Impact du bruit synaptique sur les performances des réseaux de cliques neurales

Eliott COYAC, Vincent GRIPON, Charlotte LANGLAIS

Electronics Department, Institut Mines-Telecom, Telecom Bretagne, CNRS UMR 6285 Lab-STICC, CS 83818 - 29238 Brest Cedex 3, France

prénom.nom@telecom-bretagne.eu

Résumé – Les réseaux de neurones s'inspirent du fonctionnement et de la structure du cerveau, et l'argument de la plausibilité biologique est souvent utilisé pour défendre ou critiquer un modèle par rapport à un autre. La littérature neurobiologique nous renseigne sur le fait que le cerveau est un système bruité, dans lequel les composants et leurs connexions sont non fiables. Pourtant, le comportement des réseaux de neurones face à ce bruit a rarement été étudié. Ce papier propose un modèle de bruit cérébral et analyse son impact sur les performances des réseaux de cliques neurales. Nous montrons que le bruit peut les améliorer, notamment en évitant des minima locaux. Nous analysons également l'impact de ce bruit sur les performances des réseaux de Hopfield.

Abstract – Artificial neural networks are inspired by biological neural networks present in the brain, and biological plausibility is often used as an argument to validate or criticize a neural network proposal. However, the brain is a system with a lot of interferences and the behaviour of neural networks with respect to this noise has not often been studied. This paper introduces a model to represent noise inside the brain, and studies how neural clique networks respond to that noise. It is shown that the noise can improve the neural clique network performance by avoiding local minima. We also show the impact of this noise on the widely-known Hopfield networks.

1 Introduction

Le cerveau a inspiré de nombreux modèles de réseaux de neurones. Parmi eux, les réseaux de Hopfield [5], les machines de Boltzmann [1], les réseaux de Willshaw [6] et plus récemment les réseaux de cliques neurales [3] implémentent des mémoires associatives. Ces mémoires, par contraste avec les mémoires indexées classiques, permettent d'accéder à un élément d'information stocké à partir d'une fraction de son contenu. Parmi les modèles de réseaux de neurones, deux grandes familles s'opposent. La première considère que l'information est stockée dans le poids des synapses reliant les neurones (e.g. Hopfield, Boltzmann). La seconde propose que l'information est stockée dans l'existence de connexions formant des motifs graphiques (e.g. Willshaw, réseaux de cliques neurales).

Réciproquement, ces réseaux ont souvent été utilisés pour modéliser la mémoire cérébrale. En tant que tels, il est possible d'imaginer qu'ils soient résistants face aux différentes contraintes du cerveau, comme les perturbations présentes. Des publications de la littérature neurobiologique [2] nous renseignent sur le fait que les connexions entre neurones biologiques sont non fiables. Pourtant peu d'entre elles s'intéressent au comportement des réseaux de neurones face à cette forme de bruit.

Dans ce papier, nous introduisons un modèle de bruit s'inspirant de celui présent dans le cerveau et s'appliquant aux réseaux de neurones artificiels. Nous analysons l'impact de ce bruit sur deux modèles de mémoires associatives : le réseaux de Hopfield et les réseaux de cliques neurales. Pour ce dernier, nous dérivons analytiquement sa performance après une itération de décodage. Puis nous montrons par simulation que les performances peuvent être améliorées pour certains paramètres de bruit, conformes aux valeurs biologiques.

Le reste du papier est organisé de la manière suivante. La section 2 introduit un modèle de bruit pour les réseaux de neurones artificiels. La section 3 présente les modèles de mémoires associatives de Hopfield et les réseaux de cliques neurales. La section 4 étudie l'impact du bruit proposé sur les performances de ces modèles.

2 Modèle de bruit dans le cerveau

Le cerveau est un système non fiable. Il y a plusieurs raisons à cela, des interférences produites par des neurones s'activant spontanément jusqu'à des causes chimiques. Dans ce papier nous nous intéressons à la faillibilité des synapses [7]. Chaque neurone possède un axone, s'attachant à de nombreux autres neurones par le biais de synapses. Un axone se connecte à un neurone par plusieurs synapses, et les synapses libèrent parfois des neurotransmetteurs quand le neurone d'origine est stimulé. La probabilité qu'une synapse libère des neurotransmetteurs peut varier de 10% à 80% [2].

Nous proposons un modèle où chaque neurone est connecté à un autre neurone par n_{syn} synapses. Chaque synapse est munie d'une probabilité p_{rel} de libérer des neurotransmetteurs lorsqu'elle est stimulée. Pour simplifier, nous prenons l'hypothèse que ces probabilités sont indépendantes. Par conséquent, lorsqu'un neurone émet un signal, les neurones auxquels il est

connecté reçoivent une stimulation suivant une loi binomiale $B(n_{syn}, p_{rel})$.

3 Mémoires associatives

3.1 Réseaux de Hopfield

Un réseau de Hopfield [5] est porté par un graphe complet reliant n neurones. Un tel graphe peut stocker des messages binaires ($\{-1, 1\}$) de longueur n. Chaque neurone est étiquetté de 1 à n, de sorte qu'un message à stocker corresponde à une valeur -1 ou 1 attribuée à chaque neurone du réseau. Les connexions entre les neurones ont chacune un poids qui vaut 0 lorsque le réseau est vide.

Lorsqu'un message m est stocké, le poids des connexions entre les neurones observant la même valeur pour m sont incrémentés de 1, les poids des autres connexions sont décrémentés. Formellement, si $\mathcal{M}$ est l'ensemble des messages binaires à stocker, alors la matrice d'adjacence $W(\mathcal{M})$ du réseaux de Hopfield correspondant est :

$$W(\mathcal{M}) \triangleq \sum_{m \in \mathcal{M}} m \cdot m^\top \ , \tag{1}$$

où $m^\top$ est la transposée de m.

Le réseau de Hopfield dans certaines conditions peut retrouver un message stocké à partir d'une fraction de son contenu en entrée. Nous appelons un $\tilde{m}$ la version partiellement effacée d'un message m en remplaçant certaines de ses coordonnées par 0. Pour retrouver m à partir de $\tilde{m}$ et de $W(\mathcal{M})$, les réseaux de Hopfield utilisent l'algorithme itératif suivant :

Data: $W(\mathcal{M})$,$\tilde{m}$
$v \leftarrow \tilde{m}$
Faire
  $w \leftarrow v$
  $v \leftarrow sgn(W(\mathcal{M}) \cdot w)$
Tant que $v \neq w$;
Rendre v;

Algorithm 1: Algorithme pour retrouver un message à partir d'une version effacée dans un réseau de Hopfield. La fonction sgn est la fonction qui à un entier associe son signe.

3.2 Réseaux de cliques neurales

Un réseau de cliques neurales [3] est porté par un graphe composé de c parties contenant chacune ℓ nœuds. Contrairement aux réseaux de Hopfield, les connexions ne sont pas pondérées. Une information est alors représentée dans le réseau sous la forme d'un motif complètement interconnecté (clique).

Formellement, les messages pouvant être stockés dans un tel réseau sont des vecteurs binaires ($\{0, 1\}$) de longueur $c \cdot \ell$ pouvant être découpés en c blocs de ℓ coordonnées, de sorte que dans chaque bloc n'apparaisse qu'un seul 1. De tels messages sont typiquement obtenus en utilisant un codage disjonctif complet.

La matrice d'adjacence du réseau de cliques neurales dans lequel est stocké l'ensemble $\mathcal{M}$ des messages est :

$$\mathfrak{W}(\mathcal{M}) \triangleq \max_{m \in \mathcal{M}} m \cdot m^\top \ , \tag{2}$$

où $\max$ est appliqué à chaque coefficient de la matrice indépendamment.

Dans le contexte des réseaux de cliques neurales, on appelle message effacé $\tilde{m}$ une modification d'un message stocké m où certains 1 représentant les noeuds actifs ont été transformés en 0.

Nous introduisons la fonction de *winner-takes-all* local suivante :

$$u(v)_i \triangleq \begin{cases} 1 & \text{si } v_i \text{ est le score maximal dans le bloc} \\ & \text{où se trouve la coordonnée } i \\ 0 & \text{sinon} \end{cases} \tag{3}$$

Pour retrouver m à partir de $\tilde{m}$ et $\mathfrak{W}(\mathcal{M})$, on utilise l'algorithme suivant :

Data: $\mathfrak{W}(\mathcal{M}), \tilde{m}$
$v \leftarrow \tilde{m}$
Faire
  $w \leftarrow v$
  $v \leftarrow u(W(\mathcal{M}) \cdot w)$
Tant que $v \neq w$;
Rendre v;

Algorithm 2: Algorithme pour retrouver un message à partir d'une version effacée dans un réseau de cliques neurales.

4 Impact du bruit sur les performances

4.1 Cas des réseaux de Hopfield

La Figure 1 montre la comparaison des performances obtenues par un réseau de Hopfield selon si le bruit introduit dans la section 2 est pris en compte ou non. Ces réseaux contiennent $n = 2048$ neurones et sont capables de stocker environ 100 messages de façon à pouvoir les retrouver lorsque la moitié des coordonnées sont effacées. Les paramètres pour le modèle de bruit sont : $n_{syn} = 10$ et $p_{rel} = 50\%$. Ces paramètres sont choisis arbitrairement mais justifiables au niveau biologique comme discuté dans la section 5.

Nous observons un léger déclin en performance lorsque le bruit est appliqué. Le fait que les performances ne soient pas impactées de façon significative est probablement dû au grand nombre de signaux s'échangeant entre les nœuds, moyennant les effets du bruit.

4.2 Réseaux de cliques neurales

Le comportement des réseaux de clique neurales face au bruit a déjà étudié, mais pour un bruit qui est la conséquence d'une implémentation électronique [4]. Dans notre cas, chaque connexion d'un nœud s avec un nœud activé ajoute à son score la valeur d'une variable aléatoire suivant la loi binomiale mentionnée dans la section 2.

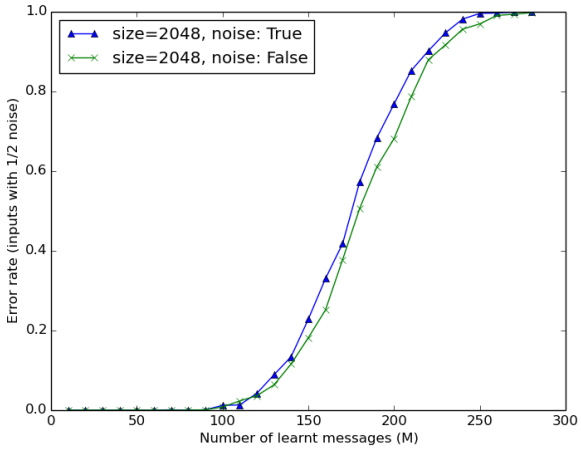

FIGURE 1 – Résultats simulés sur un réseau de Hopfield de 2048 nœuds, avec ou sans bruit et $\frac{1}{2}$ des données effacées.

4.2.1 Après une itération

Dans cette section, nous analysons la probabilité de retrouver un message partiellement effacé après une itération de l'algorithme 2.

Soit un réseau à cliques ayant stocké M messages. Notons c_k le nombre de coordonnées non effacées (1 non transformés en 0) et c_e le nombre de coordonnées effacées ($c_e = c - c_k$). Les performances dépendent grandement de la densité du réseau, c'est-à-dire le rapport entre le nombre de connexions utilisées et le nombre de connexions maximum. Dans le cas où les messages stockés sont indépendamment et uniformément distribués, celle-ci est définie par [3] :

$$d = 1 - \left(1 - \frac{1}{\ell^2}\right)^M. \tag{4}$$

Considérons un bloc effacé dans le message $\tilde{m}$. On appelle le nœud correspondant à la coordonnée effacée le nœud correct, il est noté s_0 et son score est n_{s0}. Le score d'un nœud s'apparente au nombre de synapses qui libèrent des neurotransmetteurs à destination du neurone représenté par ce nœud. Un nœud connecté à i autres nœuds actifs (nœuds dont la coordonnée est à 1) peut donc obtenir un score entre 0 et $i \cdot n_{syn}$. Le nœud correct est bien sûr connecté aux c_k autres nœuds non-effacés. Ainsi, pour la première itération, pour $x \in [0, n_{syn} \cdot c_k]$, nous avons :

$$\begin{aligned} P(n_{s_0} = x) &= P\left(B(n_{syn} \cdot c_k, p_{rel}) = x\right) \\ &= pmf(x, n_{syn} \cdot c_k, p_{rel}) \end{aligned} \tag{5}$$

où pmf est la fonction de masse de la loi binomiale. En effet la somme de c_k variables aléatoires suivant une loi binomiale $B(n_{syn}, p_{rel})$ est $B(n_{syn} \cdot c_k, p_{rel})$. Nous avons la fonction de probabilité du score que le nœud correct obtient. Pour déterminer cette fonction pour un nœud incorrect, nous devons tout d'abord savoir à combien de nœuds corrects non-effacés ce nœud est connecté. La probabilité que le noeud incorrect soit connecté à i noeuds corrects non-effacés est égale à :

$$P_E(i) = \binom{c_k}{i} d^i (1-d)^{c_k - i}. \tag{6}$$

Ainsi, la probabilité qu'un nœud incorrect (noté s) obtienne un score inférieur ou égal à $x_0 \in [0, n_{syn} \cdot c_k]$ est

$$P(n_s \leqslant x_0) = \sum_{x=0}^{x_0} \sum_{i=0}^{c_k} P_E(i)\ pmf(x, n_{syn} \cdot i, p_{rel}). \tag{7}$$

À l'aide de ces formules, nous pouvons écrire la probabilité que le nœud correct soit parmi les nœuds avec le plus haut score dans sa partie du graphe :

$$P_{succ}(s_0) = \sum_{x_0=0}^{n_{syn} \cdot c_k} P(n_{s_0} = x_0) P(n_s \leqslant x_0)^{\ell - 1}. \tag{8}$$

La probabilité de succès globale, c'est-à-dire que dans toutes les parties du graphe le nœud correct soit parmi les nœuds avec le plus haut score de cette partie est $P_{succ} = P_{succ}(s_0)^{c_e}$. Le taux d'erreur est $1 - P_{succ}$.

Cette approche n'est pas assez précise. À l'itération finale, un nœud unique par partie du graphe a besoin d'être choisi, pour obtenir un résultat final comportant c nœuds. C'est-à-dire, lorsque plusieurs nœuds partagent le plus haut score dans une partie du graphe, un nœud doit être choisi. Il y a une chance $\frac{1}{k+1}$ de trouver le bon nœud si k est le nombre de nœuds incorrects partageant le meilleur score avec le nœud correct.

Pour prendre cela en compte, $P_{succ}(s_0)$ est réécrite. Tout d'abord on introduit la probabilité de choisir le bon nœud si son score est x_0 :

$$\begin{aligned} P_{succ}(s_0 | n_{s_0} = x_0) = \sum_{k=0}^{l-1} \frac{1}{k+1} \binom{l-1}{k} P(n_s = x_0)^k \\ \times P(n_s < x_0)^{\ell - 1 - k} \end{aligned} \tag{9}$$

et

$$P_{succ}(s_0) = \sum_{x_0=0}^{n_{syn} \cdot c_k} P(n_{s_0} = x_0) P_{succ}(s_0 | n_{s_0} = x_0). \tag{10}$$

Ces formules ont été testées par une simulation, et comme montré sur la figure 2 les deux courbes théorique et simulée sont très proches. La différence peut être expliquée par le fait que l'on a fait l'hypothèse de l'indépendance des connexions entre elles.

4.2.2 Après plusieurs itérations

Condition d'arrêt Il s'agit d'abord de définir une condition d'arrêt pour l'algorithme. En effet, le bruit implique une absence d'état stable dans l'algorithme et un nouveau critère d'arrêt pour l'algorithme doit être choisi. On choisit de limiter le nombre d'itérations à 100 pour limiter le temps d'exécution. De plus, l'algorithme s'arrête si le résultat est stable après n_{st} itérations où n_{st} est choisi au préalable. Des tests montrent que

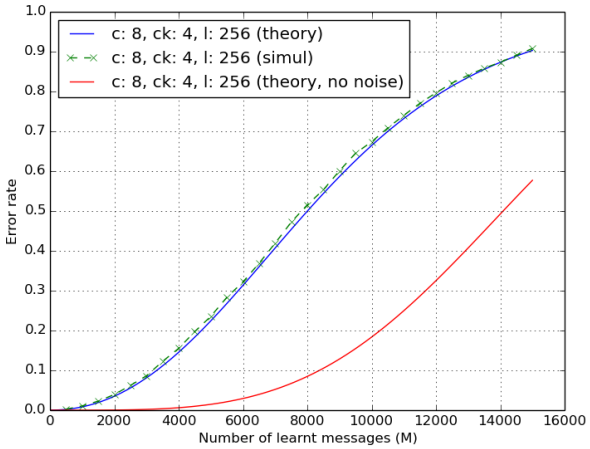

FIGURE 2 – Résultats théoriques et simulés après une itération avec $c = 8, c_k = 4, \ell = 256$.

pour un nombre très petit de messages, $n_{st} = 4$ est meilleur, mais ensuite $n_{st} = 3$ est préférable jusqu'à un taux d'erreur de 20%. Nous choisirons donc cette valeur pour les simulations suivantes.

Comme l'algorithme est itératif, il est très difficile d'obtenir une dérivation analytique des performances après plusieurs itérations. Nous proposons donc d'étudier l'impact du bruit sur les performances à l'aide de simulations.

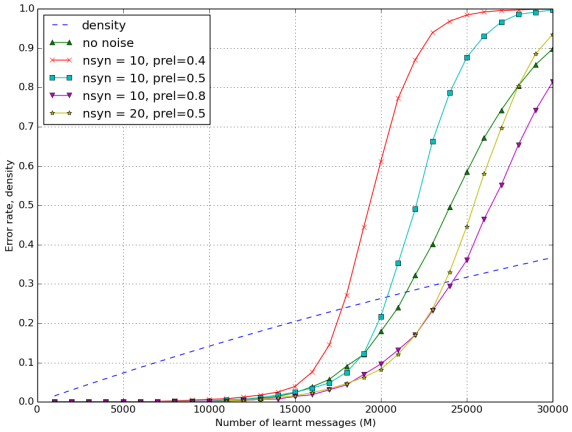

FIGURE 3 – Résultats simulés sur plusieurs itérations avec p_{rel} de 0.4 à 0.8 et $c = 8, c_k = 4, \ell = 256$.

Comme la figure 3 le montre, le bruit peut avoir des effets bénéfiques pour le réseau de cliques neurales. Pour $p_{rel} = 0.5$, une amélioration est observée pour les taux d'erreurs inférieurs à 10%. Pour $p_{rel} = 0.8$, il y a une nette amélioration pour tous les taux d'erreurs envisageables.

Ces résultats peuvent être attribués au fait d'éviter les minima locaux grâce au bruit, de la même manière que l'algorithme du recuit simulé.

5 Discussion et conclusion

Certains critères dans ce papier ont été choisis de manière arbitraire, comme le nombre de contacts synaptiques par connexion entre deux neurones ou la probabilité d'une synapse de libérer des neurotransmetteurs. 50% ou 80% pour cette probabilité peut sembler un chiffre élevé, mais d'après [2] cette probabilité s'ajuste en fonction des besoins du réseau. De plus, choisir $n_{syn} = 20$ au lieu de 10 est plus efficace que d'augmenter p_{rel} à 80% dans le cadre des réseaux de cliques neurales d'après la figure 3. Enfin, l'hypothèse d'évènements indépendants dans le temps a été faite dans ce papier, mais il est facile d'imaginer qu'une synapse a plus de probabilité de libérer des neurotransmetteurs pour une stimulation si elle n'en a pas émis à la stimulation précédente, et inversement [2]. Ceci réduirait la variance du bruit.

Dans cet article, le bruit a été introduit de manière artificielle et rajoute un coût en temps de calcul lors des simulations. Une architecture réelle possédant un bruit naturellement similaire à celui étudié ici pourrait au contraire en tirer pleinement parti pour diminuer la complexité du circuit et en même temps améliorer les performances.

Remerciements

Ce travail a été partiellement financé par le Conseil de Recherche Européen par le biais du programme *European Union's Seventh Framework Programme (FP7/2007-2013) / ERC grant agreement* n° 290901.

Références

[1] D. H. Ackley, G. E. Hinton, and T. J. Sejnowski. A learning algorithm for boltzmann machines*. *Cognitive science*, 9(1) :147–169, 1985.

[2] T. Branco and K. Staras. The probability of neurotransmitter release : variability and feedback control at single synapses. *Nature Reviews Neuroscience*, 10(5) :373–383, 2009.

[3] V. Gripon and C. Berrou. Sparse neural networks with large learning diversity. *Neural Networks, IEEE Transactions on*, 22(7) :1087–1096, 2011.

[4] F. Leduc-Primeau, V. Gripon, M. Rabbat, and W. Gross. Cluster-based associative memories built from unreliable storage. In *ICASSP*, pages 8370–8374, May 2014.

[5] C.-Y. Liou and S.-K. Yuan. Error tolerant associative memory. *Biological Cybernetics*, 81(4) :331–342, 1999.

[6] D. J. Willshaw, O. P. Buneman, and H. C. Longuet-Higgins. Non-holographic associative memory. *Nature*, 1969.

[7] A. Zador. Impact of synaptic unreliability on the information transmitted by spiking neurons. *Journal of Neurophysiology*, 79(3) :1219–1229, 1998.