CRÉATION DE RÉSEAUX DE NEURONES À L'AIDE DE NEURAL SIMULATION LANGUAGE

Corina CIOBANU[1*]
Anastasia GHERMAN[1]

[1]Universitatea Tehnica a Moldovei, Facultatea Calcuatoare, Informatică și Microelectronică, Departamentul Inginerie Software și Automatică, grupa FI 191, Chișinău, Republica Moldova

*Autorul corespondent: Ciobanu Corina ciobanu.corina@isa.utm.md

Résumé. *Dans cet article on se cible vers l'étude du fonctionnement du cerveau et les processus qui s'y déroulent à l'aide de théories et d'outils mathématiques qui ont été faite depuis quelques années, comment les technologies de l'information pourraient créer des outils pour augmenter le volume et les modalités des données neuroscientifiques. Ainsi, l'étude du fonctionnement des neurones et les étapes de modélisation sont les principaux pour la mise en œuvre de la simulation du fonctionnement des réseaux neuronales. .*

Les mots clé : *neuro-informatique, données, Neural Simulation Language, neurone, modélisation, simulation.*

Introduction

Les données sont à l'origine du progrès technologique, les données collectées en médecine deviennent de plus en plus complexes. La valorisation de ces données peut révéler de nouvelles hypothèses sur l'origine des maladies neurologiques et psychiatriques et en même temps contrecarrer le développement d'outils technologiques pour un meilleur diagnostic, prédiction et traitement des patients.

Étudier, analyser et obtenir des données en neuro-informatique est devenu possible grâce à de nombreuses recherches dans le domaine, aujourd'hui la collecte et l'analyse des données sont effectuées à l'aide du langage de programmation Neural Simulation Language (NSL), „ publié en 1989, qui a évolué au fil des décennies, le système d'origine a été écrit en langage C (NSL1) en 1989, en 1991 apparaît la deuxième version écrite en C ++, basée sur la technologie orientée sur objet. Les deux ont été développés à l'USC (Université de Californie du Sud) par Alfredo Wetzenfeld et Michael Arbib." [1]. La version actuelle de NSL 3 est une version complètement restructurée et supérieure aux précédentes en tant que système, ainsi que dans la manière dont ils effectuent la modélisation et la simulation, y compris la modularité et la concordance. NSL 3 comprend deux environnements différents, l'un en Java (NSLJ, développé à l'USC par l'équipe d'Amanda Alexander) et l'autre en C (NSLC, développé à ITAM au Mexique par l'équipe d'Alfredo Weitzenfeld). Le langage de simulation neuronale NSL fournit une plate-forme pour construire des architectures neuronales, en d'autres termes les modéliser et les exécuter appelées simulation.

Neural Simulation Language

Neural Simulation Language (NSL) est un langage de programmation qui permet aux neurologues de créer des réseaux virtuels de neurones. Cela leur a donné la possibilité de tester certaines hypothèses sur des modèles créés précédemment en modifiant certains paramètres et conditions de travail. Comment est-il possible de construire un réseau virtuel de neurones ? Le processus donné comprend 2 étapes principales : la modélisation et la simulation.

La modélisation est basée sur l'établissement de modèles mathématiques qui reflètent divers changements ou analysent la structure des neurones. „Cette étape est divisée en deux niveaux : modules et réseaux de neurones. Les modules sont la façon dont les neurones sont situés dans le système. Un modèle complet se compose des éléments suivants : 1) un ensemble de modules qui composent le modèle ; 2) les neurones qui composent chaque module ; 3) les

connexions entre les neurones ; 4) dynamique neuronale et 5) méthodes numériques pour résoudre des équations différentielles. Les modules sont disposés hiérarchiquement de sorte que chaque module peut comprendre des sous-modules" [2]. Le réseau neuronal englobe la façon dont les neurones interagissent les uns avec les autres. Ainsi, à chaque neurone peut correspondre 3 paramètres : entrée, potentiel d'action et combustion.

Les étapes de modélisation sont :

1) Déterminez le nombre de neurones impliqués, puis ils sont organisés en modules. Parce que chaque neurone est difficile à nommer, ils sont organisés en modules sous forme de tableaux bidimensionnels ;
2) Il détermine comment les neurones sont disposés dans l'espace ainsi que les ensembles de données de sortie et d'entrée (également appelés ports de données.

```
nslModule Name (arguments)
{
 structure
 behavior
}
```

La structure comprend les données saisies et le comportement inclut les opérations à exécuter (exécution, initialisation, achèvement).

3) Planifiez l'ordre dans lequel les modules sont exécutés et spécifiez la fréquence à laquelle les ports lisent et écrivent les données dans et hors des modules ;
4) Les relations entre les neurones sont établies, en tenant compte de la dynamique neuronale (les processus changeants qui ont lieu à l'intérieur du neurone et dans l'espace synaptique). Ceux-ci sont décrits par des équations différentielles ;
5) L'architecture du modèle est visualisée soit par la révision du code, soit par le biais du Schematic Capture System (SCS). Bien que la deuxième variante n'ait pas le même niveau de fonctionnalité que le code, elle offre une certaine clarté du fait que les modèles peuvent être visualisés ;
6) Le modèle est mis en œuvre.

La simulation commence avec un modèle déjà construit. Cette procédure comprend la saisie par l'utilisateur de paramètres, de valeurs et de modèles. Dans le même temps, la spécification du contrôle de simulation est impliquée, ainsi que la visualisation des graphiques et des images. Au cours de la simulation, il est important d'obtenir différents graphiques qui reflètent le comportement des neurones pour mieux comprendre comment le modèle réagit à certains moments. Le système de simulation comprend plusieurs sous-systèmes :

1) Contrôle des E / S - les aspects externes de la simulation par Script Interpreter et Window Interface sont traités ici ;
2) Le planificateur qui exécute les modèles et modules dans une certaine séquence ;
3) Compilateur - le code est compilé et la connexion est établie avec les bibliothèques NSL pour générer un fichier exécutable ;
4) Script Interpreter - spécifie les paramètres et dirige la simulation ;
5) Sortie graphique - contient des bibliothèques NSL qui affichent des graphiques, des canevas ou des cadres ;
6) Entrée graphique - dirige la simulation via les paramètres.

Les étapes de simulation sont :

1) Sélectionnez un modèle ;
2) L'interface de simulation s'ouvre ;
3) Contrôle de simulation où le modèle est exécuté ;
4) Visualisation
5) Introduire le code ;
6) Introduire les paramètres.

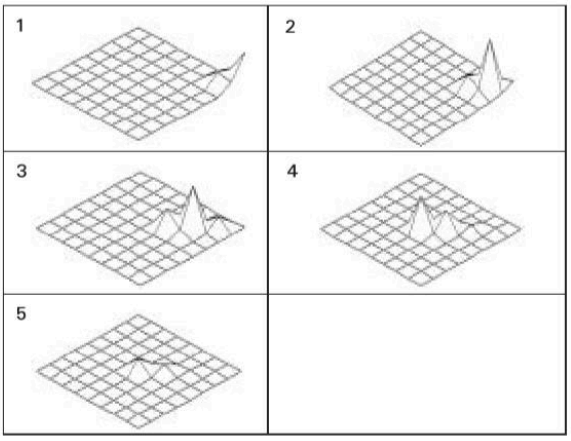

Figure 1. L'activité des cellules nerveuses

Sur la Figure 1 est représentée l'activité des cellules nerveuses après simulation des sous-modules du Coliculum Supérieur du modèle Crowley et Arbib.

Le modèle Hopfield

Les modèles déjà développés et exposés pour un accès libre sont stockés dans la bibliothèque NSL. Il existe de nombreux types de modèles bien connus qui étudient différents aspects du cerveau. „L'un d'eux est Hopfield qui se concentre sur la mémoire et l'association. Il se compose de 3 modules : entrée, hopfield lui-même et sortie" [2]. Ce modèle est basé sur la récurrence et en même temps il existe des attracteurs (états stables) qui permettent d'associer un modèle à d'autres modèles enregistré, appelés processus et mémoire associative. Les neurones envoient des signaux à d'autres neurones, mais aucun neurone ne peut recevoir un signal par lui-même. Une autre caractéristique du modèle est la mise à jour asynchrone, c'est-à-dire qu'une seule unité, choisie au hasard, peut changer son état à un certain moment. A l'intérieur du réseau se trouve une certaine énergie. Suite à des mises à jour asynchrones, l'énergie diminue. Le point où l'énergie a la valeur minimale est également appelé l'attracteur. Les neurones peuvent être organisés de telle sorte que leurs associations occupent ces points d'énergie minimum, et par conséquent le réseau recherche ces associations.

```
nslModel HopfieldModel ()
{
 private int size = 10 ;
 public Hopfield hopfield(size);
 public HopfieldInput in(size);
 public HopfieldOutput out(size);
 public void makeConn(){
   nslConnect(in.out,hopfield.pat);
   nslConnect(hopfield.mf,out.in);
 }
}
```

Public signifie que les données peuvent être consultées par n'importe qui, et privées sont limitées à un groupe particulier. Dans ces séquences, vous pouvez voir les 3 modules, et à la fin les connexions sont formées dans la fonction makeConn.

La simulation de ce modèle se compose de 2 étapes : la phase d'entraînement, où le poids synaptique est affecté aux valeurs souhaitées et la phase de roulement où l'état initial de chaque neurone est défini dans la configuration d'entrée à tester. „Ce modèle a été testé en mémorisant les lettres A, B, C, D et E. Les lettres ont été choisies à condition qu'elles diffèrent par leur forme pour une meilleure association. La phase de formation se limite au fait que le poids des connexions n'est pas appris, mais est ajusté directement à partir des modèles introduits" [2].

Une fois les lettres lues, la simulation est effectuée jusqu'à ce qu'une solution stable soit trouvée.Dans l'une des expériences, représenté dans la Figure 2, la lettre A a été introduite, avec quelques pixels supprimés. Malgré la mauvaise qualité d'image, le modèle a reconstruit la lettre.

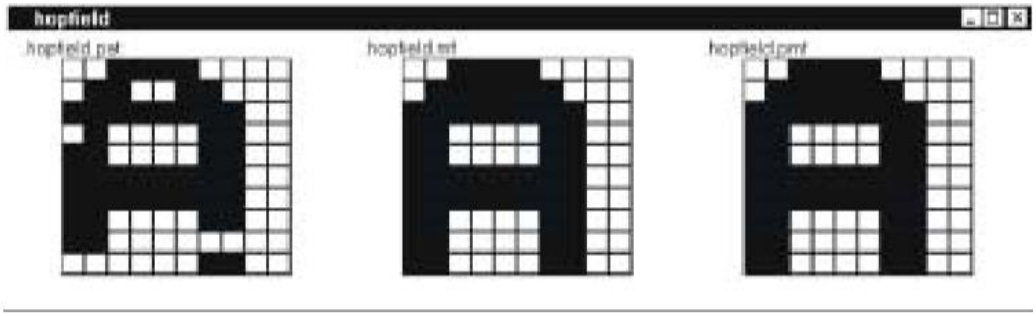

Figure 2. La simulation

Parfois, le modèle peut "mémoriser" des images qui ne sont pas dans l'ensemble de formation. C'est aussi ce qu'on appelle un faux état, dans lequel les neurones adoptent une trajectoire inattendue (l'énergie est bloquée dans des états indéfinis), ce qui dénote l'imperfection du modèle et même du langage NSL.

Conclusion :

Le langage de simulation neurale est un langage de programmation grâce auquel des recherches sont effectuées dans le domaine de la neuro-informatique. La complexité de ce langage de programmation révèle de la manière dont la recherche se déroule, toute recherche dans ce langage de programmation est effectuée en deux étapes : la modélisation et la simulation, qui à leur tour sont divisées en plusieurs sous-étapes, permettant une analyse détaillée, enregistrant des résultats étonnants qui parviennent à porter la médecine à un nouveau niveau. NSL est un programme de recherche et développement qui s'étend des études biologiques aux applications artificielles. La modélisation développe une architecture neuronale qui peut expliquer et reproduire les données expérimentales anatomiques et physiologiques. La plateforme de développement de réseaux neuronaux offre un environnement de modélisation et de simulation pour des réseaux neuronaux à grande échelle, qui, grâce à l'utilisation de modules, peuvent être interconnectés hiérarchiquement pour permettre la construction de modèles très complexes.

Nous remercions Mme Daniela Istrati, lecteur universitaire au Département Génie Logiciel et Automatique, Université Technique de Moldova, pour l'aide à l'élaboration de cet article.

Bibliographie :

1. Michael A. Arbib, Jeffrey S. Grethe „Computing The Brain: A Guide To Neuroinformatics", ACADEMIC PRESS, 2001. ISBN 0-12-059781-0.
2. Weitzenfeld, A., Arbib, M.A, Alexander, A. The Neural Simulation Language: A System for Brain Modeling, MIT Press, 2002. ISBN 0-262-73149-5. weitzenfeld.robolat.org/neural-simulation-language/?fbclid=IwAR23lGxKbmR0ktonk0hqGcLcgA9oy1SeH6nrcMQu8EtyXvOPnJOLoyA9xM0
3. Article publié par l'Institut du Cerveau et de la Moelle épinière sur la thème „La neuroinformatique: enjeux pour la recherche et la médecine de demain". https://sante.lefigaro.fr/article/la-neuroinformatique-enjeux-pour-la-recherche-et-la-medecine-de-demain/?fbclid=IwAR3l1u2O_LRrMH1fq0Kzk-FE2KXA_JNEKq36sJH6eEUfO1ItUs27dZju5bo