Réseaux de Clusters de Neurones Restreints

Robin DANILO[1], Vincent GRIPON[2], Philippe COUSSY[1], Laura CONDE-CANENCIA[1], Warren J. GROSS[3]

[1]Université de Bretagne Sud 4 Rue Jean Zay, 56100, Lorient, France

[2]Telecom Bretagne 655 Avenue du Technopole, 29200, Plouzané, France

[3]McGill University 845 Rue Sherbrooke O, Montréal, QC H3A 0G4, Canada

robin.danilo@univ-ubs.fr, vincent.gripon@telecom-bretagne.eu
philippe.coussy@univ-ubs.fr laura.conde-canencia@univ-ubs.fr
warren.gross@mcgill.ca

Résumé – Les mémoires associatives sont capables de restituer un message précédemment stocké lorsqu' une version incomplète de celui-ci leur est présentée. Un modèle de mémoire associative basé sur des neurones et des connexions binaires, nommé Réseau de Clusters de Neurones (RCN), a été récemment introduit. Les performances de ce modèle chutent lorsque les messages stockés suivent une loi de distribution non-uniforme. L'objectif de cet article est de proposer un nouveau modèle de mémoire associative inspiré des RCNs et des machines de Boltzmann restreintes, afin de diminuer la vulnérabilité à la non-uniformité des messages. Une mise en œuvre matérielle de ce modèle est également présentée dans cet article. Cette dernière permet de multiplier le nombre de messages stockés par 3, avec une augmentation de la complexité matérielle de seulement 40%.

Abstract – Associative memories are capable of retrieving a message previously stored when an incomplete version of this message is presented. A model of associative memory based on binary neurons and binary connections, named Clustered Neural Network, has been recently introduce. The performance of this model drops when the stored message distribution is non-uniform. The goal of this paper, is to propose a new model of associative memory inspired by Clustered Neural Network and Restricted Boltzmann Machine, in order to decrease the vulnerability to non-uniform distribution.In addition, a fully parallel hardware design of the model. The proposed implementation multiplies the number of stored messages by a factor of 3 with an increase of complexity of 40%.

1 INTRODUCTION

Les mémoires associatives permettent d'accéder aux données stockées à partir d'une version partiellement effacée ou bruitée de ces données. Ces modèles de mémoire stockent des messages (ou "pattern") en les superposant les uns sur les autres au sein du même espace mémoire. Les performances de ces modèles sont évaluées à l'aide de la diversité, c'est à dire du nombre maximal de messages pouvant être stockés, tout en maintenant la probabilité de ne pas retrouver un message sous un certain seuil.

Un modèle de mémoire associative nommé Réseau de Clusters de Neurones (RCN) [1, 2], a permis de lever les limitations des précédents modèles de mémoire associative tel que le modèle d'Hopfield en terme de diversité lorsque la distribution des messages suit une loi uniforme. Cependant comme pour beaucoup de modèle de mémoire associative [3], la diversité chute lorsque la distribution des messages ne suit pas une loi uniforme. En effet, la proximité entre les messages rend plus difficile le processus de restitution.

Cet article propose d'appliquer les principes utilisés dans les Machines de Boltzmann Retreintes [4, 5] au RCN afin de diminuer la vulnérabilité aux distributions non uniformes. Le réseau résultant est nommé RCN Restreint (R-RCN). Cet article présente également une mise en œuvre matérielle massivement parallèle de ce nouveau modèle.

Cet article est organisé comme suit : la Section II présente le RCN. En Section III nous présentons le nouveau modèle et sa mise en œuvre matérielle. La Section IV est dédiée aux résultats de simulation et à la comparaison des architectures de notre modèle avec celles d'un RCN classique. Enfin, la Section V présente les conclusions et perspectives de notre travail.

2 RESEAU DE CLUSTERS DE NEURONES

2.1 Principes

Afin de stocker un ensemble de messages noté $\mathcal{M}$, où chaque message $m \in \mathcal{M}$ est composé de c symboles $m_0, \ldots, m_{c-1}$ sur un alphabet de taille ℓ, un RCN consiste en un réseau de $c \times \ell$ neurones binaires répartis en c parties (nommés clusters) chacune contenant ℓ neurones. Chaque neurone d'un cluster peut être connecté à l'aide d'une connexion binaire avec n'importe lequel des neurones situés dans un autre cluster. Ces connexions binaires sont stockées dans une matrice d'adjacence $W_A(\mathcal{M})$. Durant la phase de stockage, chaque message m est stocké dans le réseau en mettant à jour $W_A(\mathcal{M})$. Durant la phase de restitution, $W_A(\mathcal{M})$ est utilisée pour retrouver un message m à partir d'une version incomplète de celui-ci. A cause de la structure du réseau, le coût mémoire pour stocker $W_A(\mathcal{M})$ est calculé de la manière suivante :

$$cost_{CNN} = \frac{c(c-1)\ell^2}{2}. \quad (1)$$

2.2 Loi de distribution non uniforme

Les RCNs offrent une diversité maximale lorsque les messages stockés suivent une loi de distribution uniforme. Cependant les données naturelles (images, mesure biologique...) présentent rarement cette propriété. La non-uniformité de la loi de distribution des messages conduit à l'apparition plus fréquente de certains symboles par rapport aux autres entrainant une surutilisation de certains neurones.

La distribution non-uniforme des messages conduit à une diversité bien inférieure à celle obtenue dans un cas uniforme. Ce phénomène a été étudié dans [6] et des stratégies ont été proposées pour gérer les lois de distribution non-uniforme. Lorsque aucun a priori existe sur la distribution, la meilleure de ces stratégies consiste à augmenter le nombre de neurones par cluster. Ainsi plusieurs neurones sont associés à un même symbole. Ces neurones sont sélectionnés aléatoirement lors de la phase de stockage. Lors de la phase de restitution, l'ensemble des neurones associés au symbole est activé.

3 RESEAU DE CLUSTERS DE NEURONES RESTREINT

Dans cette section nous présentons une nouvelle stratégie inspirée des machines de Boltzmann restreintes permettant d'augmenter la diversité lorsque la distribution des messages suit une loi non-uniforme. Cette stratégie consiste à ajouter une couche cachée au réseau composée de plusieurs clusters et d'autoriser uniquement des connexions entre la couche d'entrée et la couche cachée. Le réseau résultant est appelé, RCN Restreint (RCN-R).

3.1 Principes

Le nombre de clusters cachés c^h et le nombre de neurones par cluster caché l^h doivent être choisis par le concepteur. Ces paramètres dépendent des caractéristiques de l'application (diversité attendue). Les neurones de la couche d'entrée et de la couche cachée sont notés n^h_{ij} et n^{in}_{ij} respectivement. Les connexions sont stockées dans une matrice d'adjacence $W_R(\mathcal{M})$, par convention, $w_{(ij)^{in}(i'j')^h} = 1$ signifie qu'une connexion binaire entre n^{in}_{ij} et $n^h_{i'j'}$.

Durant le processus de stockage, chaque nouveau message m^{in} est associé avec un message caché m^h généré aléatoirement. m^{in} conduit à l'activation d'un neurone dans chaque cluster de la couche cachée tandis que m^h conduit à l'activation d'un neurone dans chaque cluster de la couche cachée. Les connexions entre les neurones actifs de la couche d'entrée et de la couche cachée sont ensuite stockés dans $W_R(\mathcal{M})$. La Fig. 1 représente un RCN-R durant le stockage du message d'entrée $m^{in} = 0, 1, 3, 0$, le message caché $m^h = 0, 2, 0, 3$ lui a été associé, les connexions stockées dans le réseau sont représentées par les lignes vertes.

Durant la phase de restitution, un message d'entrée incomplet $\tilde{m}^{in}$ est présenté au réseau conduisant à l'activation de neurones dans les clusters d'entrées. Ensuite, un processus itératif est appliqué afin de retrouver le message complet. Ce processus se divise en deux étapes :

1. Activation des neurones de la couche cachée à partir des neurones de la couche d'entrée
2. Activation des neurones de la couche d'entrée à partir des neurones de la couche cachée

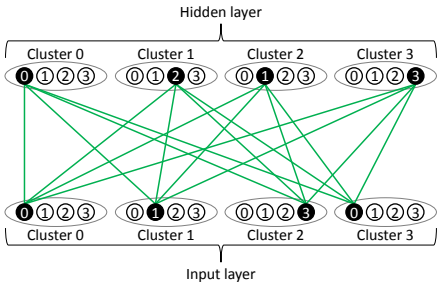

FIGURE 1 – Représentation graphique d'un RCN-R durant la phase de stockage du message d'entrée $m^{in} = (0, 1, 3, 0)$

Les neurones sont activés à l'aide des deux fonctions booléennes :

$$v^{h\ t+1}_{ij} = \bigwedge_{i'} \left(\bigvee_{j'} \left(w_{(i'j')^{in}(ij)^h} \wedge v^{in\ t}_{i'j'} \right) \vee d^{in\ t}_{i'} \right) , \quad (2)$$

où $d^{in\ t}_{i'} = \overline{\bigwedge_{j'} v^{in\ t}_{i'j'}}$,

$$v^{in\ t+1}_{ij} = \bigwedge_{i'} \left(\bigvee_{j'} \left(w_{(ij)^{in}(i'j')^h} \wedge v^{h\ t}_{i'j'} \right) \right) . \quad (3)$$

Notons qu'après la première étape, chaque cluster caché possède au moins un neurone actif. A cause de la structure du réseau, le coût mémoire du stockage de $W_R(\mathcal{M})$, est donné par :

$$cost_{R-CNN} = c.\ell.c^h.\ell^h. \quad (4)$$

3.2 Architecture proposée

Une mise en œuvre matérielle massivement parallèle est présentée dans cette section. Tout d'abord la génération aléatoire des messages de la couche cachée est expliquée. Puis les modules de stockage et de restitution sont décrits.

3.2.1 Génération des Messages Aléatoires

Les messages générés pour la couche cachée doivent suivre une loi de distribution uniforme. Pour cela, nous utilisons l'algorithme proposé dans [7]. Dans cet algorithme, plutôt que de diviser l'ensemble des neurones de la couche cachée en clusters de taille équivalente, la taille de chaque cluster est choisie pour être relativement premier avec la taille de chacun des autres clusters. Relativement premier signifie que le plus grand diviseur commun est égal à 1. La taille du cluster i de la couche cachée est ℓ^h_i et la valeur assignée à ce cluster est égale à m^h_i. Lors du stockage du premier message m^{in} de $\mathcal{M}$, la valeur 0 est assignée à chaque cluster de la couche cachée. Ensuite, pour chaque nouveau message, les valeurs assignées à chaque cluster de la couche cachée sont calculées de la manière suivante :

$$m^{h\ t+1}_i = (m^{h\ t}_i + 1) \pmod{\ell^h_i}. \quad (5)$$

Les intérêts d'un tel algorithme sont :

– il produit de manière automatique des messages suivant une loi de distribution uniforme,

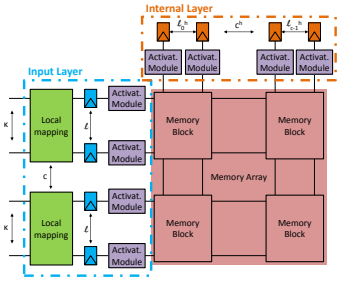

FIGURE 2 – Schéma simplifié de l'architecture générale

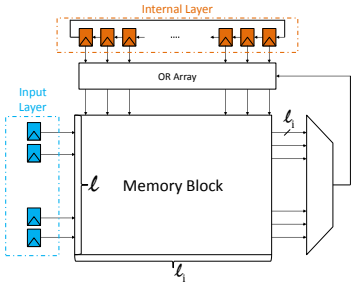

FIGURE 3 – Diagramme block d'un module de stockage entre un cluster de couche d'entrée et le i-ème cluster de la couche cachée

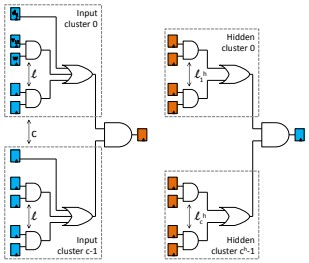

FIGURE 4 – Diagramme logique des modules d'activation de la couche cachée à gauche et de la couche d'entrée à droite

- il maximise la distance de Hamming entre les messages produits,
- il est facile à mettre en œuvre à l'aide de registres à décalage.

3.2.2 Architecture Générale

La Fig. 2 montre l'architecture générale du système. Ce dernier est composé de deux couches de neurones (couche d'entrée, couche cachée) chacune divisée en c et c^h clusters respectivement. A la fois durant la phase de stockage et de restitution, des messages de taille $\kappa.c$ ($\kappa = log_2(\ell)$) sont découpés en c vecteurs de taille κ et fournis aux Modules de Mapping Locaux qui réalisent un encodage one-hot afin de produire des vecteurs de taille ℓ bits dans lesquels seul un bit est mis à un. Ces c vecteurs de taille ℓ sont ensuite stockés dans les registres d'états des neurones de la couche d'entrée. $W_R(\mathcal{M})$ est stocké à l'aide de $c \times c^h$ blocs mémoire composés de registres (flip-flops) et permettant ainsi un accès parallèle.

3.2.3 Module de Stockage

Durant la phase de stockage, chaque message d'entrée est associé avec un message caché, généré à l'aide de l'algorithme proposé dans [7]. Pour générer ces messages, chaque cluster de la couche cachée comprend un registre à décalage dont la taille dépend de la taille ℓ_i^h du cluster. Lors du stockage du premier message, chaque registre à décalage a un seul registre mis à un. Après le stockage de chaque nouveau message, le bit à un est décalé.

La Fig. 3 représente le module de stockage entre un cluster de la couche d'entrée et i-ème cluster de la couche cachée. Le cluster de la couche d'entrée est composé d'un ensemble de ℓ neurones tandis que le cluster de la couche cachée est composé d'un registre à décalage de taille ℓ^h. Entre eux, un bloc mémoire de taille $\ell \times \ell_i^h$ dans lequel chaque ligne stocke les connexions d'un neurone du cluster de la couche d'entrée avec la totalité des neurones du cluster de la couche visible. Durant le stockage d'un nouveau message, la ligne du neurone actif est sélectionnée à l'aide du multiplexeur et une aire de OU est utilisée pour accumuler la valeur du registre à décalage avec les connexions précédentes.

3.2.4 Module d'Activation

Durant la phase de restitution, les modules d'activation sont utilisés pour calculer les valeurs des registres d'état des neurones de la couche cachée et de la couche d'entrée. La Fig. 4 représente un module d'activation pour la couche cachée (à gauche) et un module d'activation pour la couche d'entrée (à droite). Ces modules d'activation mettent en œuvre les fonctions booléennes Eq. 2 et 3.

4 RESULTATS

Dans cette section nous comparons le RCN-R avec le RCN en termes de coût mémoire pour une diversité donnée. Ensuite, l'architecture proposée est comparée avec une mise en œuvre parallèle du RCN proposée dans [8].

4.1 Restitution de messages

La base de données Yeast est utilisée pour comparer les deux modèles, RCN-R et RCN. Cette base de données est constituée de 1484 instances ayant chacune 8 attributs dont deux constantes. Nous ne conservons que les 6 attributs non constants que nous quantifions sur 64 intervalles. Chaque instance peut ainsi être vue comme un message de 6 symboles ayant un alphabet de taille 64. Les 1484 messages de la base sont tout d'abord appris dans le réseau testé (RCN-R ou RCN), puis un message m est sélectionné au hasard et deux symboles sont effacés aléatoirement pour obtenir $\tilde{m}$. $\tilde{m}$ est ensuite envoyé au réseau testé qui a pour charge de retrouver m. Cette opération est répétée 10000 fois et le taux d'erreur est calculé (nombre de messages non retrouvés divisé par le nombre de tests).

Pour le RCN, la configuration minimale pour apprendre des messages issus de la base Yeast, est un nombre de clusters égal à 6 et des tailles de clusters égales à 64. Lorsque les 1484 messages de la base Yeast sont stockés dans le réseau, cette configuration offre un taux élevé d'erreurs. Le moyen utilisé pour

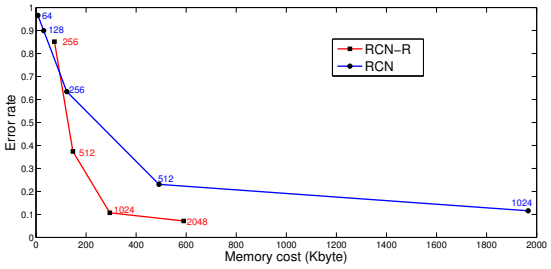

FIGURE 5 – Taux d'erreur en fonction du coût mémoire pour différentes configurations de RCN et de RCN-R

	RCN [8]	RCN-R
Registers	1.36×10^6	1.63×10^6 (+20%)
Storing	0.97×10^6	1.17×10^6 (+20%)
Retrieving	0.62×10^6	1.41×10^6 (+127%)
Total	2.95×10^6	4.01×10^6 (+43%)

TABLE 1 – Estimation de la complexité en portes NAND eq.

diminuer le taux d'erreurs est l'augmentation de la taille des clusters. Pour le RCN-R, le nombre et la taille des clusters de la couche d'entrée sont fixsé à 6 et 64 respectivement. Le nombre de clusters de la couche cachée est fixé à 6. Différentes configuration de RCN-R sont testées et correspondent à différentes tailles de clusters dans la couche cachée.

La Fig. 5 montre le taux d'erreur en fonction du coût mémoire pour les différentes configurations de réseau testées. Le coût mémoire est calculé pour les RCNs et RCN-Rs à l'aide des Eq. 1 et 4 respectivement. Les tailles des clusters des RCNs sont signalées sur la figure. Pour les RCN-Rs, les clusters de la couche cachée n'ayant pas tous la même taille, la taille moyenne des clusters de la couche cachée est signalée sur la figure. Nous pouvons voir que les RCN-R offrent un taux d'erreur plus faible pour un coût mémoire plus faible.

4.2 Complexité matérielle

Nous comparons maintenant la complexité matérielle de l'architecture proposée avec une mise en œuvre matérielle de RCN proposée dans [8]. Cette comparaison est faite en portes NAND équivalentes (basée sur une bibliothèque STMicroelectronics 90nm). Deux configurations équivalentes de réseau sont choisies : un RCN avec $c = 6$ et $\ell = 64$ et un RCN-R avec $c = c^h = 6$ et $\ell = \ell^h = 64$. Les deux réseaux ont besoin de 4 cycle au maximum pour retrouver un message.

Le Tab. 1 présente la complexité matérielle pour les deux mises en œuvre. Le coût total est divisé en trois parties : le coût des registres utilisés pour la zone mémoire et pour les neurones, le coût des modules de stockage et le coût des modules d'activation.

Si les deux réseaux offrent les mêmes performances pour des lois de distribution des messages uniformes, le surcoût payé pour le RCN-R est compensé par de meilleures performances sur la base de données Yeast. Avec le RCN , seulement 25 messages peuvent être stockés en maintenant un taux d'erreur inférieur à 0.01. Avec le RCN-R 76 messages peuvent être stockés (diversité multipliée trois) pour le même taux d'erreur.

5 CONCLUSION

Nous avons introduit dans cet article un nouveau modèle de mémoire associative appelé Réseau de Clusters de Neurones Restreint (RCN-R). Ce modèle est inspiré du Réseau de Clusters de Neurones (RCN) et utilise les principes de la Machine de Boltzmann Restreinte afin d'augmenter la diversité dans le cadre de messages suivant des lois de distribution non-uniformes. Nous avons réalisé des comparaisons entre notre modèle et le RCN en termes d'utilisation de mémoire et taux d'erreurs afin de conclure que le R-CNN offre de meilleures performances que le RCN pour une utilisation de mémoire moindre.

Dans cet article, nous avons également proposé une mise en œuvre matérielle du RCN-R complètement parallèle. Cette mise en œuvre augmente la complexité d'environ +43% par rapport à une implémentation d'un RCN équivalent. Cependant, la diversité a été multipliée par un facteur de 3.

Enfin, l'algorithme de stockage que nous proposons est simple et facile à implanter. Il consiste à associer un message caché, uniformément distribué, avec le message d'entrée suivant une loi non uniforme. Nous envisageons par la suite l'étude d'algorithmes de stockage plus intelligents qui choisissent par eux-même les messages cachés en fonction de la distribution des entrées.

Références

[1] V. Gripon and C. Berrou, "A simple and efficient way to store many messages using neural cliques," in *Proceedings of IEEE Symposium on Computational Intelligence, Cognitive Algorithms, Mind, and Brain*, Paris, France, April 2011, pp. 54–58.

[2] ——, "Sparse neural networks with large learning diversity," *IEEE Transactions on Neural Networks*, vol. 22, no. 7, pp. 1087–1096, July 2011.

[3] A. Knoblauch, G. Palm, and F. T. Sommer, "Memory capacities for synaptic and structural plasticity," *Neural Computation*, vol. 22, no. 2, pp. 289–341, 2010.

[4] P. Smolensky, "Information processing in dynamical systems : Foundations of harmony theory," 1986.

[5] G. Hinton, "Training products of experts by minimizing contrastive divergence," *Neural computation*, vol. 14, no. 8, pp. 1771–1800, 2002.

[6] B. Boguslawski, V. Gripon, F. Seguin, and F. Heitzmann, "Huffman coding for storing non-uniformly distributed messages in networks of neural cliques," in *AAAI 2014 : the 28th Conference on Artificial Intelligence*, vol. 1, 2014, pp. 262–268.

[7] E. B. Baum, J. Moody, and F. Wilczek, "Internal representations for associative memory," *Biological Cybernetics*, vol. 59, no. 4-5, pp. 217–228, 1988.

[8] R. Danilo, H. Jarollahi, V. Gripon, L. Conde-Canencia, P. Coussy, and W. J. Gross, "Algorithm and implementation of an associative memory for oriented edge detection using improved clustered neural networks," in *Circuits and Systems (ISCAS), 2015 IEEE International Symposium on*. IEEE, 2015, to appear.