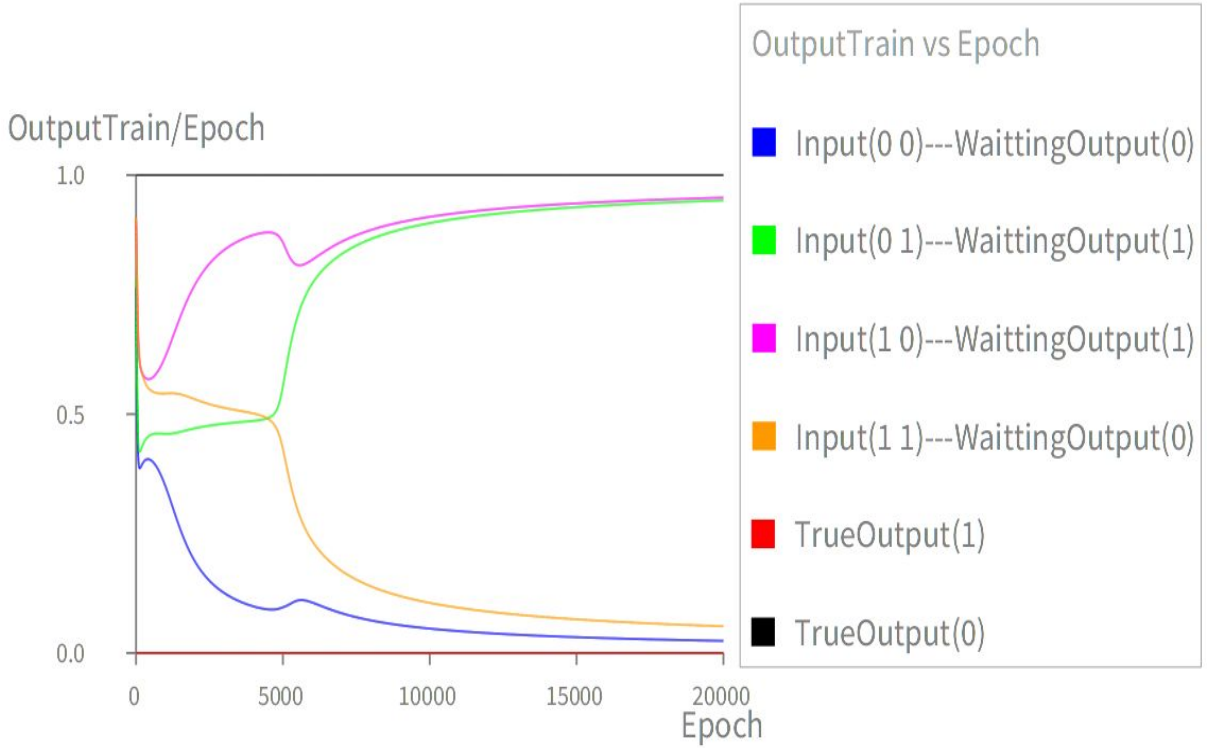
Dans les visualisations plus haut nous pouvons voire des variation pendant le même entraînement de plusieurs variable pour chaque entité de neuron. ces information pourront aider les concepteurs de modèle sur comment produire une architecture plus puissance pour obtenir des bon résultats après un nombre limité d'entraînement.

De même, pour la configuration ci-dessous, nous obtenons des résultat différents comparé à notre architecture donné plus haut :

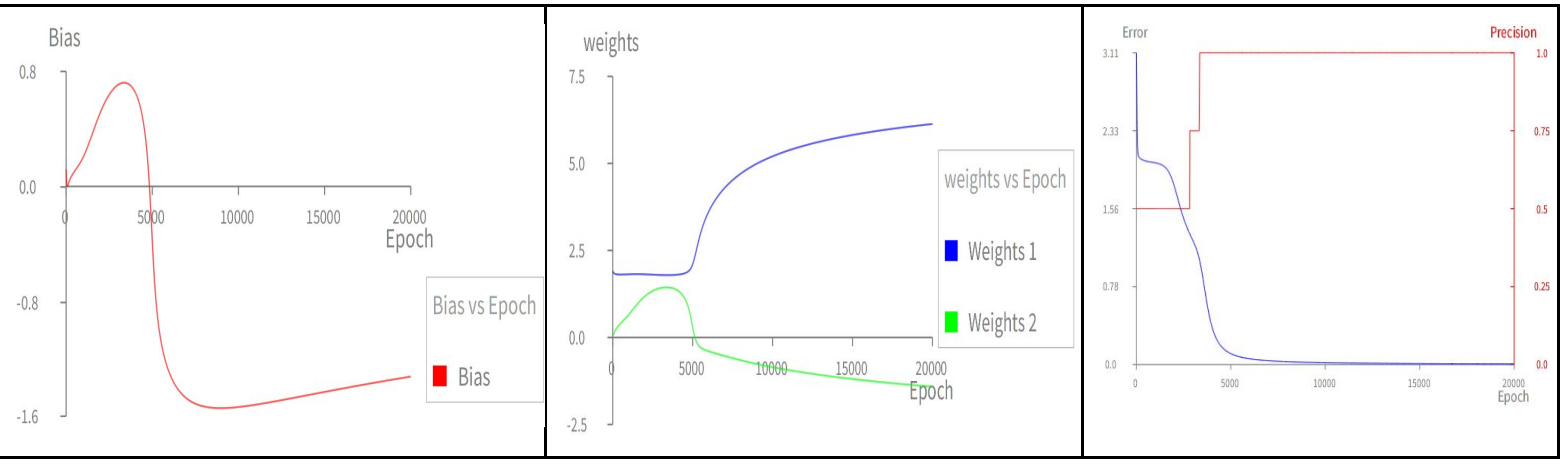
```
n := NNetwork new.
n configure: 2 hidden: 3 nbOfOutputs: 1.

0 to: 20000 do: [ :nbEpoch |
      n train: #(0 0) desiredOutputs: #(0).
      n train: #(0 1) desiredOutputs: #(1).
      n train: #(1 0) desiredOutputs: #(1).
      n train: #(1 1) desiredOutputs: #(0).


n positionNeuron:3.
n trainPos: nbEpoch.
n constructionGrapheDePoint .
].

n grapheEntrainement .
```

- passage de 2000 apprentissage à 20.000 apprentissage.

Dans cet échantillon de code nous avons pour les même paramètres mais avec cette fois le nombres d'entraînement de 20.000 on obtient les représentations suivantes:



Nous observons dans cette courbes de variation qu'a partir de 10 000 mile entrainements, notre réseaux devine très bien le bon résultat attendue.



Courbe représentative visualisation bias, poids et erreur pour 3 neurone pendant 20000 époques d'entrainements

Nous pouvons faire des même remarques présente sur les même représentation après changement de paramètre sauf qu'ici on arrive a atteindre l'objectif le réseau apprend très bien et nous pouvons tirer des conclusion que le réseau apprend très bien a partir de 15 000 mile époques

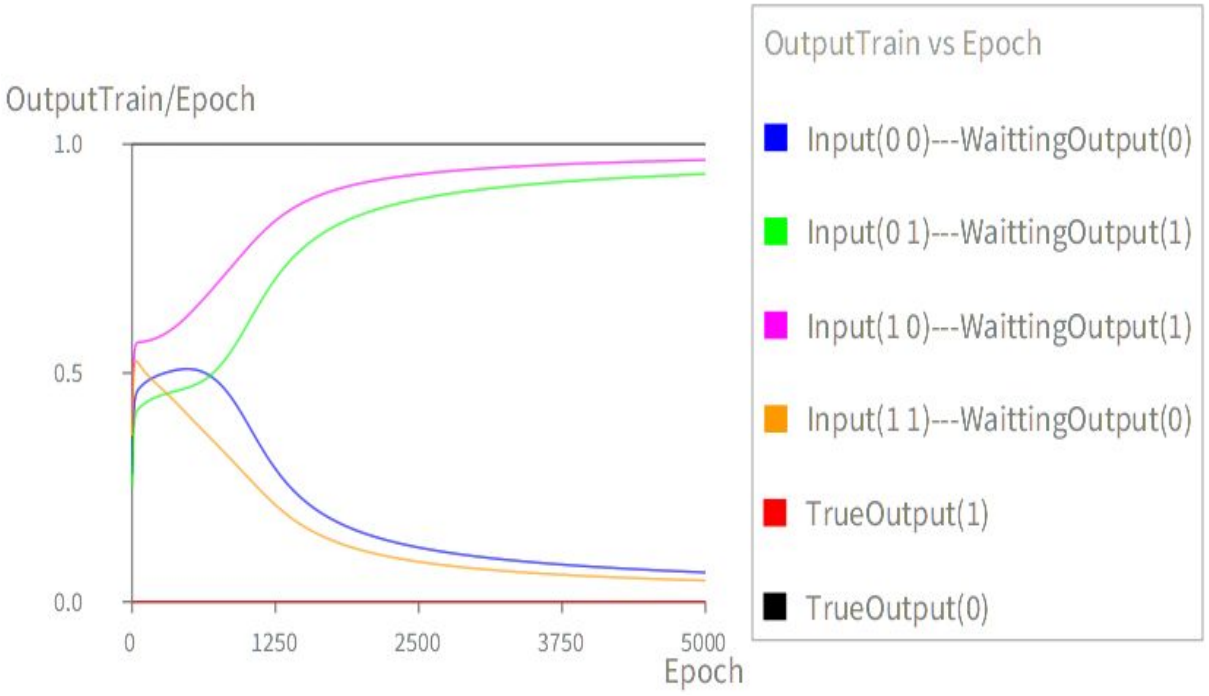
Pour la troisiemes expérimentation, nous avons :

```
n := NNetwork new.
n configure: 2 hidden: 10 hidden: 10 nbOfOutputs: 1.

0 to: 5000 do: [ :nbEpoch |
      n train: #(0 0) desiredOutputs: #(0).
      n train: #(0 1) desiredOutputs: #(1).
      n train: #(1 0) desiredOutputs: #(1).
      n train: #(1 1) desiredOutputs: #(0).


n positionNeuron:3.
n trainPos: nbEpoch.
n constructionGrapheDePoint .
].

n grapheEntrainement .
```

- 2 couches
- 10 neurones/couches
- 5000 cycles

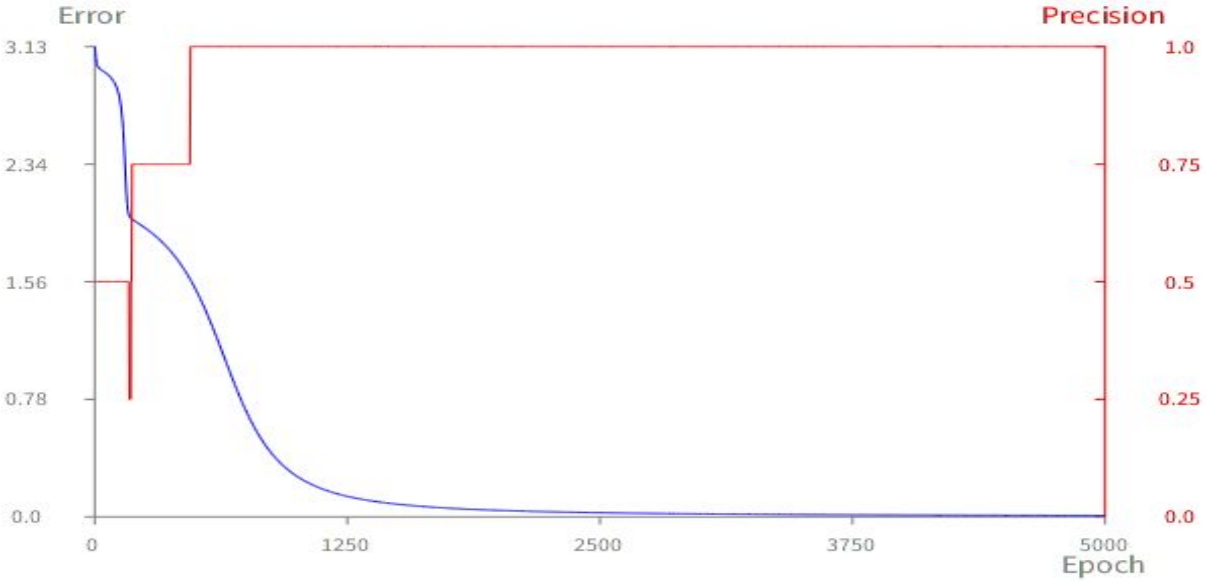
nous avons dans cette expérimentations 2 couches cachés et 10 neurons par couches sur 5000 époques d'entraînement.

les résultats obtenue sont les suivants :



Courbe représentative visualisation output pour 5000 entrainements et 2 couches cachés époques d'entrainements

pour cet entraînement nous avons la représentations de l'erreur et de la précision :



de la même façon nous pouvons voir la représentation dans GToolkit d'un exemple d'implémentation de notre deuxième solutions celui de créer une structure d'entraînement interactif liant code texte et puis visualisation nous avons pour cet exemple que nous avons visualiser plus haut une représentation ci dessus ou l'utilisateur a la possibilité d'interagir directement avec son code et du texte en même temps .

a DocumentationVisualisationNNeuralExample class (DocumentationVisualisationNNeuralExample)

Comment Definition Methods InstVars Examples References All Ref Outside Subclasses Raw Connections Print Meta

Allez exécuter le code si-dessous et vous aurez les résultats:

```
|ReseauN|
   ReseauN := NNetwork new.
   ReseauN configure: 2 hidden: 3 hidden: 5 nbOfOutputs: 1.
```

Après la configuration de votre modèle, vous devez pensez a y introduire un jeux de données pour commencer a entrainé le réseaux a repondre éfficacement. exécuter le code ci-dessous:

```
|ReseauN|
   ReseauN := NNetwork new.
   ReseauN configure: 2 hidden: 3 hidden: 5 nbOfOutputs: 1.

   0 to: 2000 do: [ :nbEpoch |
          ReseauN train: #(0 0) desiredOutputs: #(0).
          ReseauN train: #(0 1) desiredOutputs: #(1).
          ReseauN train: #(1 0) desiredOutputs: #(1).
          ReseauN train: #(1 1) desiredOutputs: #(0).

   ReseauN positionNeuron:3.
   ReseauN trainPos: nbEpoch.
   ReseauN constructionGrapheDePoint ].
```

Nous venons d'introduire notre jeux de donnée pour la porte logique Xor donc il est question nous allons realiser 2000 entrainement par defaut mais vous pouvez modifier cette valeur et le resultat de l'entrainement sera different.

Il faut noter que la fonction d'entrainement train a deja ete implementer dans le livre d'alexandre bergel je vous invite a y faire un tour `https://github.com/AgileArtificialIntelligence/PharoSourceCode`.

-**"ReseauN train: #(0 0) desiredOutputs: #(0).**" cette partie du code concerne votre entrainement train prend en paramètre une collection comme input et le desiredOuput désigne pour sa part la sortie attendue par le jeux de donnée.

a DocumentationVisualisationNNeuralExample class (DocumentationVisualisationNNeuralExample)

Comment | Definition | Methods | InstVars | Examples | References | All Ref Outside | Subclasses | Raw | Connections | Print | Meta

-**"ReseauN train: #(0 0) desiredOutputs: #(0).**" cette partie du code concerne votre entrainement train prend en paramètre une collection comme input et le desiredOuput désigne pour sa part la sortie attendue par le jeux de donnée.

-**"ReseauN positionNeuron:3.**" cette fonction position permet de se positionner sur un neurone en particulié sur une couche vous verrez son importance dans la suite.

-**"ReseauN trainPos: nbEpoch.**" elle désigne en effet l'etat du cicle d'entrainement cette fonction vous aideras au moment d'afficher votre graphe.

-**"ReseauN constructionGrapheDePoint**" cette fonction tres tres capitale servira pour la récupération des paramètres pendant la formation du Réseaux de Neurones.

l'Entrainement étant terminer nous allons procéder a la visualisations de la varriations des éléments transmis pendant l'apprentissage. Exécuter le code ci-dessous pour avoir les dernier rendu.
NB : pour un rendu totale vous pouvez copier l'intégraliter de se code dans votre playground et observer les résultats avec Roassal allez ne vous arrêter pas!!!

```
|ReseauN|
   ReseauN := NNetwork new.
   ReseauN configure: 2 hidden: 3 hidden: 5 nbOfOutputs: 1.

   0 to: 2000 do: [ :nbEpoch |
           ReseauN train: #(0 0) desiredOutputs: #(0).
           ReseauN train: #(0 1) desiredOutputs: #(1).
           ReseauN train: #(1 0) desiredOutputs: #(1).
           ReseauN train: #(1 1) desiredOutputs: #(0).

   ReseauN positionNeuron:3.
   ReseauN trainPos: nbEpoch.
   ReseauN constructionGrapheDePoint ].

ReseauN grapheBias .
```

-**"n grapheWeights**" est une fonction prédéfie pour afficher le graphe de variation de **Bias**.

a DocumentationVisualisationNNeuralExample class (DocumentationVisualisationNNeuralExample)

Comment | Definition | Methods | InstVars | Examples | References | All Ref Outside | Subclasses | Raw | Connections | Print | Meta

#Visualisation Weights

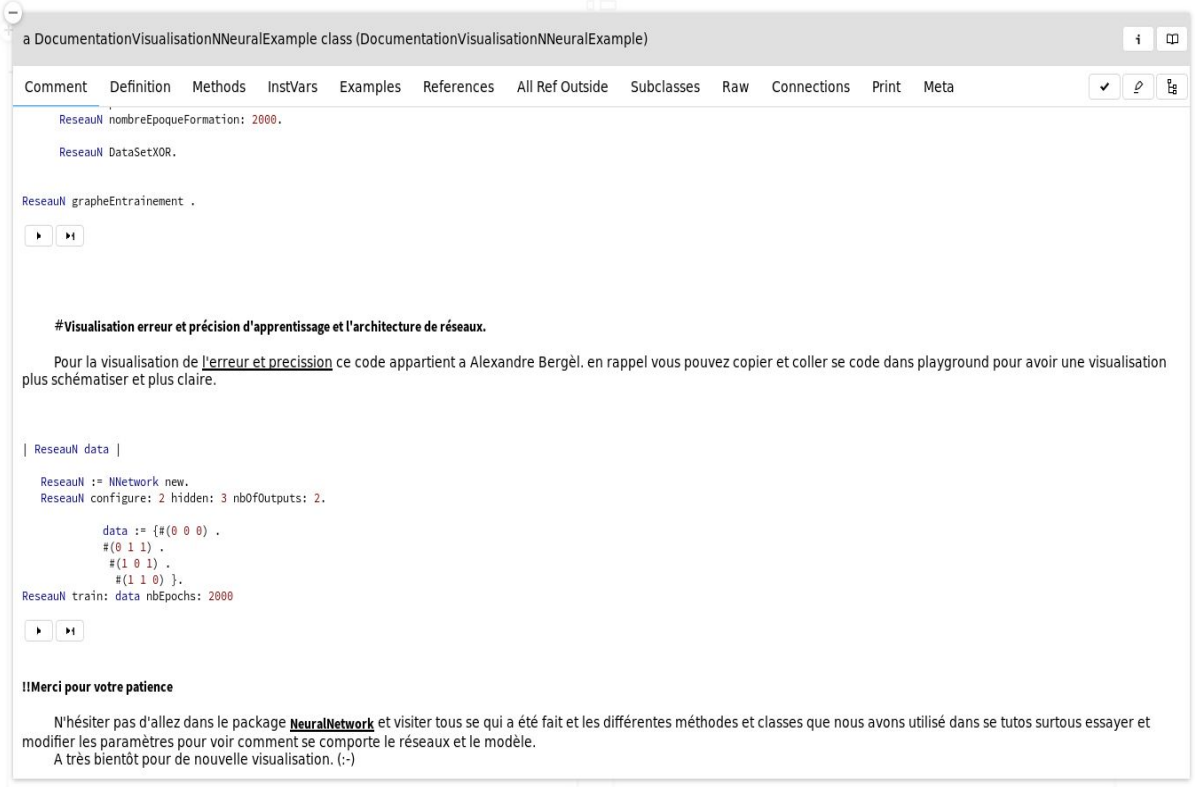
Pour la visualisation des poids(weight) le seule cahngement dans le code est au niveau de la fonction **"ReseauN grapheWeights .**" qui permet d'afficher le graphe de variations des Weight. en rappel vous pouvez copier et coller se code dans playground pour avoir une visualisation plus schématiser et plus claire.

```
|ReseauN|

   ReseauN := NNetwork new.
   ReseauN configure: 2 hidden: 3 hidden: 3 nbOfOutputs: 1.

      ReseauN positionNeuroneaVisualiser: 2.
      ReseauN nombreEpoqueFormation: 2000.

      ReseauN DataSetXOR.

ReseauN grapheWeights .
```

#Visualisation Entrainement

Pour la visualisation des ouput pendant l'entrainement, le seule changement dans le code est au niveau de la fonction **"ReseauN grapheEntrainement .**" qui permet d'afficher le graphe de variations des Output. en rappel vous pouvez copier et coller se code dans playground pour avoir une visualisation plus schématiser et plus claire.

```
|ReseauN|

   ReseauN := NNetwork new.
   ReseauN configure: 2 hidden: 3 hidden: 3 nbOfOutputs: 1.

      ReseauN positionNeuroneaVisualiser: 2.
      ReseauN nombreEpoqueFormation: 2000.

      ReseauN DataSetXOR.
```



Dans les captures précédentes, nous avons présenté quelques expérimentations de notre approche “literate programing “ qui fait intervenir code et puis texte en même temps tout en donnant la possibilité au différents acteurs d’interagir directement avec le code en apprenant en même temps.

III- Comparaison

Dans le chapitre concernant l’état de l’art, nous avons présenté plusieurs outils existant mais ces dernières ont plusieurs limites auxquels nous pouvons citer :

- ➢ Elles offres des plates formes d’interaction.
- ➢ Pas possible d’interagir avec le code
- ➢ Abstrait et moins Explicite
- ➢ Adapté pour les utilisateurs Expert en IA
- ➢ Ne permettent pas de conclure par exemple et d’apprendre pour les non expert en intelligence artificielle

Par ailleurs, nous avons également grâce à notre expérimentation pu tirer des conclusion et interagir directement avec notre code non seulement mais aussi participer à la formation et interactions entre différents acteurs en intelligence artificielle. cette approche nous a permis de tirer les conclusions suivantes de notre travail :

- ➢ Possibilités d'interagir directement avec le code
- ➢ Possibilité de construire une structure un peu plus intransigeante que le notebook python qui offre juste pour sa part la possibilité de taper du code et d'avoir des résultats or avec cette approche dans GToolkit nous pouvons non seulement faire des saisie textuels, introduire en même temps du code, ce qui est beaucoup plus intéressant .
- ➢ possibilité d'ajuster des paramètres manuellement et aussi de modifier la structure Neuronale
- ➢ Très avantageux pour les concepteurs pour la validation et la comparaison des modèle avant déploiement
- ➢ Mieux adapté pour toutes les catégorie d'acteurs travaillant sur le réseau de neurones.

Conclusion & Perspectives

En sommes, il a été question pour nous de se préoccuper de la Compréhension par l'exploration et la visualisation des réseaux de neurones afin de concilier les différents points de vues acteurs et utilisateurs de ces réseaux afin de permettre leurs compréhension dans leurs différentes dimensions .

A cet effet nous avons proposé deux approche de solution :

1) la première étant l'Utilisation d'une approche d'outils adaptables (Moldable tools) permettant facilement aux différentes catégories d'utilisateurs de construire et d'adapter de moyens de visualisation et de compréhension des réseaux de neurones.

2) la deuxième quant à elle prévoyait l'Utilisation d'une démarche exploratoire proche du "literate programming" permet dynamiquement de lier texte narratif, code et visualisation.

Pour faciliter le travail, nous avons utilisé l'implémentation de ces deux approches qui existe déjà sur la plate forme GToolkit basé sur Pharo.

Par la suite nous avons utilisé les modèles d'implémentation de réseau de neurones réalisé par Alexandre Bergel en Pharo pour cette plate forme GToolkit afin de valider ces solutions.

Nous l'avons réalisé et nous avons présenté les résultats obtenue plus haut nous avons fait des comparaison avec des études existant plus particulièrement celui de l'article de *Fred Hohman, Minsuk Kahng, Robert Pienta, et Duen Horng Chau, "Visual Analytics in Deep Learning: An Interrogative Survey for the Next Frontiers" arXiv:1801.06889v3 [cs.HC] 14 May 2018.*

qui fourni une très bonne démarche expérimentale pour la visualisation des réseaux de neurones.

En perspective

nous comptons pour la suite de nos travaux :

- continuer de travailler sur un modèle de visualisation plus complexe, un outils qui s'adapte à tous les jeux de données.
- Nous aimerions également produire une application interactive avec des interface pour épargner au utilisateur de modèle d'avoir un contact avec le

code sources(une plate forme de visualisation comme Tensor flow Playground.

- Comment concilier les deux visions lorsqu'on développe un réseau de neurones: vision code vs vision architecturale?