Modèle et Architecture de Réseaux de Neurones Récurrents à Clones

HUGUES WOUAFO CYRILLE CHAVET PHILIPPE COUSSY

Lab-STICC, Université de Bretagne-Sud, Lorient
hugues.nono-wouafo@univ-ubs.fr, cyrille.chavet@univ-ubs.fr, philippe.coussy@univ-ubs.fr

Résumé - Les réseaux de neurones artificiels sont utilisés pour réaliser des tâches complexes dans de nombreux domaines tels que le traitement d'image ou du signal. Récemment, un nouveau modèle de réseau de neurones a été proposé pour concevoir des mémoires associatives et il est plus performant que les réseaux de Hopfield. Alors que sa performance est optimale face des messages uniformément distribués, celle-ci se dégrade amplement face à des distributions non-uniformes. Des solutions ont été proposées pour résoudre ce problème mais leurs architectures matérielles sont très coûteuses. Dans cet article, nous présentons une extension du modèle capable d'offrir de meilleures performances avec un coût matériel moindre par rapport aux solutions de l'état de l'art.

Abstract - Artificial neural networks are used for various complex tasks like signal and image processing. Recently, a new neural network has been proposed to design associative memories and outperforms Hopfield Neural Networks. While its performance is maximal when dealing with uniformly distributed messages, it highly degrades when considering non-uniformly distributed messages. Some solutions have been proposed to solve that problem but those are costly in terms of hardware resources. In this paper, we propose an extension of the original model, more performant with a lower cost compared to the solutions of the state of the art.

1 Introduction

Les réseaux de neurones biologiques sont des unités de traitement efficaces et peu énergivores qui dépassent largement les ordinateurs les plus puissants dans des tâches complexes. La volonté de reproduire cette capacité dans des systèmes a motivé la création de réseaux de neurones artificiels qui sont maintenant utilisés dans de nombreux domaines tel que la biologie [1], l'économie [2] ou de la vision par ordinateur [3]. Parmi les nombreux travaux existants, beaucoup s'intéressent à la capacité qu'ont les cerveaux des mammifères à apprendre des informations puis pour s'en rappeler à partir de quelques-uns de leurs fragments. Ces types de réseaux de neurones artificiels, dont les réseaux de Hopfield (HNN) [7] sont un représentant, sont utilisés pour concevoir des mémoires associatives [4]. Les mémoires associatives peuvent être utilisées pour réaliser des routeurs [5] ou faire du traitement d'image [6]. Récemment, un nouveau modèle de réseau de neurones nommé GBNN (pour Gripon Berrou Neural Network) [8] a été proposé. Ce modèle est plus performant que les HNN en termes de capacité c.à.d. de nombre maximal de messages pouvant être appris. Cependant, comme nous le verrons dans la section suivante, ses performances se dégradent pour des messages non uniformément distribués [9]. Pour pallier ce problème, des solutions ont été proposées mais malheureusement leurs architectures matérielles sont très coûteuses. Dans cet article, nous présentons une extension du modèle GBNN qui offre de meilleures performances et un coût matériel moindre par rapport aux solutions de l'état de l'art.

Cet article est organisé comme suit. Dans la deuxième section, les travaux existants autour du GBNN sont présentés. Dans la troisième section, le principe du modèle amélioré est introduit puis, son fonctionnement et son architecture matérielle sont détaillés. Finalement, la dernière section compare les performances et le coût des architectures matérielles de ce modèle par rapport aux solutions existantes.

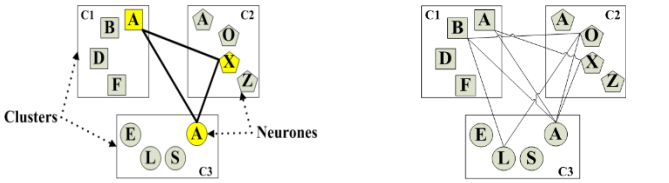
2 GBNN et réseaux de neurones totalement binaires

Un GBNN [8][11] est un réseau de neurones partiellement récurrent, à poids binaires, et où l'ensemble de neurones est divisé en un ensemble arbitraire de parties appelées *clusters*. Dans un GBNN, chaque neurone d'un cluster peut uniquement être connecté à des neurones contenus dans d'autres clusters. Si deux neurones sont connectés, leur poids est mis à '1'. Ainsi, avant l'apprentissage, tous les poids sont initialisés à '0'. On note $n_{i,j}$ un neurone i dans un cluster j et $v(n_{i,j})(t)$ l'état d'un neurone $n_{i,j}$ à un instant t. Quand $v(n_{i,j})(t) = '1'$, le neurone $n_{i,j}$ est dit *actif* sinon il est dit *inactif*. On note $w(n_{i,j}; n_{i',j'})$ le poids entre deux neurones $n_{i,j}$ et $n_{i',j'}$.

2.1 Apprentissage et décodage dans un GBNN

Soit E_j un ensemble fini de valeurs tel que $|E_j| = L_j$. Soit $P = (S_1, \dots S_j \dots, S_C)$ un message de C symboles tels que $S_j \in E_j, \forall\, j \in \{1..C\}$. Alors P peut être appris par un GBNN avec C clusters où chaque cluster j contient L_j neurones c.à.d. en associant un cluster par symbole et un neurone pour chaque valeur possible du symbole. Un GBNN avec C clusters ayant chacun L neurones, est noté GBNN(L,C). Son nombre total de poids est égal à $|W| = L * L * C * (C-1)/2$. L'apprentissage consiste à mettre à '1' les poids entre les neurones actifs suivant la

loi d'Hebb [10]. La Figure 1 montre un GBNN capable d'apprendre des messages composés de 3 lettres de l'alphabet latin c.à.d. un GBNN(26,3) ayant 3 clusters (1 cluster par caractère dans le message) avec 26 neurones chacun (un neurone par lettre possible).



a. Apprentissage de 'AXA' b. 'BOA' et 'BOL' ont été appris

Figure 1 Aperçu d'un GBNN(26,3)

Un symbole $S_j \notin E_j,\ j \in \{1 \dots C\}$ est dit *inconnu* car sa valeur ne fait pas partie de l'ensemble des valeurs possibles. Dans ce cas, aucun neurone du cluster auquel il est associé ne peut être initialement activé. Un GBNN peut à partir d'un message contenant des symboles inconnus retrouver le message original le plus proche qui a été appris. Ainsi, quand un GBNN reçoit un message P, chaque symbole connu active un neurone. Ensuite, les symboles inconnus sont cherchés lors du décodage. Pendant cette étape, l'état de chaque neurone dans les clusters effacés est calculé et plusieurs itérations peuvent être nécessaires pour retrouver le message le plus proche au sens de la distance de Hamming. Une itération correspond au calcul de l'état de chaque neurone concerné. L'état d'un neurone $n_{i,j}$ est calculé par l'équation suivante [11]:

$$v(n_{i,j})(t+1) = \bigwedge_{j'=1, j' \neq j}^{C} \left(\left(\bigvee_{i=1}^{L} \left(v(n_{i',j'})(t) \wedge w(n_{i,j}; n_{i',j'}) \right) \right) \vee \left(\bigvee_{i'=1}^{L} v(n_{i',j'})(t) \right) \right) \quad (1)$$

2.2 Architecture matérielle

L'architecture matérielle d'un GBNN(L,C) se compose typiquement d'un ensemble de C modules (les clusters) intégrant chacun deux sous-modules : le sous-module d'apprentissage contenant les poids associés aux L neurones de ce cluster et le sous-module de décodage réalisant le calcul des états des L neurones de ce cluster (Figure 2). Plusieurs mises en œuvre du GBNN ont été proposées dans la littérature. Une première architecture totalement parallèle du GBNN [8] est présentée dans [12]. Etant coûteuse, plusieurs améliorations du coût matériel ont été proposées dans [11]: transformation du modèle arithmétique en modèle purement logique, partage de poids entre neurones, sérialisation des calculs et des communications inter-clusters.

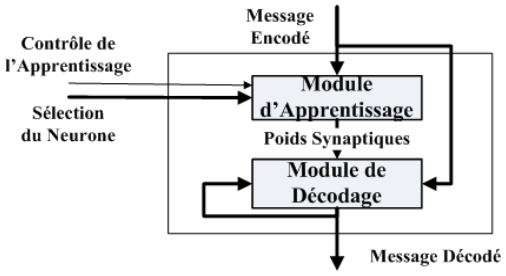


Figure 2 Aperçu de l'architecture d'un cluster

2.3 Performances

La performance du décodage d'un GBNN est évaluée par son Taux d'Erreur de Décodage (TED) qui est donné par le nombre de décodages réussis divisé par le nombre total de tentatives. Le TED est maximal quand les probabilités d'apparition des symboles dans les messages sont égales c.à.d. que leur distribution est uniforme [9]. Malheureusement, ce n'est généralement pas le cas et les performances s'en trouvent grandement affectées [14]. Pour pallier ce problème, plusieurs solutions ont été proposées dans [9]. Une première approche dite *« bits additionnels »* consiste à ajouter des informations à chaque symbole. Deux méthodes de génération de la séquence des bits additionnels ont été proposées : la méthode dite « Aléatoire » (ALE) qui consiste à générer ces bits de manière aléatoire et la méthode dite « Moins Récemment Utilisé » (MRU) qui consiste à utiliser la séquence la moins récemment utilisée. La seconde solution dite *« encodage d'Huffman »* (HUF) consiste à réaliser un encodage d'Huffman [13] de l'ensemble des messages à apprendre. Les codes sont ainsi appris à la place des messages dans le GBNN. Le codage étant statique, l'apprentissage est réalisé hors-ligne et aucune nouvelle information ne peut être apprise après cette opération.

Malheureusement, bien que ces deux approches améliorent grandement les performances lorsque les messages ne sont pas uniformément distribués (la solution par encodage d'Huffman étant la meilleure), leur coût matériel est très élevé. En effet, les deux solutions nécessitent d'utiliser des réseaux plus larges pour apprendre les messages. Ainsi, plutôt que d'utiliser un GBNN(256,8) pour mémoriser des messages de 8 symboles pouvant avoir 256 valeurs chacun, le réseau requis est un GBNN(1024,8) ce qui équivaut à 16 GBNN(256,8) en terme de points mémoire. Face à un coût aussi élevé, une nouvelle approche est donc nécessaire.

3 Réseau de neurones à clones

Afin de répondre efficacement au problème, nous proposons de *cloner* les neurones d'un GBNN. Les raisons motivants ce choix sont les suivantes. Les performances du GBNN [8][11] chutent lorsque les messages ne sont pas uniformément distribués c.à.d. lorsque certains neurones sont « trop utilisés ». En effet, l'apprentissage d'un grand nombre de messages comportant un symbole de même valeur se traduit par un grand nombre de poids synaptiques mis à '1' (grande densité) pour le neurone concerné. Ainsi, l'utilisation de plusieurs instances de neurones permet de répartir les poids synaptiques mis à '1' parmi les clones.

Formellement, un réseau de neurones à clones, nommé *Clone-Based Neural Network* (CBNN) associe plusieurs *clones* à chaque neurone. Un *clone* est une instance d'un neurone associée à un sous-réseau (Voir Figure 3). Un CBNN est alors un réseau de neurones possédant un nombre arbitraire de sous-réseaux compétitifs capable d'apprendre et restituer les messages appris. Chacun de ses sous-réseaux contient exactement un clone par neurone.

Un CBNN avec X sous-réseaux, C clusters dans chaque sous-réseau et L clones par cluster est noté CBNN(L,C,X) et a une mémoire égale à : $|W| = (X * L * L * C * (C-1))/2$. Un CBNN(L,C,X) possède alors une taille mémoire équivalente à X * GBNN(L,C).

Soit Gc un CBNN et $k \in Gc$ un sous-réseau. On note $n^x_{(i,j,k)}$ le clone x d'un neurone $n_{(i,j)}$ contenu dans un cluster j du sous-réseau k. On note $v(n^x_{(i,j,k)})(t)$ son état lors d'une itération de décodage t. Soit $D_k(j')$ l'ensemble des clones dans un cluster $j' \neq j$ de k. Si $d \in D_k(j')$ alors on note $w(n^x_{(i,j,k)}; d)$ le poids synaptique entre $n^x_{(i,j,k)}$ et d

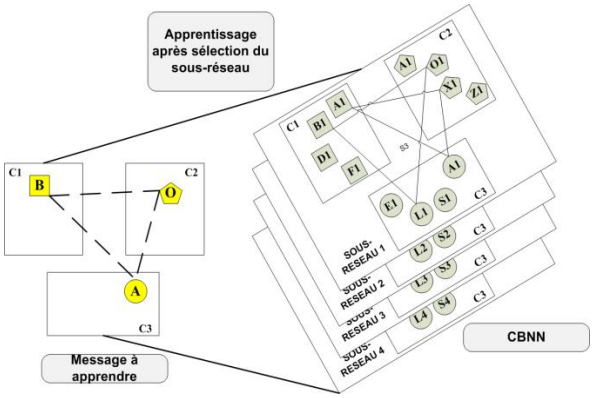


Figure. 3 Aperçu d'un CBNN(26,3,4).

3.1 Apprentissage et décodage dans un CBNN

Dans un CBNN, l'apprentissage se fait en deux étapes : (1) activation de tous les clones associés aux neurones déclenchés par le message et (2) sélection du sous-réseau qui apprendra le message.

Plusieurs stratégies peuvent être utilisées pour choisir le sous-réseau apprenant. Parmi ces stratégies, deux ont été évaluées: « Moins Récemment Utilisé » (MRU) et « Moins Dense » *(*MD). La stratégie MRU consiste à sélectionner le réseau le moins récemment utilisé. La stratégie MD est une heuristique mettant en compétition les clones. Cette compétition est mise en œuvre en calculant le score de chaque sous-réseau et en choisissant celui avec le meilleur score c.à.d. le score le plus faible dans notre cas. Considérons un message P contenant C symboles à apprendre. Soit Gc un CBNN(L,C,X). Quand Gc reçoit le message P, les clones relatifs aux neurones associés à la valeur des symboles présents dans le message fourni vont s'activer. Etant donné que chaque cluster dans Gc contient un clone par neurone alors seul un clone sera actif par cluster dans chaque sous-réseau. Soit $n^x_{(*,j,k)}$ le clone contenu dans le cluster j du sous-réseau k dans Gc est activé par P. Alors le score de k dans la compétition est définie par (2):

$$sc(k) = \sum_{j=1}^{C} \left(\sum_{n^{x'}_{(i',j',k)} \in D_k(j')} \left(v(n^{x'}_{(i',j',k)}) \oplus w(n^x_{(*,j,k)}; n^{x'}_{(i',j',k)}) \right) \right) \quad (2)$$

Le but de cette compétition est de sélectionner le sous-réseau possédant la densité la plus faible tout en prenant en compte qu'il est capable de réutiliser les poids déjà actifs (via l'opération logique XOR).

Le décodage est itératif et se fait en deux étapes : (1) activation de tous les clones associés aux neurones déclenchés par le message, et (2) décodage par les clones situés dans les clusters effacés. Un neurone est actif dans un sous-réseau si au moins un de ses clones est actif dans ce sous-réseau. Un CBNN retrouve le message si à la fin du décodage, il possède un unique sous-réseau avec un neurone actif par cluster. A une itération t, le prochain état d'un clone $n^x_{(i,j,k)}$ est donné par une formule identique à (1):

$$v(n^x_{(i,j,k)})(t+1) = \bigwedge_{j'=1, j' \neq j}^{C-1} \left(\left(\bigvee_{d \in D_k(j')} (v(d)(t) \wedge w(n^x_{(i,j,k)}; d)) \right) \vee \left(\overline{\bigvee_{d \in D_k(j')} v(d)(t)} \right) \right) \quad (3)$$

3.2 Architecture matérielle

L'architecture matérielle d'un CBNN(L,C,X) (Figure 4) se décompose en trois parties :

- X blocs *sous-réseaux* intégrant chacun un module de score (lorsque l'approche MD est utilisée) et un ensemble de C modules (les clusters) ayant chacun deux sous-modules : le sous-module d'apprentissage contenant les poids associés aux L clones de ce cluster et le sous-module de décodage réalisant le calcul des états des L clones de ce cluster ;
- Un bloc *contrôleur d'apprentissage* qui reçoit le score de chaque sous-réseau et sélectionne le gagnant. Le signal *learn* est mis à '1' durant la phase d'apprentissage du CBNN. L'apprentissage effectif est réalisé en modifiant le signal *slearn* qui contrôle le sous-réseau sélectionné ;
- Un bloc *contrôleur de sortie* qui sélectionne la sortie issue du sous-réseau ayant réalisé un décodage réussi.

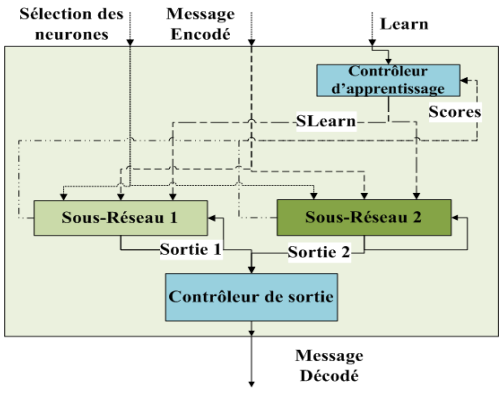


Figure 4 Aperçu d'un CBNN(26,3,2) et de son architecture

4 Résultats expérimentaux

Afin de valider le modèle proposé, deux ensembles d'expériences ont été réalisés. Le premier étudie le taux d'erreur de décodage de notre modèle face aux performances des techniques proposées dans [9] et présentées dans la section 2.3 (Figure 5). Le deuxième compare les coûts en ressources (surface) via la conception matérielle des différentes architectures (Figure 6). Les coûts sont normalisés par rapport au coût d'un GBNN(256,8). Plusieurs réseaux sont considérés pour ces études, tous ayant le même nombre de clusters et le même nombre de neurones par cluster :

(1) Deux GBNN(256,8) : l'un apprenant des messages basés sur une distribution uniforme et l'autre

apprenant des messages basés sur une distribution gaussienne.

(2) Trois GBNN(1024,8), chacun mettant en œuvre les stratégies proposées dans [9] (GBNN_ALT, GBNN_MRU et GBNN_HUF)

(3) 4 versions de notre modèle : un CBNN_MRU(256,8,8), un CBNN_MD(256,8,8), un CBNN_MRU(256,8,16), un CBNN_MD(256,8,16). Les CBNN_MRU(256,8,8) et CBNN_MD(256,8,8) possèdent une taille mémoire équivalente à 50% des GBNNs utilisant les stratégies de [9] tandis que les CBNN_MRU(256,8,16) et CBNN_MD(256,8,16) possèdent une taille mémoire égale à celle des GBNNs utilisant les stratégies de [9].

La Figure 5 montre le TED de l'ensemble de ces réseaux. L'ensemble des messages à apprendre est basé sur une distribution gaussienne (moyenne = 125, variance = 25) tel que proposé dans [9]. Le TED est déterminé par le décodage de 1000 messages aléatoirement sélectionnés et aléatoirement effacés à 50% comme proposé dans [9]. L'évaluation des performances a été réalisée sous MATLAB 2014. Les résultats montrent que notre CBNN_MRU(256,8,8) surpassent toutes les propositions de [9] en terme de TED mis à part la solution GBNN_HUF avec une amélioration en performance capable d'attendre 50% (+10% pour le CBNN_MD(256,8,8)). D'un autre côté le CBNN_MD(256,8,16) rivalise avec le GBNN_HUF (-5% dans le pire cas). Il surpasse les autres propositions (95%) avec un coût matériel identique à ces dernières. La Figure 6 montre aussi que la différence du coût en ressources entre le CBNN_MRU et le CBNN_MD est négligeable (1,8%) alors que la stratégie MD est plus efficace (10%).

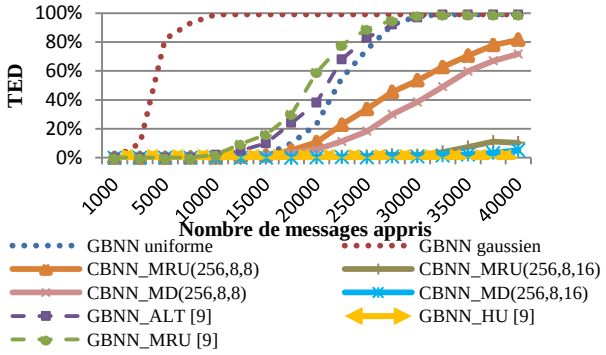


Figure 5 TED des différents réseaux évalués

La Figure 6 illustre l'évaluation matérielle. Comme indiqué dans [11], la surface des architectures est dominée par la mémoire qui peut représenter plus de 95% du coût total. Ainsi la mémoire des réseaux de neurones à un fort impact dans les résultats. Ainsi, afin de comparer les architectures de toutes les approches, l'évaluation matérielle a été réalisée en portes NAND équivalentes (NAND Equivalent Gates : NEG) basé sur des libraires de composants synthétisés (Technologie 90nm de STMicroelectronics). La Figure 6 montre que les CBNN(256,8,8) (MRU et MD) que nous proposons réduisent de 50% le coût comparées aux solutions présentées dans [9].

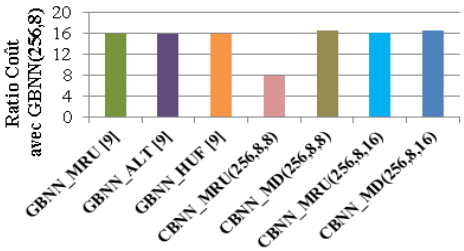


Figure 6 Evaluation du coût en ressources

5 Conclusion

Dans cet article, une évolution du modèle de réseau de neurones GBNN est proposée afin d'en améliorer les performances lorsque une distribution non-uniforme des messages à apprendre est considérée. Le modèle que nous proposons est basé sur le concept de clone et est nommé Clone-Based Neural Network (CBNN). Les résultats montrent que ce modèle offre des performances meilleures ou comparables aux méthodes existantes et permet une optimisation significative du coût des architectures matérielles. Bien que ce modèle soit intéressant, les nombreuses possibilités fournies par le concept de clones restent à explorer. Ainsi, nos travaux futurs concerneront la conception de CBNNs admettant plus d'un clone par sous-réseau ou encore des CBNNs dynamiques qui ne créent des clones uniquement qu'en cas de besoin.

6 Références

[1] M. Saglam, Y. Hayashida, and N. Murayama, "A retinal circuit model accounting for wide-field amacrine cells," *Cogn. Neurodyn.*, vol. 3, no. 1, pp. 25–32, Mar. 2009.

[2] A. A. Godarzi, R. M. Amiri, A. Talaei, and T. Jamasb, "Predicting oil price movements: A dynamic Artificial Neural Network approach," *Energy Policy*, vol. 68, pp. 371–382, May 2014.

[3] N. Hoft, H. Schulz, and S. Behnke, "Fast Semantic Segmentation of RGB-D Scenes with GPU-Accelerated Deep Neural Networks," *hgpu.org*, Aug. 2014.

[4] G. E. Hinton and J. A. Anderson, *Parallel Models of Associative Memory: Updated Edition*. Psychology Press, 2014.

[5] J. G. Delgado-Frias, J. Nyathi, and L. Bhuyan, "A Wave-pipelined Router Architecture Using Ternary Associative Memory," in *Proceedings of the 10th Great Lakes Symposium on VLSI*, New York, NY, USA, 2000, pp. 67–70.

[6] E. Guzmán, O. M. C. Jiménez, A. D. Pérez, and O. Pogrebnyak, "Using Associative Memories for Image Segmentation," in *Advances in Neural Networks – ISNN 2011*, D. Liu, H. Zhang, M. Polycarpou, C. Alippi, and H. He, Eds. Springer Berlin Heidelberg, 2011

[7] J. J. Hopfield, "Neural networks and physical systems with emergent collective computational abilities.," *Proc. Natl. Acad. Sci. U. S. A.*, vol. 79, no. 8, pp. 2554–2558, Apr. 1982.

[8] V. Gripon and C. Berrou, "Sparse neural networks with large learning diversity," *IEEE Trans. Neural Netw*, vol.22, no.7, 2011.

[9] B. Boguslawski, V. Gripon, F. Seguin, F. Heitzmann, "Huffman Coding for Storing Non-uniformly Distributed Messages in Networks of Neural Cliques," in *AAAI 2014 : the 28th Conference on Artificial Intelligence*, Québec, Canada, 2014, vol. 1, pp. 262–268.

[10] R. G. Morris, "D.O. Hebb: The Organization of Behavior, Wiley: New York; 1949," *Brain Res. Bull.*, vol. 50, no. 5–6, p. 437, Dec. 1999.

[11] P. Coussy, C. Chavet, L. C. Canencia, and H. N. Wouafo, "Fully-Binary Neural Network Model and Optimized Hardware Architectures for Associative Memories," *ACM J. Emerg. Technol. Comput. Syst.*, To appear.

[12] H. Jarollahi, N. Onizawa, V. Gripon, W. J. Gross, "Architecture and implementation of an associative memory using sparse clustered networks," in *2012 IEEE International Symposium on Circuits and Systems (ISCAS)*, 2012, pp. 2901–2904.

[13] D. Huffman, "A Method for the Construction of Minimum-Redundancy Codes" *Proc. IRE*, vol.40, no.9, pp.1098–1101, Sep. 1952.

[14] 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.