Projet Spiking SOM

Introduction

L'objectif de ce projet est de comprendre et de répliquer les résultats d'un article scientifique dans le domaine des neurosciences computationnelles.
Il s'agit de reproduire le comportement d'une carte auto-organisatrice (SOM) à l'aide de neurones impulsionnels.
Vous retrouverez dans cet article des modèles présentés en cours et vus en TP : modèle de neurone *leaky integrate-and-fire*, modèle de synapse avec des *fonctions alpha*, synapses excitatrices/inhibitrices, apprentissage (i.e. modification des poids synaptique) avec la règle *STDP*.
Le projet devra être réalisé à l'aide du simulateur *Brian2*.
Vous vous appuierez pour ce projet sur l'article de Rumbell et al. [1].
Dans votre présentation orale (15 mn de présentation et 5 mn de questions), vous veillerez à résumer le cadre de l'étude de l'article.
Quelques consignes :

- Vous utiliserez python comme langage de programmation et le simulateur Brian2.
- Vous pouvez installer le paquet *brian2tools* qui vous permettra de créer plus facilement des graphiques.
- Vous ne serez pas évalué sur la qualité du code. Faites donc au plus simple et au plus rapide.
- Testez régulièrement votre code de façon modulaire.
- Consultez la documentation ainsi que les tutoriels de Brian2 en cas de difficulté.
- Vous ne considérerez pas le terme de bruit blanc gaussien dans vos modèles neuronaux.

1 Implémentation de l'architecture (3h)

1. On commence par encoder le stimulus extérieur en tension en entrée pour la population u. Pour réaliser cela on implémentera une classe *ReceptiveField* dont le squelette est fourni. Complétez le squelette en vous basant sur l'article de Rumbell et al. [1]. pour trouver les valeurs des paramètres et sur les équations suivantes dérivées de l'article de Bohte et al. [2] :

$$\sigma = \frac{1}{\gamma}(I_{max} - I_{min})$$

avec $\gamma = 1.5$

pour $i \in 1, ..., N$ avec N étant le nombre de neurones utilisés pour encoder une valeur réelle :

$$\mu_i = I_{min} + \left(\frac{2i-2}{2} * \frac{I_{max} - I_{min}}{N-1}\right)$$

la fonction gaussienne utilisée prends la forme :

$$f(x) = e^{-(x-\mu)^2/2\sigma^2}$$

Testez ce que vous retourne la fonction *float_to_potential* avec un tableau numpy contenant une entrée normalisée (i.e. dont la valeur varie entre 0 et 1).

2. Ecrivez les équations rendant compte du comportement d'un neurone au sein de la population u dans un objet *u_ layer_ neuron_ equ* de type *multi-line string*. Ajoutez un terme I_{ext} sans dimension pour l'entrée extérieure. Pour le moment on ne tiendra pas compte de l'influence du neurone inhibiteur dans les équations.
3. Instanciez un groupe de neurones *u_ layer* avec le modèle de neurone *u_ layer_ neuron_ equ*. Initialisez I_{ext} avec le tableau de valeur retourné par la fonction *float_ to_ potential*.
4. Testez le comportement.
5. Ecrivez les équations rendant compte du comportement du neurone Inh dans un objet *inhibition_ neuron_ equ* de type *multi-line string*. Ajoutez dans les équations l'influence des synapses excitatrices et inhibitrices provenant du groupe de neurone u.
6. Instanciez le neurone inhibiteur *inhibition_ neuron* avec le modèle *inhibition_ neuron_ equ*.
7. Instanciez un objet *u2inh_ excitation* de type *Synapses* avec les règles présynaptiques et postsynaptiques adéquates ainsi que le modèle. Le groupe de neurones source est *u_ layer* et le groupe de neurones cible est *inhibition_ neuron*.
8. Connectez les synapses *u2inh_ excitation*.
9. Testez le comportement.
10. Instanciez un objet *u2inh_ inhibition* de type *Synapses* avec les règles présynaptiques et postsynaptiques adéquates ainsi que le modèle. Le groupe de neurone source est *u_ layer* et le groupe de neurones cible est *inhibition_ neuron*
11. Connectez les synapses *u2inh_ inhibition*.
12. Testez le comportement.
13. Modifiez les équations de la population u contenues dans *u_ layer_ neuron_ equ* de telle sorte à rendre compte de l'influence reçue du neurone inhibiteur *inhibition_ neuron*.
14. Instanciez un objet *inh2u_ inhibition* de type *Synapses* avec les règles présynaptiques et postsynaptiques adéquates ainsi que le modèle. Le groupe de neurone source est *inhibition_ neuron* et le groupe de neurones cible est *u_ layer*.
15. Connectez les synapses *inh2u_ inhibition*.
16. Testez le comportement.

2 Suite de l'implémentation de l'architecture (3h)

1. Ecrivez les équations rendant compte du comportement d'un neurone au sein de la population v (correspondant à la SOM) dans un objet *v_ layer_ neuron_ equ* de type *multi-line string*.
2. Instanciez un groupe de neurones *v_ layer* avec le modèle de neurone *v_ layer_ neuron_ equ*.
3. Instanciez un objet *v2v* de type *Synapses* avec les règles présynaptiques et postsynaptiques adéquates ainsi que le modèle. Le groupe de neurone source est *v_ layer* et le groupe de neurones cible est *v_ layer*.
4. Connectez les synapses *v2v*.
5. Instanciez un objet *u2v* de type *Synapses* avec les règles présynaptiques et postsynaptiques adéquates ainsi que le modèle. Le groupe de neurone source est *u_ layer* et le groupe de neurones cible est *v_ layer*.
6. Connectez les synapses *u2v*.
7. Testez le comportement de la SOM (*v_ layer*).
8. Testez le comportement de l'architecture complète en fournissant en entrée un tableau numpy comportant 2 valeurs réelles normalisées entre 0 et 1.

3 Résultats (3h)

1. Vous répliquerez les tests avec des données en deux dimensions présentés en section III-D [1].
2. Question bonus (2 points) : vous pouvez choisir de répliquer les tests de catégorisation avec le dataset *IRIS* (section III-F) ou le dataset *Wisconsin Breast Cancer Dataset* (section III-G). Vous n'avez pas à comparer vos résultats avec les autres approches (e.g. *k-Means*).

Références

[1] Rumbell, T., S. L. Denham et T. Wennekers: *A Spiking Self-Organizing Map Combining STDP, Oscillations, and Continuous Learning*. IEEE Transactions on Neural Networks and Learning Systems, 25(5) :894–907, mai 2014, ISSN 2162-237X.

[2] Bohte, S.M., H. La Poutre et J.N. Kok: *Unsupervised clustering with spiking neurons by sparse temporal coding and multilayer RBF networks*. IEEE Transactions on Neural Networks, 13(2) :426–435, mars 2002, ISSN 1941-0093.