U.R.S.I. FRANCE JS 26

OUTILS ET MÉTHODES INTELLIGENTES POUR LES RADIO-SCIENCES

SOM embarquée pour localisation de source: pré-entraînement, affinage et implémentation FPGA

Embedded SOM for source localization: pre-training, fine-tuning and FPGA implementation

Loïc Thomas[1], Gaël Loubet[2], Gabriela Nicolescu[3], Daniela Dragomirescu[4]

[1]*LAAS-CNRS, Université de Toulouse, CNRS, loic.thomas@laas.fr*
[2]*LAAS-CNRS, Université de Toulouse, CNRS, gael.loubet@laas.fr*
[3]*École Polytechnique de Montréal, gabriela.nicolescu@polymtl.ca*
[4]*LAAS-CNRS, Université de Toulouse, CNRS, daniela.dragomirescu@laas.fr*

Mots clés : Implémentation FPGA, IA en périphérie, Apprentissage non supervisé, FPGA implementation, Edge-AI, Unsupervised learning

---

Résumé/Abstract

This work proposes a pipeline for real-time, online source localization using a logic-only FPGA implementation of a Self-Organizing Map (SOM) with GCC-PHAT features. The SOM is trained offline on a noise-free theoretical dataset and fine-tuned on real recordings from the SSLR dataset captured by four microphones on a Pepper robot. A simplified learning rule enables deployment on low-resource FPGA devices. While validated on acoustic signals, this method exploits generic inter-sensor time delays from cross-correlation features, supporting extensions to RF localization and Intrusion Detection Systems (IDS).

Ce travail propose un système permettant la localisation de sources en temps réel et en ligne, reposant sur une implémentation FPGA (en logique seule) d'une Self-Organizing Map (SOM) utilisant des GCC-PHAT de signaux comme entrées. La SOM est entraînée hors ligne sur un jeu synthétique sans bruit, puis affinée sur des enregistrements réels du jeu de données SSLR acquis avec quatre microphones sur la tête du robot Pepper. Une règle d'apprentissage simplifiée permet un déploiement sur des petits FPGA. Bien que validée sur des signaux acoustiques, la méthode exploite des délais inter-capteurs issus de la corrélation croisée, ouvrant la voie à la localisation RF et à des systèmes de détection d'intrusion (IDS).

1 Introduction

Une part importante des traitements d'IA est aujourd'hui exécutée sur des serveurs distants, ce qui implique la transmission de données depuis les capteurs vers le cloud. Cette organisation peut dégrader la latence et augmenter les coûts énergétiques et de communication, tout en posant des problèmes de confidentialité et de sécurisation des données. Dans ce contexte, l'intelligence artificielle en périphérie du réseau (Edge AI) vise à rapprocher l'inférence des capteurs afin d'obtenir un traitement plus réactif, moins coûteux en énergie et mieux maîtrisé [1], [2]. L'entraînement de modèles d'IA cherche à obtenir une solution capable de généraliser à des données inédites, ce qui conduit souvent à utiliser des modèles plus profonds et ainsi, plus coûteux en calcul et en mémoire. Or, en Edge-AI, l'inférence au plus près des capteurs impose des contraintes strictes de mémoire, d'énergie et de latence, rendant l'emploi de modèles volumineux plus difficile [3]. Ainsi, pour des tâches embarquées et bien définies, un modèle spécialisé et adaptable peut être plus pertinent qu'un modèle universel figé [4].
La Self-Organizing Map (SOM) est un algorithme non supervisé de quantification de vecteurs, couramment utilisé pour le clustering, la compression et la visualisation [5]. Son emploi pour la localisation de source sonore reste toutefois moins fréquent [6]. Cette tâche est centrale pour l'audition robotique et l'interaction humain–robot. Les fortes variations environnementales (réverbération, configuration des pièces) rendent difficile l'obtention d'une

solution générique et embarquable. L'approche proposée exploite donc le caractère non supervisé de la SOM pour permettre une adaptation in situ, via un pré-entraînement synthétique puis un affinage sur une version simplifiée de la SOM et implémentable facilement sur un petit FPGA. La méthode proposée n'est pas limitée à l'acoustique, la localisation sonore et la localisation RF peuvent toutes deux être formulées comme l'estimation de délais de propagation inter-capteurs à partir des signaux reçus, en s'appuyant sur la corrélation croisée ou des caractéristiques apparentées [7]. Ainsi, la chaîne de pré-entraînement synthétique et d'affinage in situ constitue une première étape vers la localisation de sources RF, avec des applications directes en surveillance de sécurité et en systèmes de détection d'intrusion basés sur la RF (IDS).

2 Self-Organizing Map (SOM)

Une Self-Organizing Map (SOM) [8] est une grille de M nœuds, où chaque nœud porte un vecteur de poids de dimension N. À l'apprentissage, un vecteur d'entrée (dimension N) est comparé aux poids de tous les nœuds. Le nœud dont les poids sont les plus proches de l'entrée est le nœud gagnant (BMU). Les poids du BMU et de ses voisins sont ensuite ajustés pour ressembler à l'entrée. Après entraînement, une classification peut être obtenue par un étiquetage des nœuds a posteriori, en repassant le jeu d'entraînement sans mise à jour et en associant à chaque nœud la classe qu'il active le plus souvent.

3 Traitement des données

Le jeu de données SSLR regroupe des enregistrements multi-pièces réalisés avec les quatre microphones du robot Pepper [9], accompagnés des coordonnées de la source sonore. Afin de réduire la dimension des données et la sensibilité à la réverbération, les cross-corrélation généralisées (GCC-PHAT) sont extraits à partir des signaux. Le bruit causé par les ventilateurs de Pepper, situé à proximité des microphones, rend les GCC-PHAT moins discriminantes (pics élargis, ajout d'autres pics) [10], en particulier pour les sons venant de l'arrière. Pour compenser, pour chaque paire de microphones, les spectres croisés du bruit seul sont calculés (enregistrements déjà présents dans le jeu de données), puis ils sont soustraits à ceux obtenus sur les signaux provenant des sources à localiser. [11], [12]. Dans un cas d'utilisation réel, le bruit ambiant peut être capturé en laissant le robot seul afin qu'il puisse se calibrer sur ce principe

Par ailleurs, ce jeu de données a été mesuré à l'aide d'une version du robot Pepper où les micros étaient directifs et orientés vers l'avant. De plus, le jeu de données est plutôt déséquilibré sur la répartition des sons venant de l'avant ou de l'arrière 71% des sons ont des azimuts compris dans $[0, 90] \cup [270, 359]$ alors que 29% viennent de l'arrière.

4 Entrainement

L'entraînement de la SOM se fait en deux parties. Une première sur des données synthétiques et une deuxième embarquable sur les données réelles.

4.1 Pré-entraînement synthétique

Un pré-entraînement est réalisé avant l'embarquement afin d'obtenir des performances initiales acceptables pendant l'adaptation et d'accélérer l'affinage sur données réelles. Un jeu de données synthétiques est créé en calculant la GCC-PHAT de signaux non-bruités et déphasés manuellement afin de reproduire le décalage entre deux microphones (en utilisant la même géométrie que Pepper). L'entraînement synthétique est fait avec l'algorithme de la SOM sur plusieurs passes de ce jeu de données.

Sur la Figure 1, les labels associés aux 72 nœuds d'une SOM sont représentés. Ils ont été calculés grâce à la moyenne des azimuts pour lesquels un nœud i s'est activé (le nœud le plus proche de l'entrée). Grâce à la fonction de voisinage, la SOM représente les différentes données comme une liste contigüe d'angles d'arrivée. Ici, contrairement aux topologies classiques, la SOM est implémentée de manière circulaire, le nœud 71 est voisin avec le nœud 0. Sur une SOM en 2D, la représentation s'apparente plutôt à une spirale. Sur des données non altérées, l'algorithme de la SOM permet de bien reproduire la géométrie des nœuds.

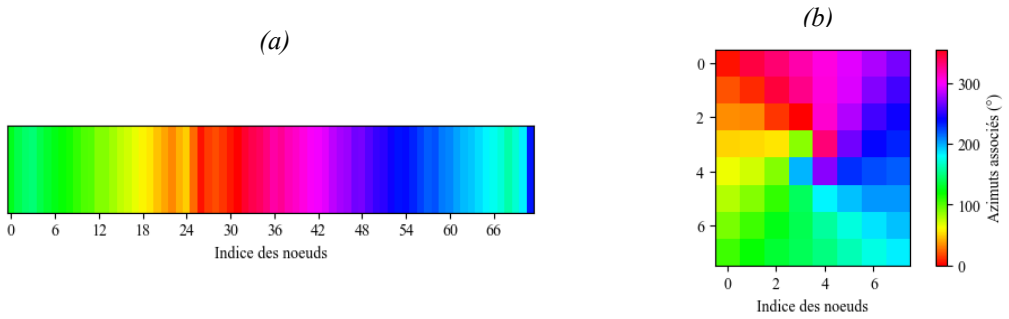

Figure 1 : Comparaison de l'étiquetage d'une SOM 1D sur 72 nœuds (a) et d'une SOM 2D 8×8 (b)

4.2 Affinage sur données réelles

L'affinage vise à simuler l'adaptation à des données réelles tout en rendant l'algorithme plus facilement implémentable sur un petit FPGA. La SOM est simplifiée en supprimant la mise à jour du voisinage : seul le nœud gagnant est mis à jour. Le taux d'apprentissage est maintenu constant, et la distance de Manhattan est utilisée pour comparer l'entrée aux poids. La Figure 2 montre l'impact de cet affinage sur des données réelles sur une SOM à 36 nœuds.

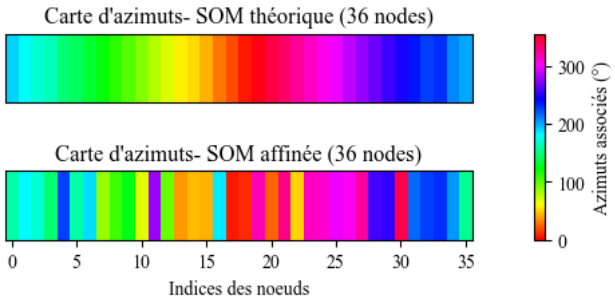

Figure 2 : Différence entre les étiquettes d'une SOM après le pré-entraînement (en haut) et affinée (en bas), les cases noires correspondent à des nœuds non étiquetés

4.3 Inférence

Sur des données réelles, l'étiquetage des données classique devient inadaptée : les classes passées ne sont pas disponibles et, même avec une labélisation issue du pré-entraînement synthétique, les nœuds dérivent lors de l'affinage. Le fait de modifier uniquement les poids du nœud gagnant, renforce cet effet en cassant la continuité de la SOM (Figure 2, 2ème ligne). L'inférence doit donc reposer directement sur les poids. Ceux-ci sont ainsi interprétés comme une quantification de GCC-PHATs associées à des directions. Lors d'une inférence, le vecteur d'entrée est masqué en le multipliant par les poids du nœud vainqueur (BMU). Une heuristique exploitant les pics des GCC-PHAT est ensuite appliquée afin d'estimer la direction d'arrivée. L'utilisation d'une SOM pour réaliser ce type de masquage a déjà été explorée, notamment pour contrôler les sorties de perceptrons au sein d'un perceptron multicouche (MLP) dans un contexte d'apprentissage continu [13].

4.4 Résultats

Le pipeline complet (pré-entrainement et affinage) a été simulé sur des SOM avec des topologies différentes et les résultats sont résumés dans le Tableau 1. Le pré-entraînement se fait sur 5 passes. Une fois entraînée la SOM est testée (sans modification de poids) sur les données réelles de test issues de SSLR. Pour la phase d'affinage, la SOM tourne sur la partie entraînement de SSLR (une passe soit environ 8 heures d'enregistrement). Ensuite, la SOM est encore une fois testée sur les mêmes données de test de SSLR. Afin d'évalué le modèle, deux métriques sont employées. Premièrement, la différence d'angle moyenne entre l'azimut de la source par rapport au robot et l'azimut proposé par le modèle. La deuxième métrique est la précision dans un intervalle donné (5°, 10° et 20°).

	Erreur moyenne (°)			Précision ± 5 °			Précision ± 10 °			Précision ± 20 °		
	Toutes dir	Avant	Arrière	Toutes dir	Avant	Arrière	Toutes dir	Avant	Arrière	Toutes dir	Avant	Arrière
SOM synthétique (36 nœuds)	15.7 °	10.1 °	30.6 °	68.9 %	74.2 %	55.3 %	83.5 %	89.6 %	67.6 %	87.4 %	93.1 %	72.6 %
SOM affinée (36 nœuds)	12.1 °	7.1 °	25.0 °	71.0 %	75.9 %	58.3 %	88.1 %	93.1 %	75.3 %	90.9 %	95.3 %	79.3 %
SOM synthétique (72 nœuds)	17.4 °	10.3 °	35.9 °	68.0 %	74.2 %	51.9 %	83.2 %	89.9 %	66.0 %	86.7 %	93.2 %	69.9 %
SOM affinée (72 nœuds)	11.2 °	6.4 °	23.6 °	77.2 %	81.7 %	65.4 %	89.8 %	94.1 %	78.6 %	91.9 %	95.9 %	81.4 %
SOM 2D synthétique (8 x 8 nœuds)	15.8 °	9.27 °	32.8 °	72.2 %	77.9 %	57.2 %	85.2 %	91.3 %	69.3 %	88.2 %	93.9 %	73.4 %
SOM 2D affinée (8 x 8 nœuds)	11.6 °	6.6 °	24.6 °	75.9 %	80.9 %	62.8 %	89.2 %	93.8 %	77.2 %	91.4 %	95.7 %	80.2 %

Tableau 1 : Performances des SOM (par colonne, les sons provenant de : toutes les directions/l'avant/l'arrière

Le Tableau 1 résume les résultats sur deux SOM 1D comptant 36 ou 72 nœuds et une SOM 2D comptant 8×8 nœuds. Les performances sont étudiées entre les poids issus de l'entraînement synthétique et de l'affinage. Sur les trois sous-colonnes, sont comparés les performances en utilisant pour le test soit : toutes les données du jeu, uniquement les sons provenant de l'avant ou uniquement les sons provenant de l'arrière. Néanmoins l'entraînement est fait sur l'entièreté des données. Les micros étant directifs vers l'avant, la détection des sons provenant de l'arrière est plus complexe. Le jeu de données synthétiques a été créé en prenant en compte le cas le plus général possible, avec des signaux simples, sans bruits et des micros omnidirectionnels. Ainsi, les résultats des SOM synthétiques sur les sons arrière sont moins bons que pour les sons venant de l'avant. Néanmoins, après affinage, la précision augmente plus pour les sons venant de l'arrière que pour les sons venant de l'avant. Par exemple pour la SOM à 72 nœuds à ± 5 ° l'affinage permet une amélioration d'environ 13 % pour l'arrière et 7 % pour l'avant. Cela démontre que le système proposé peut s'adapter à des données assez nouvelles. La SOM 2D converge vers une meilleure solution pendant l'entraînement synthétiques par rapport aux SOM 1D. Cependant, vu que l'algorithme d'affinage ne prend pas en compte le voisinage du BMU, le fait que la SOM soit 2D ou 1D n'a aucun impact pendant l'affinage. Ainsi, la SOM 2D (64 nœuds) affinée est plus précise que la SOM 1D à 36 nœuds et moins que celle avec 72 nœuds, ce qui serait attendu avec une SOM 1D de 64 nœuds.

5 Implémentation hardware

Les FPGA sont des bons candidats pour l'implémentation de SOM, en effet, la majeure partie des opérations (comparaison avec le vecteur d'entrée et modification des poids) peuvent se faire de manière complètement parallèle [5]. Un design a été créé en utilisant Vitis HLS. La cible est un petit FPGA (Basys 3 Artix-7). Deux implémentations ont été faite. Une première entièrement parallèle et une en série où la comparaison se fait sur un seul cœur de calcul. Avec 37 nœuds cette première version utilise 18 155 lookup tables (85% de l'Artix 7) avec une latence d'inférence de 131 µs. La version série n'utilise que 2612 LUTs (12%) avec une latence de 505 µs.

6 Conclusion

Un pipeline permettant l'entraînement, l'affinage et l'implémentation hardware sur FPGA d'une SOM a été réalisé. L'entraînement se fait sur des données synthétiques permettant de rendre la partie d'affinage plus simple et plus facilement implémentable en hardware en simplifiant le fonctionnement de la SOM. Ces étapes permettent d'avoir un système qui s'adapte à son environnement sans être dépendant de la cible (géométrie des micros, nombre de micros, etc.) tout en ayant une précision largement suffisante pour être utilisé dans le cadre d'un fonctionnement d'un robot interagissant avec ses interlocuteurs. Ce travail pourrait être étendu à d'autres domaines que la localisation de source sonore. En effet, cette approche d'affinage permettrait d'entraîner le modèle in-situ sans pour autant utiliser un grand jeu de données étiquetées. Par exemple il serait applicable à la localisation de source RF, notamment pour des systèmes de détections d'intrusion. L'utilisation de MLP ou d'autre type de réseau de neurones[14], [15] au lieu de l'heuristique pourrait aussi améliorer les résultats.

Références

[1] K. C. Barr et K. Asanović, « Energy-aware lossless data compression », *ACM Trans. Comput. Syst.*, vol. 24, nº 3, p. 250-291, août 2006, doi: 10.1145/1151690.1151692.

[2] R. Singh et S. S. Gill, « Edge AI: A survey », *Internet of Things and Cyber-Physical Systems*, vol. 3, p. 71-92, 2023, doi: 10.1016/j.iotcps.2023.02.004.

[3] X. Wang *et al.*, « Empowering Edge Intelligence: A Comprehensive Survey on On-Device AI Models », *ACM Comput. Surv.*, vol. 57, nº 9, p. 228:1-228:39, avr. 2025, doi: 10.1145/3724420.

[4] Y. Shi, X. Ying, et J. Yang, « Deep Unsupervised Domain Adaptation with Time Series Sensor Data: A Survey », *Sensors*, vol. 22, nº 15, p. 5507, janv. 2022, doi: 10.3390/s22155507.

[5] S. Jovanović et H. Hikawa, « A Survey of Hardware Self-Organizing Maps », *IEEE Transactions on Neural Networks and Learning Systems*, vol. 34, nº 11, p. 8154-8173, nov. 2023, doi: 10.1109/TNNLS.2022.3152690.

[6] D. Hwang et J. Choi, « Real-Time Binaural Sound Source Localization Using Sparse Coding and SOM », in *Intelligent Robotics and Applications*, H. Liu, H. Ding, Z. Xiong, et X. Zhu, Éd., Berlin, Heidelberg: Springer, 2010, p. 582-589. doi: 10.1007/978-3-642-16584-9_56.

[7] M. Kabiri, C. Cimarelli, H. Bavle, J. L. Sanchez-Lopez, et H. Voos, « A Review of Radio Frequency Based Localisation for Aerial and Ground Robots with 5G Future Perspectives », *Sensors*, vol. 23, nº 1, p. 188, janv. 2023, doi: 10.3390/s23010188.

[8] T. Kohonen, « The self-organizing map », *Proceedings of the IEEE*, vol. 78, nº 9, p. 1464-1480, sept. 1990, doi: 10.1109/5.58325.

[9] A. Politis, S. Adavanne, et T. Virtanen, « A Dataset of Reverberant Spatial Sound Scenes with Moving Sources for Sound Event Localization and Detection », 27 juin 2020, *arXiv*: arXiv:2006.01919. Consulté le: 19 novembre 2024. [En ligne]. Disponible sur: http://arxiv.org/abs/2006.01919

[10] J. Wang, X. Qian, Z. Pan, M. Zhang, et H. Li, « GCC-PHAT with Speech-oriented Attention for Robotic Sound Source Localization », in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, mai 2021, p. 5876-5883. doi: 10.1109/ICRA48506.2021.9561885.

[11] L. Netsch et J. Stachurski, « Robust low-resource sound localization in correlated noise », in *Interspeech 2014*, ISCA, sept. 2014, p. 2218-2222. doi: 10.21437/Interspeech.2014-506.

[12] F. Bin et X. Lei, « The Combination of Spectrum Subtraction and Cross-power Spectrum Phase Method for Time Delay Estimation », *Archives of Acoustics*, vol. 45, nº 3, p. 453-458, juill. 2020, doi: 10.24425/aoa.2020.134061.

[13] P. Bashivan, M. Schrimpf, R. Ajemian, I. Rish, M. Riemer, et Y. Tu, « Continual Learning with Self-Organizing Maps », 19 avril 2019, *arXiv*: arXiv:1904.09330. doi: 10.48550/arXiv.1904.09330.

[14] X. Liu, L. Mo, et M. Tang, « SOM-Associated-SNN: Enhancing audio classification with spiking neural networks through single-modality clustering and associative learning », *Neurocomputing*, vol. 640, p. 130416, août 2025, doi: 10.1016/j.neucom.2025.130416.

[15] M. L. Baptista, E. M. P. Henriques, et K. Goebel, « A self-organizing map and a normalizing multi-layer perceptron approach to baselining in prognostics under dynamic regimes », *Neurocomputing*, vol. 456, p. 268-287, oct. 2021, doi: 10.1016/j.neucom.2021.05.031.