UNIVERSITÉ DE SHERBROOKE
Faculté de génie
Département de génie électrique et de génie informatique

Traitement et compression de données en temps réel en utilisant l'intelligence artificielle pour des détecteurs à haut débit

Mémoire de maitrise
Spécialité : génie électrique

Berthié Gouin-Ferland

Sherbrooke (Québec) Canada

Avril 2023

MEMBRES DU JURY

Audrey Corbeil Therrien
Directrice

François Grondin
Évaluateur

Jean-Baptiste Michaud
Évaluateur

RÉSUMÉ

Le *SLAC National Accelerator Laboratory* démarrera bientôt la prochaine génération de lasers à électrons libres rayons X ; le *Linac Coherent Light Source - II* (LCLS-II). Les expériences scientifiques rendues possibles par ce dernier conduisent à une génération de données qui dépassera le To/s. L'objectif global de la recherche proposée est de réduire la bande passante requise pour le transfert des données générées par les expériences qui auront lieu dans ces nouvelles installations. Pour ce faire, nous manipulons localement et en temps réel les données numériques rendues disponibles par un détecteur du *SLAC National Accelerator Laboratory* (la «CookieBox»), afin de compresser les données par discrimination événementielle. Cette compression rejette les données générées avec des impulsions de rayons X non conformes, ce qui diminue la bande passante globale. Les algorithmes existants ne permettent pas d'atteindre les requis en latence pour le traitement des données. En revanche, des algorithmes d'intelligence artificielle, comme le réseau de neurones artificiels, utilisent des opérations arithmétiques simples et parallélisables. Par conséquent, ces algorithmes démontrent le potentiel d'atteindre les requis en latence pour traiter les données générées par la «CookieBox».

Malgré tout, la conception et l'utilisation d'algorithmes d'intelligence artificielle peuvent être incompatibles avec l'acquisition de données scientifiques en temps réel. Ainsi, ces travaux combinent des méthodes standards de compression de données avec les réseaux neuronaux. Cette approche permet d'atteindre la latence nécessaire pour la compression, en plus de minimiser l'empreinte matérielle nécessaire. Dans un premier temps, une méthode de quantification nonuniforme optimise la représentation de l'information à travers les données de la «CookieBox». La représentation plus dense permet de compresser les données en conservant les informations pertinentes qu'elles contiennent. Cela signifie que la taille des données est réduite sans perdre les informations importantes. Par la suite, un petit réseau neuronal utilise ces données afin de classifier les impulsions de rayons X en moins de 6 µs avec un taux de classification de 86 %. Il est possible d'atteindre une latence de moins de 2 µs en tolérant une baisse de 3 % sur le taux de classification.

Les travaux de recherche proposés offrent une solution pour réduire la génération de données massives lors des expériences scientifiques de grande envergure tout en considérant les paradigmes liés à l'intelligence artificielle en science. La communauté scientifique pourra acquérir plus de données de meilleure qualité et les analyser plus rapidement, ce qui accélérera la recherche. Ces travaux auront des retombées directes sur la performance du détecteur visé, qui contribuera à la recherche fondamentale en structure des matériaux et des molécules biologiques. Plus encore, les outils de compression développés pourront être transférés à d'autres applications, par exemple en imagerie médicale et en informatique quantique.

Mots-clés : Instrumentation, Détecteur Rayons X, Système d'acquisition temps réel, Apprentissage machine, Données massives, Détecteur à débit ultra rapide, Informatique de périphérie, Système embarqué

Merci aux personnes qui m'entourent de transformer les probabilités en opportunités

TABLE DES MATIÈRES

LISTE DES FIGURES

LISTE DES TABLEAUX

CHAPITRE 1

INTRODUCTION

1.1 Mise en contexte

La communauté scientifique connaît une grande transformation quant à l'utilisation des données massives. Dans des disciplines comme la physique, la chimie et la médecine, les détecteurs pour expériences scientifiques accumulent les données à un rythme qui s'accroît de manière exponentielle. Cependant, la bande passante des systèmes d'acquisition, les capacités de stockage numérique en plus des délais et coûts associés à cette infrastructure imposent des limites importantes aux systèmes numériques modernes. Ainsi, la communauté scientifique manifeste un grand besoin en compression de données en temps réel pour détecteurs haut débit afin d'exploiter les nouveaux instruments de recherche scientifique à leur plein potentiel.

Le premier laser à électrons libres rayons X du *SLAC National Accelerator Laboratory* ; le *Linac Coherent Light Source* (LCLS), a permis des avancées dans plusieurs domaines de la science. Par exemple, le LCLS a servi à résoudre des structures macromoléculaires protéiques complexes auparavant inaccessibles, il a capturé la formation de liaisons de réactions chimiques en état de transition et il a permis bien d'autres observations dans des disciplines comme la physique, la chimie et la médecine [1]. À l'instar du LCLS, le LCLS-II représente une grande avancée dans la technologie des lasers à électron libre rayons X et est considéré comme un outil important pour la science des énergies [1]. Il est notamment prévu que le taux d'impulsions augmente de 120 répétitions par seconde pour le LCLS à 1 million de répétitions par seconde pour le LCLS-II [2]. Cette augmentation majeure du taux de répétition, combinées à des détecteurs plus grands et plus performants, conduisent à une génération de données incroyablement élevée dépassant le To/s [1]. Cette situation entraîne une augmentation significative des besoins en infrastructure pour le stockage et le transfert de données, tout en imposant des contraintes plus strictes en termes de latence et de puissance pour le traitement et le transfert de ces données. Cela entraîne aussi des coûts considérables en termes d'investissements nécessaires pour augmenter l'infrastructure de stockage et de transfert de données, ainsi que la capacité de traitement et de bande passante, afin d'assurer le bon fonctionnement des systèmes informatiques. De plus, les

1. 10^{12} octets par seconde

contraintes plus strictes en termes de latence et de puissance ajoutent encore au défi à relever, car il faut trouver des solutions technologiques avancées capables de répondre à ces exigences croissantes.

Lorsque des impulsions de rayons X sont générées et envoyées sur une cible pour récolter des données à l'aide de détecteurs, il est crucial de s'assurer que ces impulsions sont conformes. Si une impulsion non conforme frappe la cible, les données expérimentales peuvent être compromises, ce qui rend difficile la distinction entre les caractéristiques de la cible et les effets de l'impulsion inappropriée. Ceci dit, le processus générant les impulsions de rayons X a une composante stochastique importante. Cela implique que les impulsions produites ne sont pas toujours conformes au cadre expérimental requis par les expériences en cours. Les données expérimentales générées par ces évènements non conformes sont donc compromises. On peut tirer avantage de ce phénomène pour réduire la génération de données au LCSL-II à l'aide d'une compression par discrimination événementielle. La méthode consiste à rejeter les données générées par les évènements jugés non conformes pour ne conserver que les données générées par les évènements jugés conformes. Ainsi, la compression par discrimination événementielle permet de diminuer le débit de données tout en conservant l'information utile relative aux expériences en cours. Il s'avère donc primordial que la méthode de discrimination soit fiable.

Pour ce faire, le *SLAC National Accelerator Laboratory* utilise le détecteur à balayage angulaire « CookieBox ». La « CookieBox » échantillonne les impulsions de rayons X avec une résolution temporelle en attoseconde [2] [3]. Ceci permet de caractériser une impulsion de rayons X en temps et en énergie. Cette caractérisation est faite avant que l'impulsion atteigne le site d'interaction relatif aux expériences principales en cours. Le résultat de cette caractérisation est le facteur déterminant pour faire la discrimination événementielle. La « CookieBox » sert alors de détecteur à diagnostics pour la méthode de compression par discrimination événementielle. Pour ce faire, la « CookieBox » appose un veto aux impulsions de rayons X qui ne respectent pas le cadre expérimental. Toujours est-il, le veto doit être disponible aux détecteurs rapidement. Rappelons qu'il est prévu que le LCLS-II opère à des taux d'impulsions d'un million de répétitions par seconde causant un débit de données dépassant le To/s. Les systèmes d'acquisition de données des détecteurs sont contraints de manière significative en termes de transfert et de sauvegarde temporaire de données en raison de limitations techniques et logistiques importantes imposées par un tel taux de répétition. Afin de caractériser l'impulsion et tenir compte du taux de répétition du LCLS-II d'un million de répétitions par seconde, il est nécessaire d'analyser les données

2. 10^{-18} s

numériques fournies par la « CookieBox » dans l'ordre de la microseconde pour pouvoir rapidement apposer le veto.

Les méthodes d'analyses existantes nécessitent plusieurs calculs itératifs à grande complexité algorithmique exigeant l'utilisation d'unités centrales de traitement (*CPU*) [3] et d'unités de traitement graphique (*GPU*) [4] [4]. Ces analyses sont faites hors ligne sur des nœuds de calcul. Les méthodes d'analyse actuelles ne sont pas adéquates en termes de latence et de ressources matérielles, car le traitement hors ligne avec des CPU et GPU nécessite un transfert de données trop rapide et des installations matérielles trop importantes. En revanche, des techniques récentes d'intelligence artificielle (IA) comme les réseaux neuronaux artificiels (RNA) répondent bien aux contraintes présentées. Les RNA se démarquent par leurs capacités à modéliser des problèmes hautement non linéaires à l'aide d'opérations arithmétiques simples et parallélisables.

D'autre part, de nouveaux outils logiciels permettent maintenant leur déploiement sur des *FPGA* [5]. Contrairement aux *CPU* et *GPU*, le *FPGA* peut être optimisé afin de rencontrer les requis nécessaires en termes de latence et de ressources matérielles. L'utilisation d'un RNA embarqué sur une plateforme de calcul physiquement proche des détecteurs est appelée apprentissage machine de périphérie (AMP) [6]. Plusieurs aspects sont à considérer lors du développement d'un RNA et encore plus quand ce RNA doit faire des prédictions dans l'ordre de la microseconde sur un *FPGA*.

La prochaine section présente donc le thème principal de cette recherche, de son intérêt pour la société et les domaines impliqués.

1.2 Thème principal de la recherche proposée

La recherche proposée vise à utiliser des techniques d'IA implémentée sur *FPGA* pour traiter en temps réel les données de la « CookieBox ». Ce traitement ultra rapide permettra d'apposer un veto aux impulsions de rayons X qui ne respectent pas le cadre prescrit pour les expériences du LCLS-II. Ce veto permettra de réduire la génération de données produites par les expériences du LCLS-II à l'aide d'une compression par discrimination événementielle.

Nous proposons l'utilisation de RNA pour caractériser les impulsions de rayons X à l'aide du spectre en énergie rendu disponible par la « CookieBox ». Cependant, le développe-

3. *Central Processing Unit*
4. *Graphics Processing Unit*
5. *Field Programmable Gate Array*
6. *Edge Machine Learning (EdgeML)*

ment de RNA dans le contexte énoncé touche à plusieurs domaines spécifiques comme la physique nucléaire, la théorie de l'information, l'IA et l'informatique. À partir de ces quatre domaines, il sera possible de mieux définir l'information fournie par les détecteurs et comment traiter cette information en temps réel pour caractériser les impulsions de rayons X du LCLS-II.

1.3 Problématique

L'IA est un domaine en effervescence : les modèles d'apprentissage profond (AP) sont de plus en plus gros et de plus en plus performants. Cette tendance de l'AP n'est pas compatible avec les contraintes d'une application embarquée qui fonctionne en temps réel. Les algorithmes d'AP n'ont pas de limite théorique de taille, ce qui signifie qu'ils peuvent apprendre autant qu'ils ont accès à des données. Cette propriété est limitée par la puissance de calcul des plateformes matérielles nécessaires pour déployer les algorithmes. Comme le déploiement d'algorithmes sur des détecteurs scientifiques en périphérie implique nécessairement une empreinte matérielle limitée et une puissance de calcul restreinte, il n'est pas envisageable de toujours augmenter la taille des algorithmes afin de résoudre tous les problèmes. Accessoirement, la quantité de données nécessaire pour entraîner un RNA est aussi proportionnelle à la taille de ce dernier ; plus le RNA est gros, plus de données seront nécessaires pour l'entraîner. Les données scientifiques exigent beaucoup de ressources à produire et à stocker ce qui souligne aussi la nécessité de trouver des approches plus efficaces et plus durables pour la conception d'algorithmes d'AMP.

Pour résumer, les requis en latence et puissance du contexte de la présente recherche exigent un RNA compatible avec une solution embarquée à basse latence. Les travaux de recherche proposés offrent une solution à cette dichotomie pour réduire la génération de données massives lors des expériences scientifiques de grande envergure en utilisant l'IA.

Les prochaines sections précisent la résolution de cette problématique.

1.4 Question de recherche

Compte tenu du contexte et de la problématique, la question de recherche suivante est posée :

« Quelles techniques d'IA pourront être implémentées sur un *FPGA* et d'offrir des résultats d'inférence fiables, en moins de 100 µs tout en minimisant la puissance consommée ? »

1.5 Objectifs du projet de recherche

De cette question découlent les objectifs suivants. L'objectif global des présents travaux est la conception d'algorithmes d'IA implémentés sur *FPGA* pour traiter en temps réel les données de la « CookieBox ». Cet objectif découle directement de la question de recherche. À partir de cette question se dessinent également trois objectifs spécifiques :

1. **Objectif 1** : Concevoir des algorithmes d'IA capables d'utiliser les données numériques (réelles ou simulées) de la « CookieBox » afin d'extraire les caractéristiques d'une impulsion rayons X. Les algorithmes seront comparés en termes de qualité de prédiction.
2. **Objectif 2** : Optimiser les algorithmes développés à l'objectif 1 afin de réduire l'espace mémoire nécessaire à leur implémentation sur *FPGA* ainsi que de réduire la latence des inférences. Les performances des algorithmes optimisés seront comparées à celles des algorithmes non optimisés en termes de qualité de prédiction, d'espace mémoire et de latence. L'optimisation devra minimiser la dégradation de la qualité de prédiction tout en réduisant au maximum l'espace mémoire et la latence des inférences.
3. **Objectif 3** : Concevoir des algorithmes de RNA à décharges capables d'utiliser les données numériques (réelles ou simulées) de la « CookieBox » afin d'extraire les caractéristiques d'une impulsion rayons X. Il n'est pas attendu que les algorithmes de RNA à décharges soient implémentés sur *FPGA*. L'objectif 3 sert plutôt à démontrer la pertinence de RNA à décharges dans le contexte des présents travaux puisqu'ils sont habituellement moins énergivores. Les performances des algorithmes seront mesurées en termes de qualité de prédiction.

À terme, le système développé consistera en un algorithme d'IA implémenté sur *FPGA* qui démontrera sa capacité à servir d'outil de veto pour les données numériques générées par la « CookieBox ». Pour ce faire, le système développé doit générer des résultats d'inférence en moins de 100 µs tout en consommant le moins d'énergie possible. Il est envisageable de paralléliser plusieurs algorithmes ayant une latence allant jusqu'à 100 µs afin d'atteindre un débit global soutenant un taux d'un million d'événements par seconde.

1.6 Contributions originales

Au cours de cette maîtrise, les contributions principales des travaux ont consisté à combiner la quantification nonuniforme optimisée et les RNA sur *FPGA*. Cette méthode a permis de quantifier les données provenant de détecteurs scientifiques de manière optimale, réduisant

ainsi considérablement leur taille tout en préservant les informations utiles. En outre, elle a conduit à la création de RNA plus petits, plus rapides et plus économes en énergie, offrant ainsi la possibilité de traiter les données en temps réel pendant leur acquisition, ce qui n'est pas actuellement possible avec les systèmes existants.

Un système a été conçu pendant cette recherche, capable d'effectuer des inférences en moins de 2 µs sur un *FPGA*. Cette solution novatrice répond aux exigences en temps réel des applications embarquées en combinant les fondements théoriques de l'information et les techniques avancées d'IA. En outre, l'environnement de développement des RNA sur *FPGA* mis en place pendant cette étude a été bénéfique pour le groupe de recherche ainsi que pour le *SLAC National Accelerator Laboratory*. Ces travaux ouvrent la voie à de futurs projets dans ce domaine.

En résumé, les contributions principales de ces travaux permettront de réduire la génération de données massives lors des expériences scientifiques de grande envergure.

1.7 Plan du document

Les prochains chapitres plongent plus en profondeur dans la résolution de la problématique en lien avec le contexte.

L'état de l'art présente les thèmes principaux abordés ainsi que les travaux publiés dans le même domaine afin de mieux situer le projet de recherche. La « CookieBox », les mécanismes de compression de données, les RNA, et la réalisation matérielles de RNA sont notamment présentés.

Le développement relie tous ces éléments sous la forme de deux articles afin de répondre aux objectifs de recherches et ultimement de répondre à la problématique. Le premier et second article présentent respectivement les éléments algorithmiques et matériels permettant de répondre à la question de recherche.

Finalement, la conclusion présente une synthèse du projet de recherche. Elle résume l'atteinte des objectifs et la résolution de la problématique. Cette section finale met l'accent sur les contributions originales des travaux de recherches proposés. Elle discute des nouvelles perspectives scientifiques apportées par ces contributions et du chemin qu'il reste à faire pour répondre aux défis de demain.

CHAPITRE 2

ÉTAT DE L'ART

Les prochaines sections présentent respectivement la « CookieBox » et son fonctionnement, la théorie et l'état de l'art de l'information, la théorie et l'état de l'art des RNA et les détails concernant l'implémentation physique de RNA.

2.1 Le détecteur à balayage angulaire - La « CookieBox »

La « CookieBox » est un spectroscope à balayage angulaire. Elle permet l'échantillonnage du laser à rayons X de manière non destructive telle que représentée à la figure 2.1. Le rayon X traverse la partie centrale du détecteur qui contient du néon gazeux à basse pression. La faible densité du gaz ne modifie pas de manière significative le rayon X. Ce dernier reste virtuellement inchangé et peut donc servir au reste de l'expérimentation en cours. Le rayon X ionise le gaz émettant ainsi des photoélectrons d'énergies correspondantes. Les photoélectrons sont émis selon un dipôle suivant la polarisation du laser à rayons X.

La « CookieBox » comprend 16 canaux situés en périphérie du détecteur. Chacun de ces canaux est un spectromètre constitué de tubes à dérive coiffés de plaques à microcanaux (PMC) [1]. Son fonctionnement va comme suit : les tubes à dérive génèrent un champ magnétique et conditionnent l'énergie des photoélectrons ionisés par le laser à rayons X. Les PMC captent les photoélectrons conditionnés et amplifient leur signal pour qu'il soit numérisé à un taux de 4 milliards d'échantillons par seconde. Chaque spectromètre a donc un taux d'échantillonnage de 4 GHz. Un algorithme de recherche de pic utilise ce signal échantillonné pour calculer le temps de vol des photoélectrons. Le temps de vol est proportionnel à l'énergie du photoélectron et l'énergie est proportionnelle à la longueur d'onde. Ainsi, la matrice circulaire de 16 spectromètres permet de retrouver la distribution spectrale des photoélectrons ionisés par le laser à rayons X [3].

La « CookieBox » comprend aussi un laser infrarouge à polarisation circulaire qui tourne par rapport aux canaux de la « CookieBox ». Le champ électromagnétique du laser infrarouge accélère les photoélectrons ionisés suivant la polarisation, ce qui diminue leur temps

1. Microchannel Plate

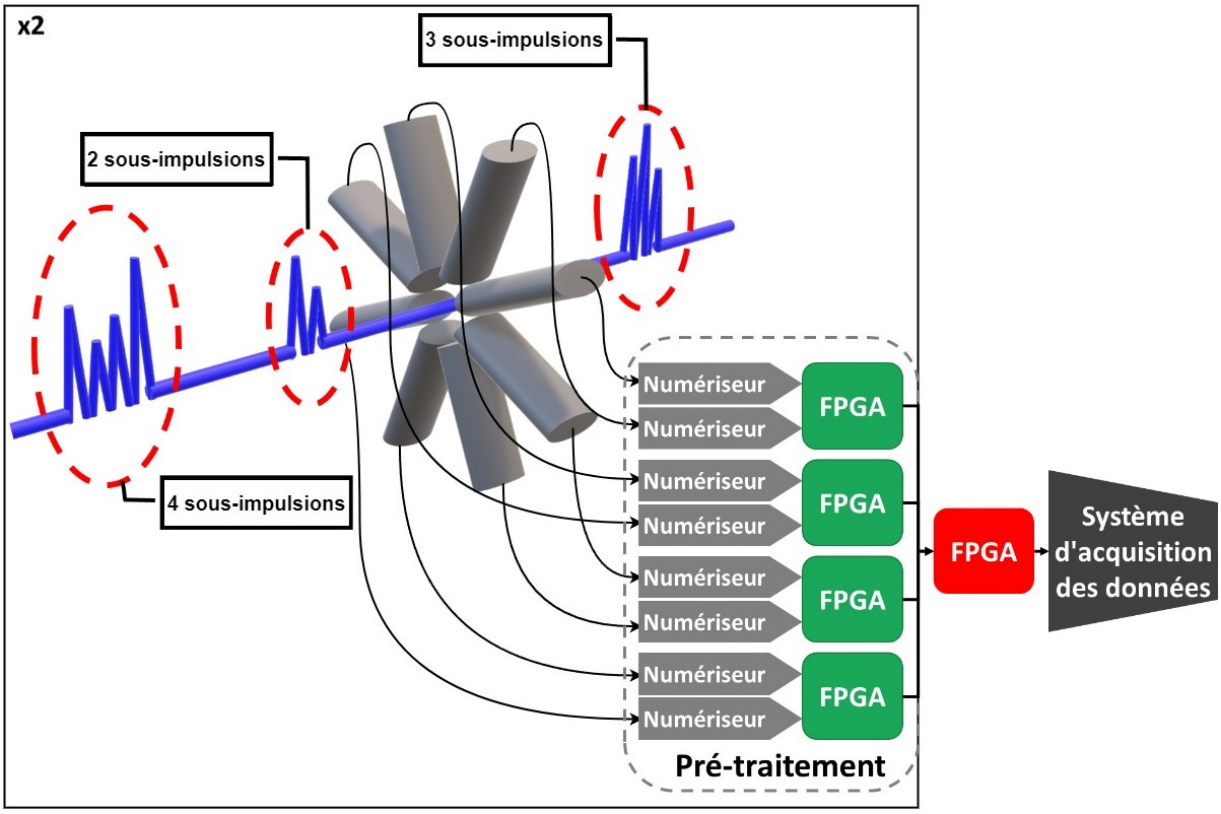

FIGURE 2.1 Schéma du flux de données de la « CookieBox ». Un circuit analogique traite les seize signaux en provenance des PMC, avant d'être numérisé. Chaque paire de numériseurs transfert les signaux numérisées aux *FPGA* (en vert), qui en extraient le spectre en énergie. Le *FPGA* central (en rouge) collecte les spectres de tous les canaux. L'algorithme d'IA développé dans ces travaux devra utiliser l'ensemble des spectres pour compter le nombre de sous-impulsions dans chaque impulsion principale. Il transmettra ensuite le compte au reste du système d'acquisition.

de vol et décale la distribution spectrale des photoélectrons en fonction de la période de rotation du champ [5, 6].

L'algorithme de reconstruction temporelle de l'impulsion de rayon X fonctionne en estimant la distribution des photoélectrons par une somme pondérée complexe de fonctions d'onde de base localisées temporellement et séquentiellement. Les coefficients complexes des fonctions d'onde de base sont déterminés par un calcul itératif, qui minimise l'erreur entre la distribution spectrale mesurée des photoélectrons et celle estimée par la reconstruction. La représentation temporelle des fonctions d'onde de base permet de retrouver le profil temporel de l'impulsion de rayon X. L'algorithme permet de caractériser une impulsion principale de rayons X avec une précision de l'ordre de l'attoseconde (10^{-18} s) à partir des mesures de distribution spectrale des photoélectrons [7]. Ce processus de

reconstruction itératif requiert une grande puissance de calcul. Les données de la « CookieBox » doivent être transférée sur des serveurs dotés de nœuds de calculs. Le processus de calcul, de transfert de données et l'algorithme de reconstruction prennent du temps, ce qui limite fortement la capacité de l'algorithme à fournir des résultats en temps réel. La discrimination des pulses avec cette méthode doit donc être faite en post-traitement.

Chaque impulsion principale de rayons X est composée de sous-impulsions. La distinction entre une impulsion de rayons X principale et ses sous-impulsions est faite à la figure 2.2. Ceci dit, le processus pour générer les impulsions de rayons X est majoritairement stochastique rendant le compte de sous-impulsions dans chaque impulsion principale impossible à prévoir [8].

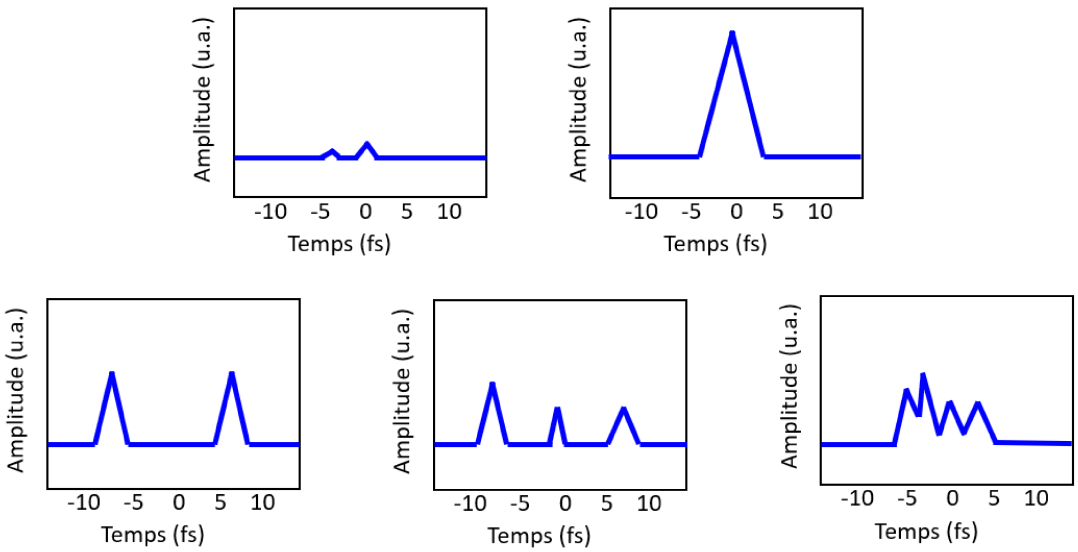

FIGURE 2.2 Exemple de cinq impulsions principales composées respectivement de zéro, un, deux, trois et quatre sous-impulsions (respectivement dans le sens de lecture)

Pour diminuer la génération de données massives sauvegardées par les expériences du LCLS-II, nous proposons l'utilisation de la « CookieBox » comme un outil de diagnostic afin de juger de la validité du rayon X en fonction de caractéristiques désirées et spécifiques à l'expérience en cours [4]. Pour ces travaux, le nombre de sous-pulses dans une impulsion principale est la seule caractéristique extraite. Lorsqu'un pulse est valide, le système sauvegardera les données pour cette impulsion. À l'inverse, l'outil appose un veto et les données sont rejetées sans être transférées, traitées et sauvegardées. C'est de cette manière qu'on applique la compression par discrimination événementielle. Ce système est présenté à la figure 2.1.

2.1.1 Simulation de la « CookieBox »

La *SLAC National Accelerator Laboratory* dispose d'un simulateur de données de la « CookieBox » pour faciliter le développement d'algorithmes servant au détecteur comme c'est le cas dans ces présents travaux. Ce simulateur est en langage de programmation *Python* et il porte le nom de « CookieSimSlim » [9].

La simulation Monte-Carlo reproduit le balayage angulaire de la « CookieBox ». Ce type de simulation est très répandu en ingénierie et en science en général [10]. La simulation permet de générer des ensembles de données contenant la distribution spectrale des photoélectrons tout en tenant compte de la variabilité des impulsions de rayons X et du mode de fonctionnement de la « CookieBox » (période du laser infrarouge, polarisation du laser infrarouge, conditionnement par les tubes à dérive ...) [3, 9, 11]. Toutes ces métadonnées sont organisées dans un fichier HDF5 avec les données de sortie correspondantes. Notamment la distribution spectrale des photoélectrons puisque c'est ce qui est utilisé pour caractériser les impulsions de rayons X. La distribution spectrale des photoélectrons générés par « CookieBox » a une résolution de 16 bits. Puisque le simulateur est en langage de programmation *Python*, il fournit en sortie la distribution spectrale des photoélectrons avec une résolution de 32 bits en virgule flottante [9].

L'avantage de ce simulateur est qu'il permet d'avoir un accès direct aux caractéristiques de l'impulsion de rayons X ayant produit une distribution spectrale de photoélectrons particulière.

Compte tenu du taux de répétition de 1 MHz du LCLS-II, le veto, et donc la caractérisation des impulsions principales de rayons X, doit être fait dans l'ordre de la microseconde. Cependant, l'algorithme de reconstruction actuelle ne permet pas d'atteindre ce requis de latence comme explicité plus haut. Ainsi, nous proposons l'utilisation d'algorithmes AMP pour caractériser chaque impulsion principale de rayons X en ligne et en temps réel. La prochaine section présente comment traiter les données de la « CookieBox » afin de permettre l'utilisation des algorithmes d'AMP en temps réel.

2.2 Théorie de l'information

La donnée numérique est une manière pratique que nous avons trouvée pour représenter, transporter et stocker l'information dans nos systèmes numériques à l'aide de codes formés strictement de « 0 » et de « 1 » (des bits). Il est avantageux d'utiliser le moins de données possible pour représenter l'information puisque cela permet de réduire l'utilisation de ressources informatiques. C'est le principe de la compression de données numériques [12].

La prochaine section explique donc comment tirer avantage des méthodes de compression de données pour utiliser des algorithmes d'AMP en temps réel.

On présente d'abord comment mesurer la quantité d'information dans des données numériques. On montre ensuite comment la modélisation de cette information permet de compresser les données numériques et de diminuer l'utilisation de ressources computationnelles. Finalement, on expose comment la quantification des données pourrait servir à compresser le spectre généré par la « CookieBox ».

2.2.1 L'information contenue dans les données

En théorie de l'information, l'entropie de premier ordre (H) est utilisée pour mesurer la quantité d'information d'une source de données [13]. La formule de l'entropie de premier ordre d'une source S qui produit un ensemble de données $X_0, X_1, X_2, ..., X_{N-1}$ chacune avec une probabilité d'occurrence $P(X_0), P(X_1), P(X_2), ..., P(X_{N-1})$ est donnée à l'équation 2.1. La base deux de la fonction logarithmique est utilisée pour exprimer l'entropie en bits.

$$\mathrm{H}(S) = -\sum_{i=0}^{N-1} P(X_i) \log_2 P(X_i) \tag{2.1}$$

En observant l'équation 2.1, on peut remarquer que lorsque toutes les valeurs X_i de l'ensemble de données $X_0, X_1, X_2, ..., X_{N-1}$ ont la même fréquence d'occurrence, l'entropie correspond simplement au nombre de bits nécessaires pour représenter les données. Par exemple, si les données sont codées sur 8 bits et que chaque valeur de 0 à 255 a une probabilité d'apparition égale dans l'ensemble de données, alors l'entropie est précisément de 8 bits.

$$\mathrm{H}(S) = -\sum_{i=0}^{N-1} P(X_i) \log_2 P(X_i) \Bigg|_{P(X_i)=\frac{1}{256}, N=2^8}$$

$$\mathrm{H}(S) = -\sum_{i=0}^{255} (\frac{1}{256}) \log_2 (\frac{1}{256})$$

$$\mathrm{H}(S) = 8\ bits$$

Cependant, un ensemble de données où la probabilité d'occurrence de chaque valeur est uniforme est en réalité un ensemble aléatoire. Il pourrait avoir été généré par un dé à 256 faces ou tout autre processus stochastique. Une chose à retenir de cet exemple est que les redondances statistiques des données sont là où résident l'information utile et le potentiel de compression des données.

2.2.2 La modélisation

La notion d'information est liée au contexte dans lequel elle est transmise ou reçue. On ne transmet pas la même information sur notre état lorsqu'on bâille à une rencontre qui s'éternise que lorsqu'on bâille suite à une nuit blanche. L'information sur l'état d'une personne qui bâille n'est alors pas définie par l'action de bâiller, mais par le contexte dans lequel elle bâille. En compression de données numériques, on utilise un modèle pour définir le contexte et extraire l'information utile des données.

La modélisation est la première étape importante dans une chaîne de compression [12]. Pendant la modélisation, on essaie d'extraire la redondance statistique des données pour l'inclure dans un modèle. Le modèle peut être physique, dans lequel la redondance statistique est directement liée au phénomène physique qui génère les données. Il peut également être probabiliste, où une certaine redondance statistique est observée, mais pas directement explicable. Néanmoins, il est généralement possible de l'exploiter.

D'autres étapes dans la chaîne de compression, telles que le codage des données, visent également à exploiter les redondances statistiques pour réduire la taille des données générées [12]. Cependant, toutes ces étapes nécessitent des ressources informatiques, il est donc préférable de modéliser le plus tôt possible et le plus fidèlement possible dans la chaîne de compression pour éviter de transférer et traiter inutilement des données redondantes et ainsi économiser des ressources computationnelles.

2.2.3 La quantification nonuniforme optimisée

En pratique, il est rarement possible d'atteindre une modélisation causale parfaite en raison de la nature complexe d'un phénomène physique ou de l'incertitude des instruments de mesure. Ainsi, des méthodes comme la quantification nonuniforme optimisée sont applicables à tout ensemble de données. C'est une technique qui permet des réattribuer les valeurs à travers un ensemble de données afin d'optimiser la distribution statistique de ce dernier [12]. Ce quantificateur permet donc de compresser les données en réduisant leur résolution (c.-à-d. le nombre de bits par échantillons), mais il agit aussi comme modèle. Il permet ainsi de compenser l'effet de la diminution de résolution par une connaissance statistique des données.

La quantification au sens large consiste à attribuer à chaque échantillon d'un signal une valeur numérique qui appartient à un ensemble fini de valeurs possibles. Pour optimiser la redondance statistique des données, le quantificateur optimisé réattribue ces valeurs vers un ensemble de valeurs possibles nonuniformément distribuées dans le but de minimiser l'erreur quadratique moyenne de quantification (*MSQE*) [2]. Le *MSQE* représente le bruit de quantification et il permet de juger de ses performances. La formule du *MSQE* est donnée à l'équation 2.2 [12].

$$\mathrm{MSQE}(X, Q(X)) = \frac{1}{N} \sum_{i=0}^{N-1} (x_i - Q(x_i))^2 \tag{2.2}$$

où X est l'ensemble de données discrètes originales et $Q(X)$ est l'ensemble de données quantifiées.

Pour extraire la connaissance statistique des données, le quantificateur nécessite d'être entraîné. L'algorithme de Lloyd-Max permet d'itérativement minimiser le *MSQE* en utilisant les méthodes du k plus proches voisins et du k moyen [14, 15]. Puisque le *MSQE* est stochastique, il augmente l'entropie du signal, mais ne contient pas d'information utile comme démontré à la sous-section 2.2.1 (exemple du dé). On a donc intérêt à minimiser le *MSQE* afin d'augmenter le rendement des données [12]. C'est ce que fait la quantification nonuniforme optimisée.

La distribution spectrale des photoélectrons générée par la « CookieBox » est représentée avec une résolution de 16 bits, ce qui signifie que le spectre en énergie des photoélectrons est défini par un ensemble fini de 65 536 (2^{16}) valeurs possibles. Toutes les informations nécessaires pour caractériser l'impulsion de rayons X sont strictement contenues dans ce spectre. Ainsi, comme expliqué à la sous-section 2.2.2, il serait judicieux de compresser ce spectre dès que possible dans la chaîne d'acquisition de données. Pour cela, la quantification nonuniforme optimisée pourrait être utilisée afin de réduire le nombre de valeurs d'énergie dans le spectre, tout en préservant l'information essentielle qu'il contient. Cette technique permettrait de diminuer les ressources informatiques nécessaires à tout algorithme d'AMP chargé de traiter ce spectre, y compris les RNA.

En résumé, en utilisant un quantificateur uniforme pour diminuer la résolution des données, on réduit la bande passante du système, mais cela s'accompagne également d'une dégradation du signal. En revanche, la quantification nonuniforme optimisée permet de minimiser cette dégradation et donc d'utiliser moins de données tout en préservant l'intégrité

2. Mean-Square Quantization Error

de l'information qu'elles contiennent. Cela permet de réduire l'utilisation des ressources informatiques sans altérer les mesures scientifiques.

L'interprétation précise des données est cruciale en sciences. Il est pertinent de modéliser l'information dès le début de la chaîne d'acquisition afin de conserver cette interprétation en plus de minimiser les ressources computationnelles. La quantification nonuniforme optimisée est une méthode simple et efficace pour modéliser les données. Le RNA est également un outil de modélisation puissant qui tire profit de la redondance statistique des données. La prochaine sous-section présentera donc les mécanismes de modélisation propres au RNA.

2.3 Théorie et état de l'art des réseaux de neurones artificiels

Le RNA est un type d'algorithme d'une branche de l'IA appelée l'apprentissage machine (AM). Ces algorithmes permettent aujourd'hui la résolution de problèmes de régressions et classifications complexes dans plusieurs domaines. Leurs propriétés non linéaires leur permettent l'approximation de n'importe quelle fonction continue (avec un degré d'exactitude arbitraire) [16]. L'hypothèse posée est que le RNA constitue un outil puissant qui pourrait bien servir aux exigences de faible latence et de basse complexité algorithmique demandées par le système de diagnostic énoncé à la section 1.1.

Il existe plusieurs façons d'interpréter le fonctionnement d'un RNA et la plus intuitive dépend souvent du contexte dans lequel on utilise ce dernier. Par exemple, dans le contexte du génie informatique, il est pratique de conceptualiser le RNA comme une suite d'instructions et d'opérations arithmétiques tandis qu'en détection d'objet par vision par ordinateur (*computer vision*), il est plus commun de conceptualiser le RNA comme un extracteur de caractéristiques propres à un type d'image. La prochaine section vise donc à faire le pont entre le contexte de la recherche proposée et la théorie des RNA.

2.3.1 Le réseau de neurones artificiels

Pour commencer, le RNA s'inspire de l'architecture densément interconnectée de l'encéphale biologique [17, 18]. Plusieurs analogies sont possibles entre un neurone biologique et un neurone artificiel, mais ce type de vulgarisation peut parfois biaiser la conceptualisation d'un RNA ou même la complexifier inutilement. Ainsi, la prochaine sous-section présente le RNA d'un point de vue arithmétique puis algébrique, afin de mieux situer la pertinence de ce dernier dans le contexte des présents travaux.

Le neurone artificiel est un nœud [3] de calcul et cinq éléments sont nécessaires pour décrire son fonctionnement : l'entrée du nœud (x), les paramètres du nœud (w et b), le nœud même, la fonction d'activation du nœud (g), et la sortie du nœud (y) [17, 16]. Ces éléments sont illustrés à la figure 2.3. La première implémentation de ce nœud de calcul fut le perceptron où g prend la forme d'un seuil fixe [19].

Les fonctions d'activation les plus courantes sont : la fonction sigmoïde, la fonction unité linéaire rectifiée ($ReLU$), la fonction *softplus* [4] et la fonction tangente hyperbolique (tanh). Ces fonctions sont montrées à la figure 2.4 [16].

3. Le terme nœud plutôt que neurone est favorisé dans cette section afin de mieux situer le RNA dans un contexte informatique.

4. La fonction *softplus* est une version plus douce de la fonction $ReLU$.

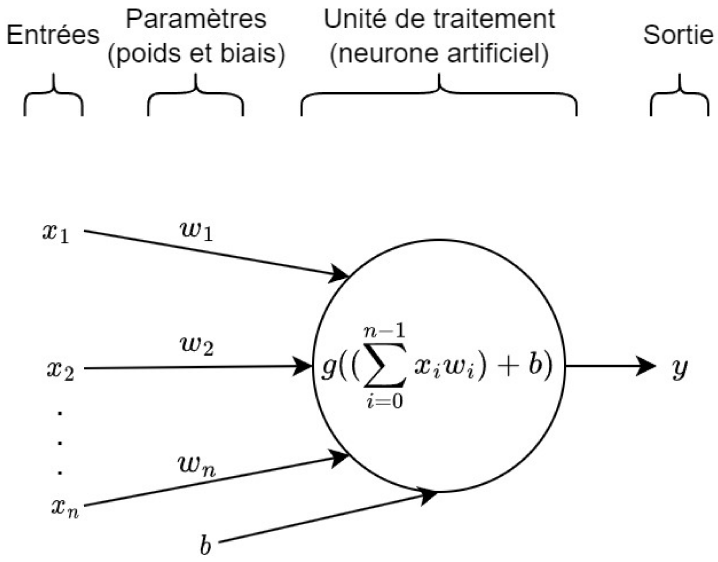

FIGURE 2.3 Diagramme d'un neurone artificiel avec x, w, b, g et y qui sont respectivement les entrées, les poids, le biais, la fonction d'activation et la sortie.

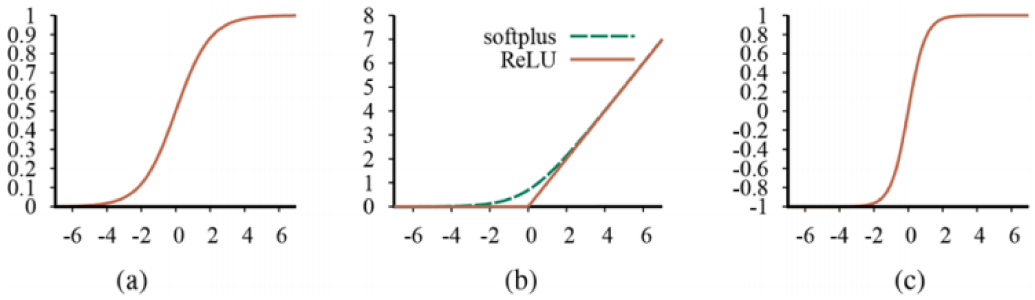

FIGURE 2.4 Fonctions d'activation couramment utilisées : (a) la fonction logistique ou sigmoïde ; (b) la fonction *ReLU* et la fonction *softplus* ; (c) la fonction tanh. La fonction d'activation permet au RNA de représenter n'importe quelle fonction arbitraire. [16]

Formellement, le nœud pondère de ses entrées en fonction de ses paramètres puis applique une fonction d'activation non linéaire comme il est montré à l'équation 2.3 [20]. L'équation 2.3 montre que le poids w_i pondère l'entrée x_i, qu'on applique une translation de b à la somme pondérée et finalement la transformation g.

$$y = g((\sum_{i=0}^{n-1} x_i w_i) + b) \tag{2.3}$$

Notez que la somme pondérée de l'équation 2.3 peut aussi s'écrire sous la forme d'un produit scalaire :

$$y = g(\vec{x} \cdot \vec{w} + b) \tag{2.4}$$

On peut ainsi représenter l'opération faite au nœud comme la projection orthogonale du vecteur d'entrée $\vec{x}$ sur le vecteur poids $\vec{w}$ (suivi d'une addition de b et d'une transformation g). Cependant, cette représentation algébrique prend tout son sens lorsqu'on connecte l'entrée x à plusieurs neurones artificiels. On parle alors d'un RNA densément connecté. La figure 2.5 montre l'architecture de base d'un RNA. Ce RNA a une couche d'entrée, une couche cachée et une couche de sortie.

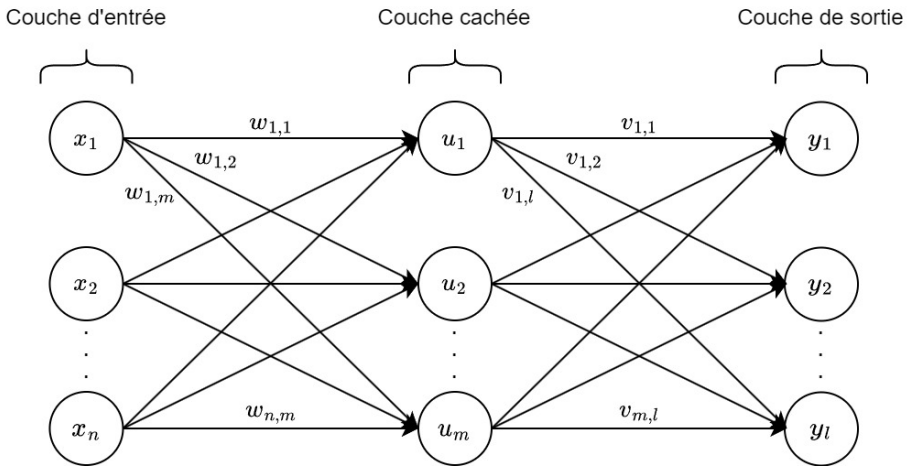

FIGURE 2.5 Architecture de base d'un RNA densément connecté. La représentation du problème par les valeurs u_i du RNA est appelée espace latent. Il s'agit d'une représentation abstraite, mais discriminante du problème.

Ainsi, le vecteur intermédiaire $\vec{u}$ est le résultat du produit matriciel entre le vecteur $\vec{x}$ et la matrice de poids $\vec{w}$ et le vecteur de sortie $\vec{y}$ est le résultat du produit matriciel entre le vecteur $\vec{u}$ et la matrice de poids $\vec{v}$. Bien-sûr avec l'ajout des biais b et des transformations g qui ne sont pas montrés dans la figure 2.5 pour alléger le schéma.

Cette précision théorique est importante à deux niveaux dans le contexte de ces travaux :

1. Premièrement, cela montre directement les éléments qui dicte la complexité algorithmique du RNA : le nombre de nœuds dans les couches d'entrée, cachées et de sortie. Ceci dit, la couche de sortie est souvent bien définie lorsqu'on utilise un RNA puisque c'est ce qu'on cherche à prédire. Par contre, la taille de la couche cachée dépend de la complexité du problème à résoudre qui, à son tour, dépend de la défini-

tion de la couche d'entrée. On a donc avantage à ce que la couche d'entrée soit petite pour diminuer la complexité algorithmique, mais qu'elle détienne tout de même le maximum d'information utile sur le problème à résoudre.

2. Deuxièmement, la méthode de calcul par produit matriciel est fortement exploitée pour paralléliser la propagation avant dans un RNA. Cette caractéristique de parallélisme inhérent au RNA est un atout pour répondre aux requis de faible latence des présents travaux.

L'apprentissage d'un RNA

Comme montré par l'équation 2.3, ce sont les paramètres d'un RNA qui dictent les calculs effectués par ce dernier. On dit alors que la connaissance «a priori» d'un RNA est contenue dans les valeurs des poids et des biais de chaque nœud. Cette connaissance statistique permet de faire la traduction d'un signal d'entrée à un signal de sortie. Cette prédiction est appelée inférence. On entraîne un RNA pour améliorer sa connaissance «a priori» d'un problème pour ainsi produire de bonnes inférences. Ce sont les valeurs des poids et des biais qui sont ajustées pendant cet entraînement [16]. L'entraînement est à lui seul un sujet d'étude en IA. Ainsi, cette sous-section vise à présenter les éléments de bases de l'apprentissage d'un RNA et à mettre en relief certaines considérations en lien avec le contexte de ces travaux. Notamment la disponibilité et la représentativité des ensembles de données scientifiques.

La rétropropagation du gradient de l'erreur est la méthode la plus courante pour entraîner les RNA. C'est un algorithme d'entraînement qui consiste à modifier les paramètres du RNA en fonction du gradient de l'erreur entre l'inférence du RNA et la réponse attendue [21]. Très souvent cet algorithme d'apprentissage est utilisé de manière supervisé (c.-à-d. que l'ensemble de données doit être étiquetée). Des problèmes qui peuvent subvenir lors de l'entraînement de RNA à plusieurs couches cachées par rétropropagation sont la disparition du gradient ou l'explosion du gradient. Cela se produit lorsque les gradients deviennent soit trop petits pour mettre à jour efficacement les poids du RNA ou trop grands pour que l'entraînement converge [22, 23, 24].

Il y a plusieurs manières d'implémenter la rétropropagation du gradient de l'erreur et celles-ci sont incluses dans des outils de développement comme *MATLAB*[5], *TensorFlow*[6] ou *Microsoft Cognitive Toolkit*[7]. Même si la qualité de l'apprentissage peut être affectée par plusieurs paramètres, les outils de développement permettent somme toute d'arriver

5. https ://www.mathworks.com/products/matlab.html
6. https ://www.tensorflow.org/
7. https ://docs.microsoft.com/en-us/cognitive-toolkit/

à des résultats satisfaisants dans des délais raisonnables [25, 26]. Ainsi, il est important de se pencher sur l'ensemble de données qui est utilisé pour faire l'entraînement du RNA puisque c'est cet ensemble qui définit ultimement le problème que le RNA doit résoudre.

L'entraînement nécessite minimalement deux ensembles de données : les données d'entraînement et les données de test. Les deux ensembles contiennent les données d'entrée et les données de sortie désirées du RNA. Les données d'entraînement sont utilisées pour l'entraînement à proprement dit. Les données de tests sont des données qui n'ont jamais été présentées au RNA pendant l'entraînement et qui vérifient s'il est en mesure de bien généraliser le problème. Si le RNA permet de bien calculer les valeurs de l'ensemble d'entraînement, mais pas de l'ensemble de tests, on dit alors que le RNA est surajusté (il a mémorisé l'ensemble d'entraînement par cœur) [16, 17]. Cela survient majoritairement lorsqu'il y a un écart trop grand entre la taille d'un RNA et la taille de l'ensemble d'entraînement. On peut donc rapetisser le RNA ou grossir la base de données.

L'entraînement supervisé nécessite donc un ensemble comprenant les entrées et sorties attendues pour chaque évènement. Il faut donc prévoir une méthode pour générer les entrées, mais aussi pour générer les sorties attendues. Par exemple, la base de données *ImageNet* comprend plus de 100 000 classes de mots/concepts avec en moyenne 1000 images pour illustrer chaque classe. Cela fait donc 100 000 000 images qui ont toutes été étiquetées à la main [27]. De plus, l'ensemble d'entraînement utilisé doit bien représenter statistiquement le problème à traiter. Tout biais présent dans l'ensemble d'apprentissage sera reflété dans les inférences du RNA. Finalement, il est important que ce RNA soit entraîné avec des données statistiquement similaires à celles qu'il traitera.

Ainsi, pour utiliser le RNA dans une chaîne d'acquisition de données scientifiques, il faut prévoir une ou des méthodes pour générer et étiqueter les bases de données. Il faut aussi que l'ensemble de données servant à l'entraînement soit caractérisée et validée par des experts afin qu'elle ne contienne aucun biais méthodologique. Finalement, il faut prévoir une calibration et/ou un conditionnement de la chaîne de traitement qui précède le RNA afin d'utiliser ce dernier dans une plage qui inclut les statistiques de l'ensemble d'apprentissages. Dans le cas de ces travaux, le simulateur « CookieSimSlim », présenté à la sous-section 2.1.1 permet de générer ces bases de données afin développer les algorithmes, mais aussi afin de tester différentes méthodes de conditionnement.

Ayant traité de l'unité de base des RNA (c.-à-d. le neurone artificiel) et de l'entraînement des RNA, la prochaine section présente différentes architectures de RNA pouvant accomplir des tâches complexes.

2.3.2 Les architectures des réseaux de neurones artificiels

À la section 2.3.1, on a présenté les concepts et la théorie de bases des RNA. Notamment, comment les données sont traitées par les nœuds du RNA et comment entraîner les paramètres d'un RNA. Des considérations pratiques quant à l'utilisation des RNA en science ont aussi été abordées. Dans la prochaine section, on présente trois architectures de RNA : le RNA entièrement connecté, le RNA à convolution et le RNA récurrent. Toutes ces architectures ont des avantages et inconvénients. Les principes globaux des architectures seront présentés et leurs points positifs et négatifs relevés par rapport aux considérations pratiquent de l'utilisation des RNA dans le contexte de la présente recherche.

Le réseau de neurones artificiels entièrement connecté

Le RNA entièrement connecté est majoritairement couvert au début de la section sous la forme 2.3.1. Cependant, certains détails théoriques y sont propres.

Selon le théorème d'approximation universelle, un RNA entièrement connecté avec une seule couche cachée peut modéliser n'importe quelle fonction avec un dégrée de précision arbitraire [20, 17]. Cela peut nécessiter une couche très large (avec beaucoup de nœuds). Le théorème d'approximation universelle est à la base des RNA, mais plus récemment, la même preuve a été faite au sujet de la profondeur du RNA [28]. De la même manière, le RNA peut nécessiter un grand nombre de couches cachées pour modéliser certaines fonctions.

Donc, l'avantage du RNA entièrement connecté est que c'est une architecture simple et intuitive. Elle peut aussi, en théorie, résoudre n'importe quel problème [29]. Plus le problème est complexe, plus le RNA devra être gros. C'est en pratique que cela devient un désavantage. Comme soulevé précédemment, plus le RNA est gros, plus la base de données pour entraîner le RNA doit être grosse et diversifiée pour prévenir le surapprentissage. Il peut être compliqué de collecter assez de données d'entraînement et cela ne devient plus possible à de très grands ordres de grandeur (dépendamment du contexte bien sûr) [30].

Les modèles d'AP démontrant les meilleures performances au concours *ImageNet Large Scale Visual Recognition Challenge* (*ILSVRC*) ont tous recours à la modification de leur base de données afin de pallier aux problèmes d'entraînement lier à un gros RNA [27, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43]. Notez que ces modèles ne sont pas spécifiquement des RNA entièrement connectés. Par contre, le RNA entièrement connecté est l'architecture la plus prompte à grandir rapidement en dimensionnalité. Il faut donc considérer qu'un RNA entièrement connecté très gros et profond aurait aussi besoin de techniques comme l'augmentation des données ou la normalisation des lots lors de l'entraînement [30].

Ainsi, à un certain point, il ne devient plus réaliste d'utiliser ce type de RNA pour traiter les problèmes à haute dimensionnalité. Dans un tel cas, on se tourne vers les RNA à convolution (RNAC) qui ont notamment l'avantage de nécessiter moins de paramètres qu'un RNA entièrement connecté pour un nombre et une taille de couche similaire [31].

Le réseau de neurones artificiels à convolution

Les RNAC sont largement utilisés dans des applications de traitement d'image. C'est le principe d'invariance locale qui rend le fonctionnement du RNAC possible. Dans le cas du traitement d'une image, on parle alors d'invariance spatiale locale ; il est estimé qu'une certaine caractéristique d'une image comme un coin de mobilier, sera semblable d'une petite section locale à l'autre d'une image [16]. Contrairement au RNA entièrement connecté, les RNAC regroupent les poids en fenêtres d'analyse appelées filtres. Le nombre et la taille des filtres d'une même couche du RNA permettent alors d'exploiter les invariances locales à différentes échelles et ainsi d'extraire l'information redondante du signal sous forme de caractéristiques. Cette extraction de caractéristiques se fait par un processus appelé convolution. Lors de la convolution, les filtres d'analyse glissent successivement sur les sections du signal d'entrée en y appliquant un produit scalaire et une fonction d'activation pour produire le signal de sortie. Les filtres peuvent glisser par pas entier positif et le signal d'entrée peut être rempli afin de dicter la taille du signal de sortie [44].

La connaissance a priori du signal par le RNA est alors contenue dans les poids des filtres. Ce sont ces valeurs qui sont ajustées lors de l'entraînement du RNAC. La structure de base d'un RNAC permettant d'identifier un chat est montrée à la figure 2.6. En général, un RNAC de base est composé d'une couche d'entrée, puis d'une succession de couches de convolution et de décimation afin d'extraire les caractéristiques de l'entrée tout en réduisant la dimensionnalité du problème. La figure 2.6 est simplifiée afin de donner une intuition du fonctionnement d'un RNAC, mais on y voit tout de même l'extraction de caractéristiques et la réduction de dimensionnalité.

Ainsi, le RNAC a l'avantage de mieux gérer les problèmes à haute dimensionnalité. Cependant, même si en théorie le RNAC sert à restreindre la taille d'un RNA, en pratique il est souvent très profonds et très gros. Des RNAC comme *ResNet* ou *VGGNet* contiennent des millions de paramètres [33, 45]. De tels modèles ne sont pas adaptés pour une utilisation en périphérie. Même les versions « mobiles » de ces modèles contiennent elles aussi des millions de paramètres [46, 47]. Finalement, un désavantage des RNAC en lien direct avec la problématique est que l'utilisation de plusieurs couches, contenant toutes plusieurs filtres, augmente le nombre d'opérations à faire lors de l'inférence ce qui peut affecter négativement la latence du réseau.

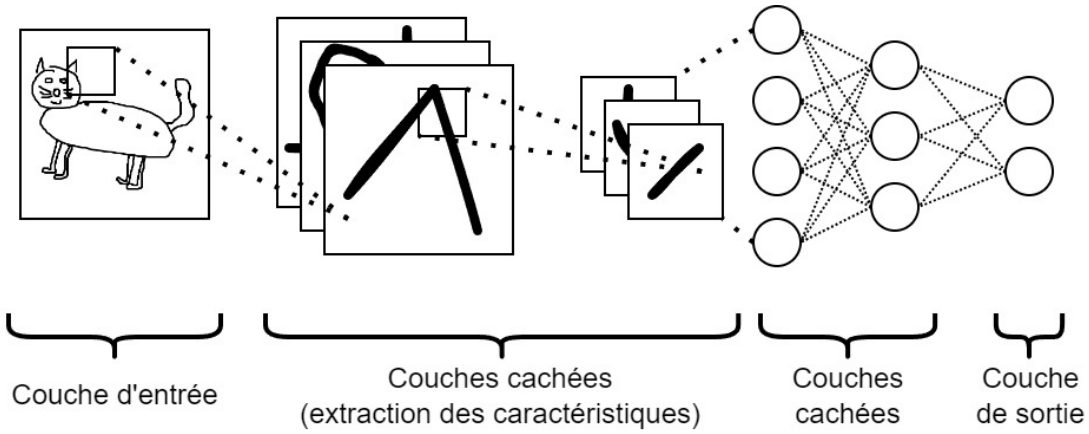

FIGURE 2.6 Architecture de base d'un RNAC. Les filtres de convolutions des couches cachées extraient les caractéristiques discriminantes de l'image en exploitant ses invariances locales. L'espace latent du RNAC est donc une représentation successive de plusieurs dégrées de caractéristiques. La dernière section est un RNA entièrement connecté pour faire l'inférence (« résoudre » le problème).

Jusqu'à maintenant, on a toujours considéré les différentes entrées du RNA comme indépendantes l'une de l'autre. Néanmoins, plusieurs problèmes en science et en génie sont dynamiques ; l'état du problème à un temps t influence l'état du même problème à un temps t'. Le prochain type d'architecture tient compte de ce dynamisme.

Le réseau de neurone artificiel récurrent

Les RNA récurrents (RNAR) sont une classe de RNA qui permet la modélisation de séries temporelles. Le RNAR utilise une rétroaction en plus d'une connexion directe entre les unités pour modéliser la dynamique sous-jacente d'une séquence [48, 49, 50]. Ce type d'architecture suscite un intérêt grandissant de plusieurs communautés pour la modélisation de séries temporelles dû à l'omniprésence de ces dernières dans nos systèmes technologiques [51]. Les séries temporelles se retrouvent en effet dans des domaines comme la météorologie, l'économie et la médecine [48, 52]. De manière analogue au RNAC, c'est le principe d'invariance locale temporelle qui permet le fonctionnement du RNAR possible. Il est cependant observé que les dépendances à long terme des séquences sont difficiles à modéliser à l'aide de RNAR dû aux problèmes de la disparition du gradient et d'explosion du gradient lors de l'entraînement de ces derniers [50, 24].

L'unité *Long short-term memory* (*LSTM*) a premièrement été développée pour contrer ce problème [53]. Bien que l'unité *LSTM* soit résistante au problème de disparition du gradient, elle est critiquée pour son caractère ad hoc et la fonction peu intuitive de ses éléments algorithmiques. Il est alors soulevé que ladite unité n'est peut-être pas optimale et que d'autres architectures pourraient être mieux adaptées pour traiter les problèmes

séquentiels [54]. À l'instar de l'unité *LSTM*, l'unité *Gated recurrent unit* (*GRU*) permet les mêmes mitigations, mais en offrant de meilleures performances pour environ toutes les tâches de modélisation du langage [54]. Avec quelques modifications, les performances du *LSTM* peuvent cependant se rapprocher du *GRU* [54, 55].

Ainsi, le RNAR a l'avantage de pouvoir incorporer la notion de temps dans la résolution du problème, mais il n'y a pas encore de consensus de la communauté sur les fondements théoriques d'une telle architecture.

Le Tableau 2.1 offre une synthèse des différents types de RNA présentés ci-haut en plus de présenter leurs avantages et inconvenants principaux.

TABLEAU 2.1 Tableau récapitulatif des différents types d'architectures de RNA

Type de RNA	**Fonction générale**	**Avantage(s)**	**Désavantage(s)**
RNA entièrement connecté	Systèmes multivariables	Hyperparamètres intuitifs ; très présent dans la littérature ; rapide d'implémentation	Non adapté pour problèmes à hautes dimensionnalitées
RNA à convolution	Images	Très présent dans la littérature ; adapté pour problèmes à haute dimensionnalité	Tendance vers l'AP ; affecte négativement la latence à l'inférence
RNA récurrent	Séries temporelles	Traite les systèmes dynamiques	Problème de disparition ou d'explosion du gradient ; jugé ad hoc par la communauté

2.4 Réalisations matérielles de réseaux de neurones artificiels

Il est important que les algorithmes précédemment présentés soient optimisés pour la plateforme matérielle sur laquelle ils seront déployés afin d'exploiter leur plein potentiel. Ce type d'implémentation est un RNA matériel (RNAM) [8]. Ce type de plateforme est aussi appelé engin d'inférence dédié puisqu'elle accélère l'inférence du RNA en profitant d'une architecture physique conçue à cet effet [56].

8. *Hardware Neural Network* (HNN)

Dans le contexte de la recherche proposée, le veto doit être disponible en moins de 100 µs pour être utile au reste du système d'acquisition de données. L'accélérateur d'inférence devra donc démontrer de sa capacité à fournir des résultats d'inférences avec une très faible latence.

La prochaine section présente d'abord une revue des différentes plateformes physiques pour le déploiement de RNAM suivi des considérations techniques pour une implémentation de RNA sur *FPGA*. La dernière section décrira les outils logiciels et les techniques d'optimisation pour programmer un RNA sur un *FPGA*.

2.4.1 Plateformes physiques

Plusieurs types de matériel informatique existent pour faire l'inférence de RNA et le bon choix dépend du contexte d'utilisation. Les *CPU* peuvent faire l'entraînement et l'inférence de RNA. Ils sont aussi très accessibles. Il s'agit donc d'une bonne base de comparaison pour d'autres plateformes dédiées aux RNA. Par ailleurs, étant donné que la base mathématique des RNA et de la manipulation d'images sont des tâches similaires, les *GPU* sont aussi très populaires pour faire l'entraînement et l'inférence de RNA. Par contre, les accélérateurs qui nous intéressent dans le contexte d'informatique de périphérie sont les plateformes reprogrammables comme les *FPGA* et les circuits sur mesure comme les circuits intégrés à application spécifique (*ASIC*) [9].

D'abord, les *CPU* sont des processeurs généraux qui ne sont pas dédiés aux RNA. De même, la logique d'application doit s'exécuter par l'entremise d'un système d'exploitation, de pilotes et de logiciels. Ceci en facilite l'utilisation pour du développement, mais ajoute des étapes qui peuvent augmenter la latence lors d'inférences. La puissance consommée les rend aussi peu utilisables pour des applications embarquées [56].

Les *GPU* accélérèrent grandement le traitement de RNA profonds comparativement aux *CPU*. Ils ont aussi un meilleur rendement énergétique que le *CPU* lors de l'entraînement de RNA. Par contre, lors de l'inférence, leur rendement énergétique est normal [57]. Les *GPU* ont aussi une architecture fixe, ce qui limite leur utilité pour le prototypage. Pour ces raisons, le *GPU* n'est, lui aussi, pas adapté pour des applications embarquées [56, 57].

En revanche, les accélérateurs sur mesure comme les *FPGA* et les *ASIC* nécessitent moins d'opérations et moins d'accès mémoire pour faire des inférences. Ceci s'explique, car l'architecture des circuits peut être optimisée pour chaque architecture de RNA [56, 58]. Cependant, la conception et la fabrication de *ASIC* sont coûteuses et nécessitent de longs cycles de développement [57]. Ceci limite son utilité pour le prototypage.

9. *Application-specific integrated circuit*

Ensuite, certains accélérateurs sont conçus pour faire l'entraînement et l'inférence de RNA tandis que d'autres le sont uniquement pour de l'inférence. Les plateformes conçues pour de l'entraînement peuvent être utilisées pour de l'inférence, mais l'inverse n'est pas toujours vrai [59]. À cet égard, il est possible de tirer avantage de l'utilisation de différentes plateformes pour l'entraînement et l'inférence. Par exemple, un modèle de RNA pourrait être entraîné à l'aide d'un *GPU* puis transféré sur un *FPGA* pour produire des inférences. C'est ce qui est d'ailleurs fait dans la preuve de concept de la Pre Corbeil Therrien [4].

Finalement, pour son haut degré de parallélisme, sa reprogramabilité et son accessibilité, le *FPGA* est un bon candidat pour la conception d'un RNAM [4, 56, 58, 57, 60].

2.4.2 Déployer un réseau de neurones artificiels sur un *FPGA*

Dans les sections 2.3 et 2.4, il a respectivement été discuté de la théorie des RNA et des plateformes physiques permettant leur implémentation, notamment les *FPGA*. Cependant, tant à un niveau logiciel que matériel, l'adaptation d'un RNA vers un *FPGA* requiert plusieurs considérations importantes dès les premières étapes de conception d'un RNAM.

La traduction d'un RNA vers un *FPGA* nécessite plusieurs étapes et considérations supplémentaires que le RNA sur *CPU* et/ou *GPU* n'inclut pas. La représentation numérique des paramètres du RNA entres autres [61]. Ainsi, des cadres logiciels permettent de faire abstraction de la complexité d'une traduction d'un RNA vers un *FPGA* afin d'en faciliter le développement. Typiquement, les grandes étapes de ces cadres logiciels sont [56] :

1. **Définir l'architecture du RNA :** les paramètres du RNA sont définis, mis en relation et entraînés (c.-à-d. le type, le nombre de neurones, le nombre de couches cachées, les fonctions d'activation) à l'aide de logiciels haut niveau comme *TensorFlow*. Cette étape constitue normalement le développement complet d'un RNA s'il était déployé sur *CPU* et/ou *GPU*.

2. **Explorer l'espace de conception :** les ressources nécessaires et contraintes de la plateforme physique sont déterminées à partir de l'architecture et des paramètres mentionnés ci-haut.

3. **Générer les fichiers en langage de description du matériel :** le RNAM est cartographié sur la plateforme matérielle et un fichier en langage de description du matériel est généré.

4. **Tester le RNAM :** les fichiers de configuration de la plateforme sont créés et les performances du RNAM sont testées.

De plus, il est nécessaire de compresser le RNA, puisque la bande passante de la mémoire dédiée aux paramètres du RNA est l'une des principales contraintes dans le déploiement d'un RNAM [62]. On a donc avantage à compresser ces derniers. *QKeras* permet de faire cette compression directement pendant la conception du RNA [63]. Ayant déjà l'architecture de base fonctionnelle du RNA, la méthode de base pour le compresser consiste à élaguer et quantifier ses paramètres. Quant à elle, la quantification consiste à compresser les paramètres du RNA à un ensemble fini de valeurs.

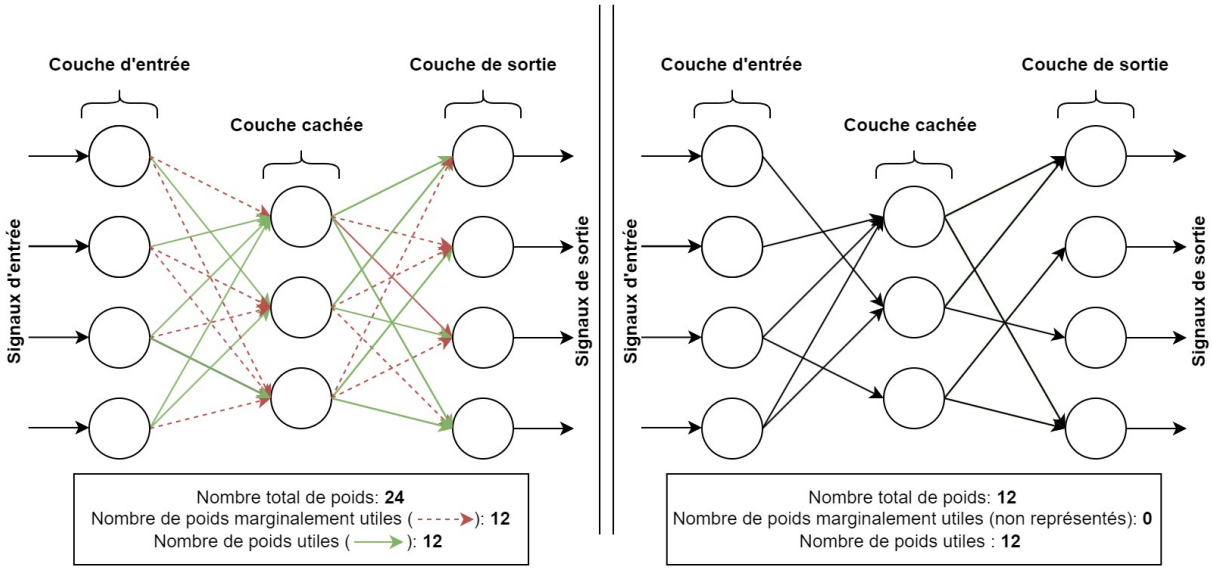

FIGURE 2.7 Visualisation de l'élagage d'un RNA ; à gauche un RNA qui contient des poids qui ne participent pas de manière significative aux prédictions (représentés en rouge) et à droite un RNA qui a été élagué. Dans cet exemple, on présume que les performances sont restées inchangées.

La figure 2.7 permet de visualiser le processus d'élagage. Cet exemple montre qu'il est possible de retirer des poids d'un RNA tout en conservant des performances de prédictions raisonnables (dans cet exemple, on présume que les performances sont restées inchangées). Il est montré qu'un élagage optimisé peut compresser un RNA d'un facteur 9 à 13 [64]. Ceci rappelle aussi que l'élagage est un processus qui doit être optimisé afin d'offrir un bon compromis entre le taux de compression du RNA et les performances de ce dernier.

Ceci dit, les cadres logiciels sont souvent développés indépendamment par différents groupes de recherche à travers le monde et leurs codes sources ou leurs détails techniques d'implémentation sont rarement disponibles avec support. Ainsi, les cadres logiciels *FINN* et *HLS4ML* ressortent comme les mieux adaptés aux présents travaux [65, 66]. Ils supportent les trois architectures de RNA présentés à la section 2.3, ils supportent les RNA compressés et leur code source est maintenu et mis à jour. Cependant, *HLS4ML* est mieux adapté

pour des applications à basse latence ce qui est un atout pour cette présente recherche [67]. *HLS4ML* est un cadre logiciel qui permet le développement haut niveau d'un RNA à l'aide d'un cadre logiciel comme *Keras* (librairie facilitant l'utilisation de *TensorFlow*), puis d'en faire la synthèse vers un langage de description du matériel pour un déploiement sur un *FPGA*. La méthode de développement est illustrée à la figure 2.8. Une fois le RNA converti en *HLS4ML*, le cadre logiciel offre plusieurs options de configurations à plusieurs niveaux différents. Par exemple, il est possible d'uniquement spécifier si on veut prioriser la latence ou l'utilisation des ressources dans le *FPGA*, mais il est aussi possible d'attribuer l'espace mémoire des paramètres individuellement.

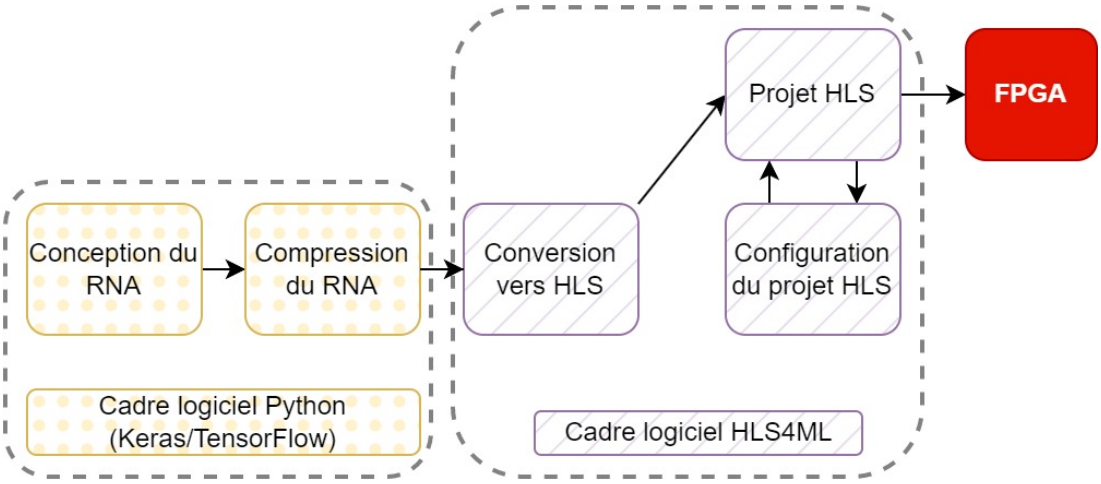

FIGURE 2.8 Méthode de développement avec *HLS4ML*. La conception et la compression du RNA est typiquement faite avec *Keras*. Le modèle est ensuite synthétisé, compilé, et exporté en fichier *.bit* pour être programmé sur le *FPGA*.

En somme, la conception d'un RNAM n'est pas une tâche triviale et des outils de développement sont nécessaires pour respecter les contraintes de planification de ce projet. Dans cette section, on a montré différents outils matériels et logiciels pour y arriver. Premièrement, le *FPGA* est la plateforme matérielle la mieux adaptée aux contextes de ces travaux, puisqu'elle est reprogrammable et rapide. Finalement, *HLS4ML* est le cadre logiciel le mieux adapté pour faire le développement d'un RNAM à basse latence sur un *FPGA*.

Dans ce chapitre d'état de l'art, la « CookieBox » a été présentée en premier lieu. Cela a défini que les techniques d'IA développées dans ces travaux devront compter le nombre de sous-impulsions dans chaque impulsion principale de rayon X capté par la « CookieBox ». En second lieu, la précision du concept d'information a permis d'identifier qu'une modélisation des signaux de la « CookieBox » doit être faite le plus tôt possible dans la chaîne de compression de données afin de diminuer au maximum les ressources computationnelles

nécessaires au traitement des données. La quantification nonuniforme optimisée est notamment ressortie comme une méthode simple et efficace pour modéliser les données. En troisième lieu, le modèle de RNA a été présenté en théorie pour ensuite faire le pont vers les différentes architectures pratiques. Finalement, on a présenté les avantages de déployer un RNA sur un *FPGA* et on a présenté l'outil logiciel *HLS4ML* pour le faire.

Le prochain chapitre présente deux articles rédigés dans le cadre de ce projet de recherche. Ensemble, ils couvrent les éléments algorithmiques et matériels permettant de répondre à la question de recherche.

CHAPITRE 3

Éléments algorithmiques permettant de répondre à la question de recherche

Avant-propos

Auteurs et affiliation :

- Berthié Gouin-Ferland : étudiant à la maîtrise, Université de Sherbrooke, Faculté de génie, Département de génie électrique et de génie informatique, Sherbrooke, Québec, Canada
- Ryan Coffee : Scientifique senior, *SLAC National Accelerator Laboratory*, *Stanford PULSE Institute*, Menlo Park, Californie, États-Unis
- Audrey C. Therrien : Professeure adjointe, Université de Sherbrooke, Faculté de génie, Département de génie électrique et de génie informatique, Sherbrooke, Québec, Canada

Date d'acceptation : 29 août 2022

État de l'acceptation : version finale publiée

Revue : *Frontiers in Physics*

Référence : B. Gouin-Ferland, R. Coffee, and A. C. Therrien, "Data reduction through optimized scalar quantization for more compact neural networks," *Frontiers in Physics*, vol. 0, p. 887, sep 2022.

Titre français : Réduction des données par une quantification scalaire optimisée pour des RNA plus compacts

Contribution au document : Cet article contribue au mémoire en montrant comment la quantification nonuniforme optimisée permet de réduire la dimensionnalité du RNA servant à compter le nombrer de sous-impulsions de rayons X.

Résumé français : La production de données brutes générées par les grandes expériences de physique dépasse désormais les taux de To/s, générant des ensembles de données intenables en très peu de temps. Ces données présentent souvent une dimensionnalité élevée tout en contenant une quantité d'informations limitée. Parallèlement, les algorithmes d'AM sont de plus en plus populaires pour le traitement et de l'analyse des données. Ces algorithmes peuvent être utilisés hors ligne pour le post-traitement et l'analyse des données, ou en ligne pour un traitement en temps réel avec une très faible latence. Les deux cas d'utilisation bénéficieraient d'une réduction du débit des données tout en préservant les

informations pertinentes : l'un en réduisant les besoins de stockage hors ligne de plusieurs ordres de grandeur et l'autre en permettant des inférences en ligne ultra rapide. De plus, diminuer la génération de données réduit également les coûts matériels et énergétiques reliés à la gestion des données. Dans ce travail, nous démontrons l'utilisation de la quantification scalaire nonuniforme optimisée pour la réduction de données à la source. Cette réduction des données permet une représentation de plus faible dimension tout en préservant l'information pertinente des données. Ceci permet la conception de RNA classificateurs de haute précision pour des inférences rapides en ligne. Nous démontrons cette approche avec une première preuve de concept ciblant la « CookieBox », un spectroscope à balayage angulaire, qui a été développé pour le *LCLS-II* comme instrument scientifique qui peut être utilisé comme un outil de diagnostic. Nous avons utilisé l'algorithme *Lloyd-Max* avec les données de la « CookieBox » pour concevoir un quantificateur scalaire nonuniforme optimisé. La quantification optimisée nous permet de réduire le volume des données d'entrée de 69 % sans impact significatif sur la précision de l'inférence. Lorsque nous tolérons une perte de 2 % sur la précision de l'inférence, nous avons obtenu une réduction de 81 % du volume d'entrée. Enfin, le passage d'une quantification des données d'entrée de 7 bits à 3 bits réduit la taille de notre RNA de 38 %.

Notes : Certaines parties de l'article répètent des éléments précédemment abordés. Cependant, la forme originale est conservée afin de ne pas nuire à la compréhension de l'article.

Data reduction through optimized scalar quantization for more compact neural networks

3.1 Introduction

Detectors for large physics and light source experiments now produce data much faster than acquisition systems can collect, triage and store it [2, 68]. The current approach of saving all raw data requires a large amount of cabling, power and downstream storage, beyond what the architecture or budget can allow [2]. Thus, several current and planned experiments would benefit from data reduction at the source. Furthermore, the initial data preparation steps before analysis tend to be very similar over time - deletion of invalid events, baseline corrections and initial information extraction, such as calculating timestamps or energy. Moving these steps at the edge –near the detector– would reduce data at the source and thus lightening the load of the high speed communication system and high-speed storage. Even so, several of this initial analysis requires complex mathematical operations which require many sequential steps, an iterative approach, and significant computational resources. This limits the capacity for true real-time data reduction [3, 7].

One strategy exploits ultra low latency Edge Machine Learning (edgeML) ; the deployment of inference models near the detector capable of real-time analysis, veto and compression of incoming data. Machine Learning (ML) models like neural networks (NN) can be trained to emulate arbitrary mathematical operations while using simpler addition and multiplication operations that can be greatly accelerated using appropriate hardware [69]. This strategy of moving much of the data preparation steps at the source enables to reduce both data velocity and data volume, resulting in resource savings in term of data transfer, processing and storage.

The LCLS-II built at SLAC National Accelerator Laboratory is capable of generating coherent x-ray shots at a 1 MHz rate [1, 2]. The experimental hutches host several dozen different instruments to capture the maximum information about each event. However, the system must be run at a lower rate to collect the data from all these instruments and send it to disk [2]. To achieve continuous full rate experiments, a first proof of concept targeting the Cookiebox detector demonstrated that deploying ML inference models on FPGA can reduce data velocity in real-time [4].

The Cookiebox is a diagnostic detector which non-destructively samples each x-ray shot to reconstruct the single shot time–energy profile via the method of attosecond angular streaking [3]. The reconstructions are to be used to select which x-ray shots fit particular experimental objectives, rejecting invalid shots, aggregating simple reference shots, or

reserving complicated shots for deeper covariance-based analysis. Such a streaming shot evaluation system significantly reduces the raw data rate from other instruments before it is written to persistent storage. However, to achieve this, each x-ray shot must be analyzed with very low latency, within about 100 µs, to avoid overly large raw data ring buffers. Such low-latency capability of edgeML has been demonstrated [4] and further work is ongoing to provide a fully working system. The Cookiebox detector produces a large volume of data, on the order of 100's of GB/s, which itself becomes a challenge when designing low complexity ML algorithms suited for limited capacity edgeML accelerators. For that reason, the compression and analysis must be distributed all along the data path, including prior to the ML algorithms. In this work, we suggest to optimally quantize the Cookiebox data before feeding it to our NN inference model. This compression strategy reduces throughput while preserving relevant information which enabled for leaner and more accurate NNs.

The Materials and Methods section begins with an overview of the CookieBox diagnostic detector, followed by a description of the quantization algorithm and the NN developed for the CookieBox. We then present the results for both the optimal quantizer model and for the ML model. Finally, we conclude with a discussion of how the quantization impacts the data and the ML model.

3.2 Materials and Methods

3.2.1 Cookiebox

The diagnostic detector that we take as our demonstration use case is an attosecond angular streaking instrument composed of an array of 16 electron time-of-flight (ToF) spectrometers, illustrated in Figure 3.1 [70]. The spectrometers are placed on a plane perpendicular to the x-ray propagation. A micrometer wavelength infra-red laser with a circular polarization modulates the central electrical field with a period of 10–30 fs [3]. A low pressure gas is present in the center chamber. When the atoms or molecules are hit by x-rays, their electrons are ejected and collected by the electron spectrometers. This instrument can measure the polarization and the time-energy spectrum of each individual x-ray shot produced by LCLS-II (diagnostic mode), or be used to measure numerous features associated with a given target atom or molecule (experimental mode).

Each spectrometer signal is fed to a 12 bit, 6.4 GS/s analog-to-digital converter. Thus, the total instrument generates data at a rate of 1.229 Tb/s. The first data reduction step consists of identifying the time at which electrons hit the spectrometer using a peak finding algorithm. The digitized signal is thus converted into timestamps corresponding to each

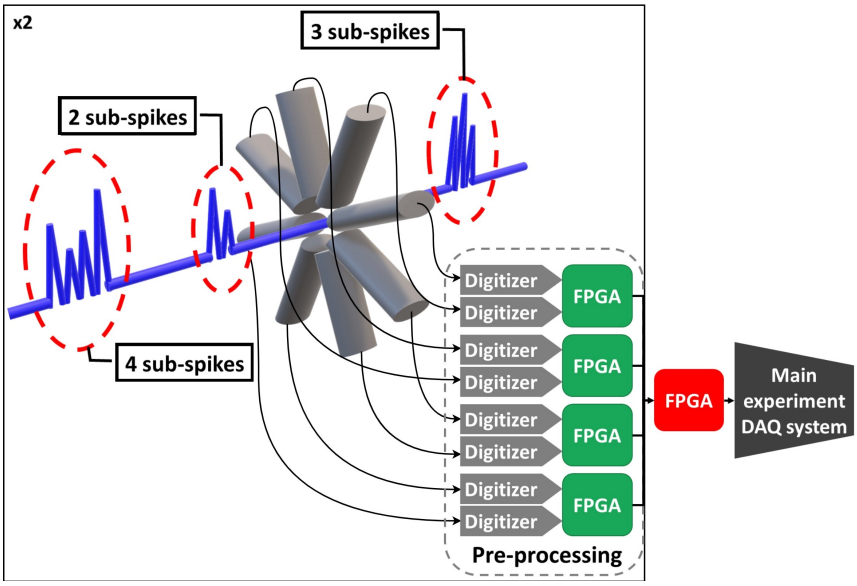

FIGURE 3.1 Diagram of the data flow of the CookieBox. The ToF spectrometers signals are processed by analog electronics before being digitized. Each 2 digitizers feed into an FPGA that will be hosting the neural network discussed in this article. The central FPGA, circled with a grey dotted line, collects the extracted features from all 8 peripheral FPGA and forwards the inference results to the rest of the DAQ system.

electron "hit". For each x-ray shot, each spectrometer collects approximately 100 hits, which are converted into 16 bit timestamps, which results in a data rate of 26 Gb/s. In experimental mode, this data is collected, however in diagnostic mode, the data must be analyzed within 100 µs to select the correct processing for each shot while avoiding large rapid memory buffers.

We choose this detector since it has recently been shown compatible with both the signal rates and the energy resolution required for the x-ray pulse reconstruction algorithms. The facility wide interest at LCLS-II has further motivated that such an instrument provide continuous streaming diagnoses of x-ray pulses, both time-energy distribution as well as polarization, at the full MHz repetition rate of the facility. As such, this instrument is now capable of working with the full data rate as soon as the LCLS-II ramps up to the highest quasi-continuous rate to provide offline diagnostic. The present work aims to move the diagnostic capability on the edge and in real-time.

3.2.2 Dataset (CookieSimSlim)

To create the initial training datasets, we use a simple simulation [9] to generate data via Monte-Carlo simulation of attosecond angular streaking. The simulation begins with a Poissonian choice of so-called self-amplified spontaneous emission (SASE) sub-spikes,

forming the x-ray shot, each with an energy consistent with the few % SASE bandwidth of the FEL process and a relative temporal delay that is chosen as an even random choice across the 2π period of the angular optical cycle.

The period of the optical cycle is chosen experimentally by the choice of dressing optical laser field and is typically in the regime of few femtoseconds total period for addressing SASE structures as targeted here [3]. The data generator also allows for sub-spike polarization variation such that our model is fully compatible with the recent developments in time-dependent polarization shaping in SASE FELs [11].

The resulting dataset from the CookieSimSlim generator is an HDF5 formatted tree of events, or "x-ray shots", each with a list of electron hit energies (Xhits) for each detector angle. This list of hit energies itself is a sampling from a smooth probability distribution Y_{pdf} that is the sum of the Gaussian energy distributions for each of the sub-spikes (offset via $\kappa \sin(\phi)$ where κ is the streaking kick strength and $\phi \in [0, 2\pi)$ is the random phase associated with the sub-spike relative arrival timing. The shot-dependent parameters such as kick strength, phase, dark-count rate, SASE width and so on are all produced as attributes of the particular shot in the HDF5 file. For convenience the output file also includes an "image" representation of the energy hit histogram X_{img}.

The hit energies in X_{hits} are represented as 32 bit floats from the generator. This bit depth is considered a "conventional" representation since in the experiment, the data will be represented as an energy conversion of an integer arrival time that is typically only of 16 bit resolution. Allowing 32 bit floating precision for the energy mapping result is therefore considered a convenient precision for sake of the arithmetic in producing that calibrated energy for each hit.

3.2.3 Quantization

All of the information on the x-ray shot is contained in the timing of the electron hits ; that is the 16 bit timestamps obtained after the pulse finding step. With CookieSimSlim, this data is provided in a 32 bit float format. We encode this large set of data to a small set of optimized values modelled on the probability distribution function (PDF) of the source detector with a nonuniform scalar quantizer [15, 14].

The mean-square error (MSE) is used to judge of the quality of the quantization. The MSE is obtained with equation 3.1 :

$$\mathrm{MSE}(Y, Q(Y)) = \frac{1}{N} \sum_{i=0}^{N-1} (y_i - Q(y_i))^2 \tag{3.1}$$

where Y is the original discrete dataset and $Q(Y)$ is the quantized dataset.

The first order entropy of the datasets is used to mesure the amount of information it contains. The first order entropy is obtained with equation 3.2 :

$$\mathrm{H}(Y) = -\sum_{i=0}^{N-1} P(y_i) \log_2 P(y_i) \tag{3.2}$$

where Y a discrete variable, that represent our dataset or quantized dataset, with possible outcomes $y_0 ... y_{N-1}$ which occur with probability $P(y_0) ... P(y_{N-1})$ [12]. The base two of the logarithm function is to gives the entropy in bits.

3.2.4 The Neural Networks

The data quantization allows for a much reduced input size for the convolutional neural network (CNN) which contributes to its size reduction. The CNN type of architecture is used to reduce the impact of the input data dimensionality on the model dimensionality itself. We also design our model to be the smallest as possible. For that, we use strategies inspired from the SqueezeNet CNN architecture [71]. However, since our CNN only has three convolution layers, our strategies boil down to using as few and as small as possible filters. We only use 3x3 filters which is the smallest kernel size to capture the notion of relative dependencies in all direction within a 2D space. We gradually double the numbers of filters between each convolution layer from the beginning to the end of the network like in the VGG model [33]. We use 10 filters in the first convolution layer allowing for below or close to 10,000 network parameters while potentiating the accuracy. The two last layers of the CNN are fully connected layer with 5 neurons each. For all layers, except the last one, the activation function is the rectified linear unit (ReLU) activation function. For the last layer, the Softmax activation function is used for classification.

A specific CNN is dedicated for each corresponding bit depth because of the model input size changes according to the number of bit used for the quantization. Except the input size, all the other configurations are the same for all CNNs. Figure 3.4 shows examples of the CNN input heatmap images in regards of the number of quantization levels. In this example, each input heatmap images (Figure 3.4-A, Figure 3.4-B and Figure 3.4-C) requires a specific CNN.

3.3 Results

Quantization Effect on Dataset

Uniform quantization allows for a quick and simple design. However, optimized nonuniform quantization minimizes the MSE, but also requires to train the quantizer beforehand. For comparison, the quantization is done using a uniform quantizer and a PDF-optimized nonuniform quantizer. In both cases, we quantized to obtain 5 strategic and realistic bit depths from 3 to 7 bits. This yields M quantization levels with $M = 2^n$ and n the bit depth.

The quantization levels for the uniform quantizer are uniformly placed within the distribution. For training the PDF-optimized nonuniform quantizer, we used the Lloyd-Max (LM) algorithm [15, 14]. The LM algorithm uses an iterative k-means clustering approach to determine which quantization level locations minimize the MSE. The initial estimate for the quantization levels are uniformly placed within the distribution. The tolerance for change in MSE after which the LM algorithm converged is set to $1e-5$. For both quantizer designs, the actual quantization is done by mapping the input value to its nearest quantization levels. Figure 3.2-A shows the original data distribution while Figure 3.2-B and Figure 3.2-C respectively show the distribution with uniform quantization and optimized nonuniform quantization.

Figure 3.3-A shows how nonuniform quantization minimized the MSE compared to a uniform quantization. In addition, Figure 3.3-B shows how nonuniform quantization also maximized the entropy. The data are quantized using the uniform and nonuniform quantizer and then converted as an heatmap with the bin intervals being the quantization levels. This result is an input heatmap image of size $16 \times$ M and it is the input of the CNN. Each pulse within an x-ray shot will create a vertical sinusoidal wave where the relative phase between waves reflects the time interval between the pulses. Figure 3.4 shows how quantizing with 5 bits over 3 bits makes the pulse count less ambiguous, but also how quantizing with 7 bits does not drastically simplify the pulse counting task (for a human eye).

3.3.1 Classification Accuracy and Model Size

We trained the CNNs to classify the pulse count in every x-ray shot event (i.e. heatmap image). A unique and dedicated CNN is trained for each bit depth and corresponding quantized heatmap image size. This is because the quantized heatmap image size determine the input size of the CNN which then impact the overall CNN dimensionality. However, all the model parameters (kernel size, number of filters...) and initial weights are kept steady for all CNNs.

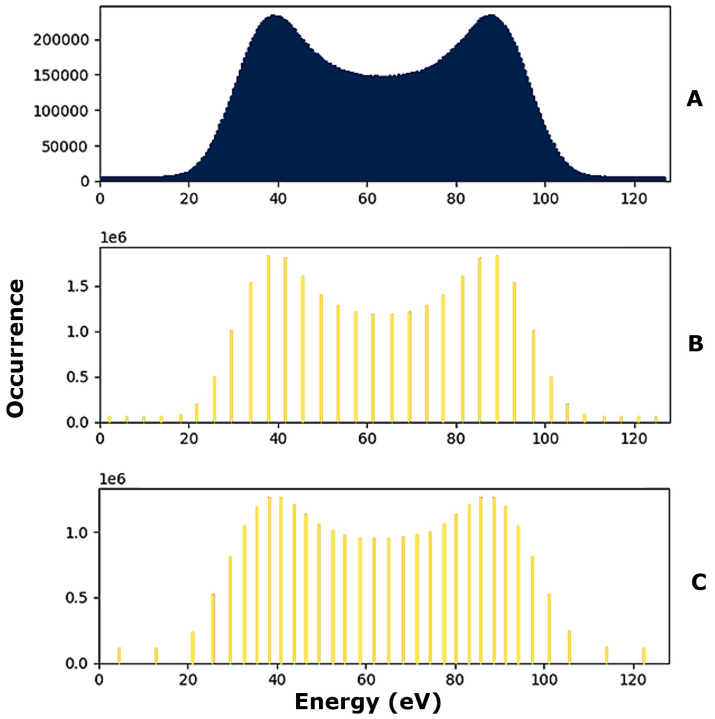

FIGURE 3.2 5-bit uniform quantization (B) and 5-bit optimized nonuniform quantization (C) effect on the original data distribution (A). The Lloyd-Max algorithm gives more granularity to the ranges where the data are more occurring within the distribution which maximizes the entropy and minimizes the mean-square error when quantizing.

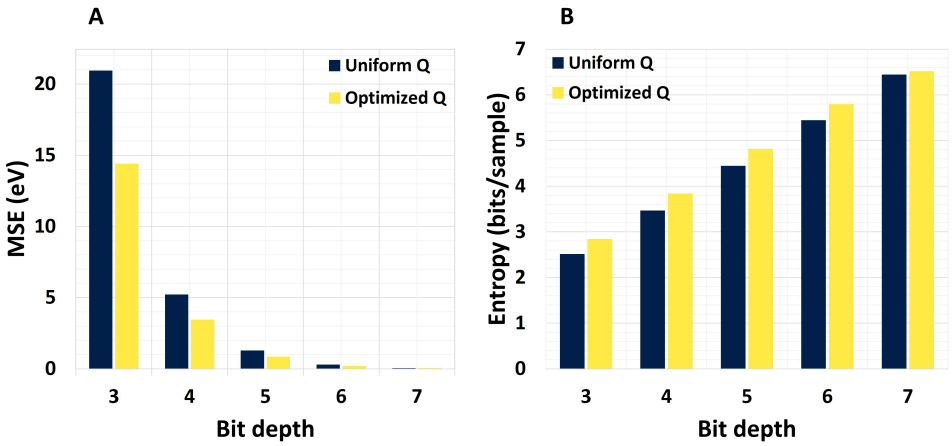

FIGURE 3.3 Mean-square error (A) and entropy (B) as a function of the quantization. We interpret low mean-square error with high entropy as a better information representation within data.

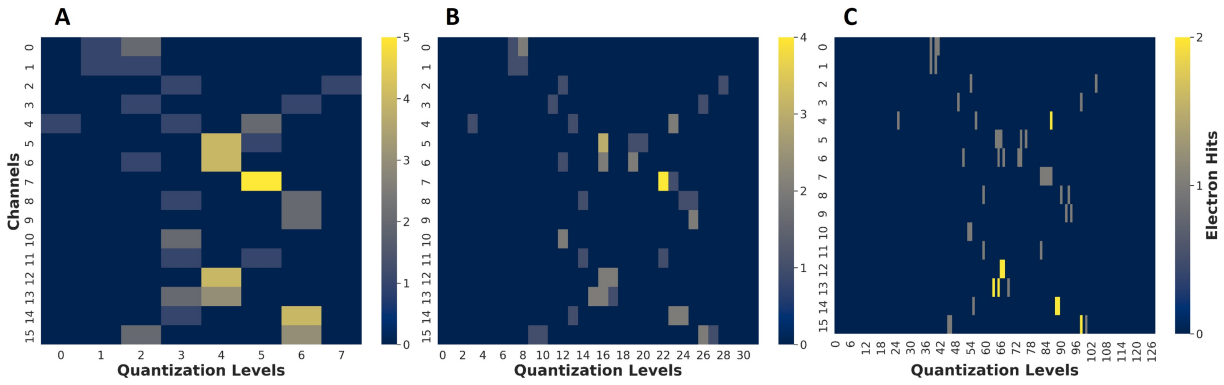

FIGURE 3.4 3, 5 and 7 bits quantized data heatmaps for a 2 pulses event (respectively A, B and C). For a human eye, the 5 bits quantized data heatmaps (B) makes the two pulses distinction less ambiguous over 3 bits quantized data heatmaps (A). However, the 7 bits quantized data heatmaps does not improve the distinction for a human eye.

The desired pulse count per x-ray shot may change between LCLS-II experiments. For that reason, we trained the CNNs on a local GPU (RTX3090) to classify 0, 1, 2, 3 and "many" pulses for every shots. The "many" class correspond to all events with 4 pulses or more, which have little value in most experiments and are generally rejected. The training setup is described in table 3.1. The training set includes 400,000 events and the test set 100,000 events. All the data are generated using the CookieSimSlim generator. Figure 3.5 shows the relation between the CNN weighted prediction accuracy as well as the CNN size when applying the method in simulation with the test set. We see that this optimized quantization scheme allows for data reduction on 5 bits while allowing for more accurate and leaner inference models then when using 7 bits. Figure 3.6 shows the confusion matrix of the 5-bit dedicated CNN in predicting the number of pulses.

TABLEAU 3.1 CNN Training Configurations

Loss Function	Sparse Categorical Crossentropy
Optimizer	Adam
Learning Rate	0.001
Batch Size	2042
Validation Split	0.2

3.4 Discussion

3.4.1 Quantization

Our goal for using optimized nonuniform quantization over uniform quantization was to maximize the information representation of the Cookiebox source on a lower bit budget. This is what Figure 3.3 suggest. We saw that while a nonuniform quantization minimi-

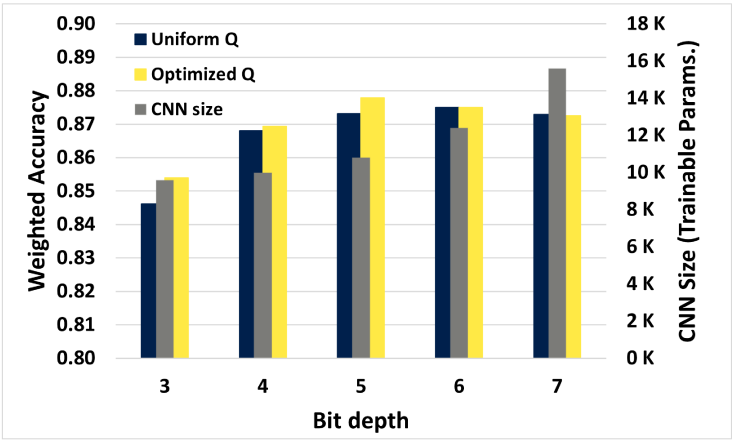

FIGURE 3.5 CNN weighted accuracy and size. Grey bars read on the right axis and the blue and yellow bars on the left axis. The model shows higher accuracy with 5-bit depth over 7-bit depth while requiring fewer model parameters.

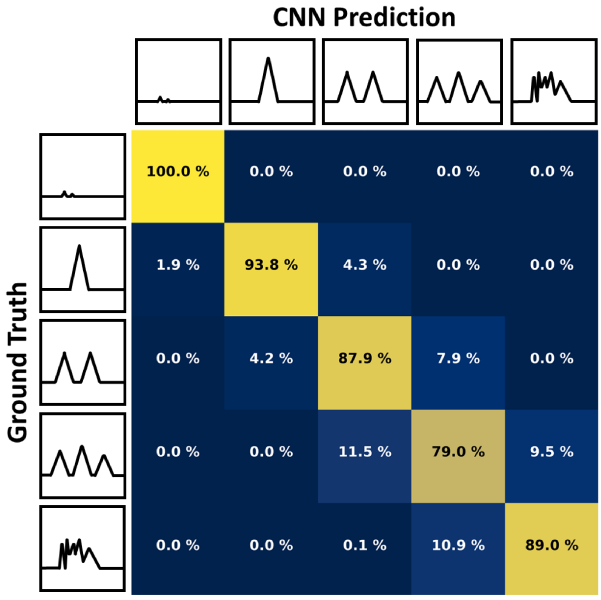

FIGURE 3.6 Confusion matrix showing the performance of the 5-bit dedicated CNN in predicting the number of pulses (0, 1, 2, 3 or "Many") in a single x-ray shot with data optimally quantized using 5 bits. A perfect CNN would have 100 % (yellow) in every box on the diagonal and 0 % (dark blue) everywhere else.

zed the MSE compare to a uniform quantization, it also maximized the entropy which represent the amount of information carried out by the data. Nonetheless, we also saw that the gap in performance tends to shrink for a larger bit depth. This is because the optimized quantization levels converge towards a uniform distribution as more levels are created within the same limited interval. We noticed that this nonuniform quantization is sensitive to changes in the data distribution ; if the source statistic changes overtime a mismatch effect could occur and change the quantizer performance. We recommend training the nonuniform quantizer on a dataset that includes those variations or to include a calibration step to train the quantizer before the data acquisition to ensure the quantizer representativity. Note that the same mismatch effect would occur to a standalone NN (i.e. without the prior optimized nonuniform quantization).

Nevertheless, quantization allows to pass from a 16-bit scalar data representation to a 7, 5 and even 3-bit representation. This yields a data reduction of 56 %, 69 % and 81 %. Even if quantization is a lossy coding, let's recall that optimized nonuniform quantization allows for data reduction while maximizing the information retention. This method avoids the computational load of lossless coding, which reduces the acquisition system latency and overall resources usage.

3.4.2 CNN Inference Model

The goal of limiting the Cookiebox data dimensionality was also to reduce the input dimensionality for our inference model. Fully connected NNs are really sensitive to their input size and tend to become very large if not contained. We used a simple CNN architecture to minimize this effect, but Figure 3.5 still shows the benefits of a small input size in term of model dimensionality. For instance, with a straightforward 16-bit representation and the same architecture, this CNN would require approximately 2.6 million parameters. By contrast, our 7, 5 and 3-bit representations shrunk the CNN size to 15,595, 10,795 and 9595 parameters respectively, a reduction of two orders of magnitude compared to the 16 bit input model. Within these smaller models, the change from a 7-bit to a 5-bit input data quantization reduces the CNN size by 31 %, with no significant impact on inference accuracy. When we tolerate a 2 % loss, the change from a 7-bit to a 3-bit input data quantization reduces the CNN size by 38 %. Note that the CNN size reduction is only due to the input dimensionality reduction and that no optimization (weight pruning, weight quantization...) is done on the model itself.

We saw small improvements in accuracy between uniform and nonuniform quantization in Figure 3.5, but as for the MSE and entropy, the difference tend to plateau beyond 6 bits. Our simulation data exhibit bimodality within the distribution for all 16 channels

dimensions and we expect a better gain of nonuniform over uniform quantization for more complex multimodal distributions. If the dimensions exhibit different distributions, we recommend to train and quantized in respect to each dimensions. With that said, our model still demonstrates better performances then the first iteration of NN that tackled the Cookiebox problem while being almost two orders of magnitude smaller [4].

The drop in accuracy from 6 and 7 bits correlates with a significant input size growth. Because the number of filters and kernel size are constant for all bit depth, it limits the learning potential of the CNN when having larger and more complex input. A solution to that is to use a bigger CNN. However, we would also need a bigger dataset to maintain the model generalization ability. Because our goal was to designed a small NN and to reduce data generation, we do not consider going bigger and deeper a viable, sustainable and elegant solution for edgeML.

Finally, we used scalar quantization as a proof of concept, but the next step is to use vector quantization with the Linde-Buzo-Gray algorithm to compress even more multidimensional data while conserving relevant information [72]. This could be even more promising for data source reduction and for smaller edgeML models.

3.5 Conclusion

Large physics experiments now produce more and more data at an ever-increasing throughput. Simultaneously, ML is becoming more popular among the community for its ability to model complex systems and the growing ML hardware accessibility. In addition to that, edgeML is a promising tool for large science experiment online data reduction. However, edgeML applications face challenges in terms of power efficiency and for hardware implementation. Moreover, some applications like the Cookiebox diagnostic detector require ultra low-latency inference.

In this work, we combined optimized data quantization with the generalization capacities of NN to reduce data source throughput while preserving relevant information and thus reducing material cost, power and data management requirements. This approach also enables smaller NN for fast real-time–on the edge–inferencing. The real-time diagnostic function would be a huge boon for upcoming LCLS-II experiments.

Beyond energy efficiency for data management system, it is worth mentioning that the cheapest data is the data which is never generated. The Jevons paradox showed us that in many technological area, increasing a process efficiency only tend to rise its absolute usage. That is already something addressed by the communication technology community

[73]. The scientific community now have a real opportunity to save in development, infrastructure and energy cost by using previously developed models directly at the source to generate as much useful information and less data.

With the development larger and faster detectors planned over the next decades in several disciplines such a medical imaging, particle physics and quantum computing, the data velocity problem will not go away. There is no universal approach ; each application presents a different set of challenges. Yet, edgeML is a powerful tool that can take advantage of the inherent structure of many data types which makes it a perfect candidate for real-time data reduction in many fields.

3.6 Acknowledgements

This research was undertaken, in part, thanks to funding from the Canada Research Chairs Program. ACT hold the Canada Research Chair in Real-Time Intelligence Embedded for High-Speed Sensors, which funded the current work, materials and personnel, including BGF. RNC thanks the Department of Energy, Office of Science, Office of Basic Energy Sciences for funding the development of the detector array itself under Grant Number FWP 100498 “Enabling long wavelength Streaking for Attosecond X-ray Science” and for funding Field Work Proposal 100643 “Actionable Information from Sensor to Data Center” for the development of CookieNetAE as well as the associated algorithmic methods, EdgeML computing hardware, and personnel. RNC also acknowledges synergistic funding for the computational method development by the Office of Fusion Energy Science under Field Work Proposal 100636 “Machine Learning for Real-time Fusion Plasma Behavior Prediction and Manipulation”.

CHAPITRE 4

Éléments matériels répondant à la question de recherche

Avant-propos

Auteurs et affiliation :

- Berthié Gouin-Ferland : étudiant à la maîtrise, Université de Sherbrooke, Faculté de génie, Département de génie électrique et de génie informatique, Sherbrooke, Québec, Canada
- Mohammad Mehdi Rahimifar : étudiant au doctorat, Université de Sherbrooke, Faculté de génie, Département de génie électrique et de génie informatique, Sherbrooke, Québec, Canada
- Charles-Étienne Granger : étudiant à la maîtrise, Université de Sherbrooke, Faculté de génie, Département de génie électrique et de génie informatique, Sherbrooke, Québec, Canada
- Quentin Wingering : étudiant à la maîtrise, Université de Sherbrooke, Faculté de génie, Département de génie électrique et de génie informatique, Sherbrooke, Québec, Canada
- Ryan Coffee : Scientifique senior, *SLAC National Accelerator Laboratory*, *Stanford PULSE Institute*, Menlo Park, Californie, États-Unis
- Audrey C. Therrien : Professeure adjointe, Université de Sherbrooke, Faculté de génie, Département de génie électrique et de génie informatique, Sherbrooke, Québec, Canada

Date d'acceptation : 12 juillet 2022

État de l'acceptation : version finale soumise

Conférence : *2022 IEEE Nuclear Science Symposium and Medical Imaging Conference (NSS/MIC)*

Référence : B. Gouin-Ferland, M. M. Rahimifar, C.-É. Granger, Q. Wingering, R. Coffee, et A. C. Therrien, "Combining Optimized Quantization and Machine Learning for Real-Time Data Reduction at the Edge," *2022 IEEE Nuclear Science Symposium and Medical Imaging Conference (NSS/MIC)*, [version finale soumise]

Titre français : Combinaison de la quantification optimisée et de l'AM pour la réduction de données en temps réel en périphérie des détecteurs

Contribution au document : Cette présentation contribue au mémoire en montrant que la combinaison de la quantification nonuniforme optimisée et des RNA permet d'atteindre une latence de moins de 100 µs par inférence.

Résumé français : De plus en plus d'expériences de physique des rayonnements sont appelées à dépasser des débits de données de l'ordre du To/s. Ces débits sont insoutenables pour les systèmes d'acquisition, de transfert et de stockage des données. Pour maintenir les coûts et les besoins en énergie des installations de stockage à des niveaux acceptables, la génération de données doit être réduite à la source : en périphérie du détecteur. Toutes les données ne sont pas équivalentes ; ce qui a vraiment de la valeur, ce sont les informations qu'elles contiennent. Les modèles d'inférence d'AMP sur *FPGA* peuvent extraire ces informations utiles des données brutes, catégoriser les événements et appliquer des actions avec une très faible latence. De plus, la quantification nonuniforme optimisée peut être utilisée encore plus près de la source – directement au niveau des numériseurs – pour maximiser la représentation de l'information utile. Cela permet d'utiliser des modèles d'AMP plus petits et plus fiables. La combinaison d'une quantification nonuniforme optimisée avec des modèles d'inférence d'AM sur *FPGA* réduit la génération de données directement à la source et réduit ainsi la complexité de l'acquisition, du transfert et du stockage des données de plusieurs ordres de grandeur. Nous démontrons cette approche en combinant deux preuves de concept antérieures ciblant la « CookieBox », un spectroscope à balayage angulaire développé pour le LCLS-II. Les données sont d'abord quantifiées de manière optimale avant d'être transmises à un modèle d'inférence parallèle et prêt à être utilisé en pipeline pour extraire les informations souhaitées. Avec cette méthode, nous avons mesuré directement sur *FPGA* une latence d'inférence de 5.78 µs avec une baisse marginale de la précision par rapport aux simulations. Si nous tolérons une autre baisse de précision de 3 %, nous atteignons une latence d'inférence de 1.93 µs.

Notes : Certaines parties de l'article répètent des éléments précédemment abordés. Cependant, la forme originale est conservée afin de ne pas nuire à la compréhension de l'article.

Combining Optimized Quantization and Machine Learning for Real-Time Data Reduction at the Edge

4.1 Introduction

The increasingly high data velocity in physics and light source experiments limits the rate of new scientific discoveries. The SLAC National Accelerator Laboratory just brought online the second iteration of the LINAC Coherent Light Source (LCLS) : the LCLS-II. The LCLS-II advancements, combined with more ambitious experiments and better performing detectors, will lead to a data generation surpassing TB/s rates [2]. The trivial use of larger acquisition systems does not scale with a pragmatic computational usage. To optimize the use of collected data we need more efficient solutions in terms of power and cost. Thus, we propose state-of-the-art optimized quantization and modern Edge Machine Learning (edgeML) techniques, both combined on FPGA, to achieve real time data reduction directly at the source.

A first proof of concept demonstrated the approach of deploying a neural network (NN) on FPGA for ultra low latency event discrimination [4]. Further work demonstrated the use of nonuniform quantization for leaner and more accurate NN [74]. In this work, we combine those two studies to offer a real-time data processing scheme on FPGAs to intelligently reduce data velocity at the source with event discrimination. The next section will give an overview of theoretical background, followed by the deployment details for combining the two methods on FPGA, before discussing the results.

4.2 Theoretical Background

4.2.1 The CookieBox Detector

The Cookiebox diagnostic detector is an angular array of electron spectrometers detector which non-destructively samples x-ray shots at the LCLS-II rate of 1 MHz to measures its position, polarity, energy spectrum and relative time structure [3]. The x-ray shot ionizes a low density gas in the Cookiebox interaction point, leaving the detector virtually unaffected to interact with the sample under study situated in the main experiment downstream. Figure 4.1 shows this setup. A circular array of 16 drift tubes placed around the interaction point collect the photoelectrons and funnel them to corresponding microchannel plates (MCPs). The MCPs amplify the photoelectron signals before sending them to 6.4 GS/s digitizers. From that digital signal, a peak-finding algorithm determines each

photoelectron Time-of-Flight (TOF). The TOF is then used to build the photoelectron energy spectrum. Figure 4.1 shows an overview of this acquisition chain.

The addition of a circularly polarized optical laser in the interaction point introduces an acceleration bias to the photoelectrons in the direction of the vector potential. This acceleration results in a shorter photoelectron TOF which then shifts the photoelectron energy spectrum profile. Knowing the polarization period of the optical laser and by correlating it with the energy shift across all TOF detectors, we can distinguish sub-spikes in a single shot with attosecond precision [3, 7]. Figure 4.1 clarifies sub-spikes from shots by showing three shots with a different number of sub-spikes in each. The next section describes how characterizing the sub-spikes leads to data reduction.

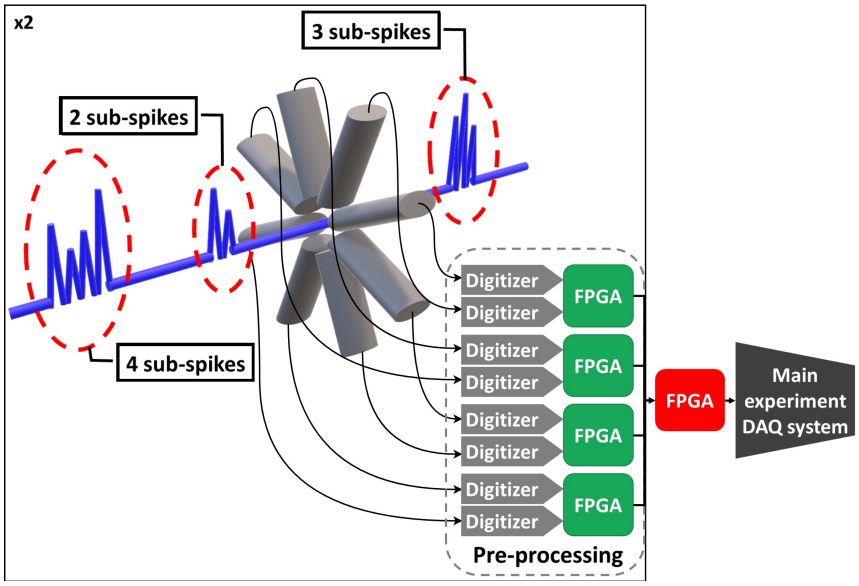

FIGURE 4.1 Diagram of the data flow of the CookieBox. The x-ray beam is represented by the blue line. The TOF spectrometers signals are processed by analog electronics before being digitized. Each 2 digitizers feed into an FPGA which computes the TOF and converts into the energy spectrum domain. The central FPGA, in red, hosts the quantizer and neural network. It collects the spectrum from all 8 peripheral FPGA and forwards the computed number of sub-spikes to the rest of the DAQ system.

4.2.2 EdgeML for Real Time Data Reduction

Specific LCLS-II experiments require specific x-ray beam shot characteristics. Counting the number of sub-spikes is one way to characterize x-ray beam shots. Moreover, the x-ray shot generation process is fundamentally stochastic, which means the number of sub-spikes in a shot varies in an unpredictable way [7, 3]. Sub-spike counting enables event discrimination which in turn leads to data reduction from the downstream main experiment by

discarding the undesired shots. In this work we call this event discrimination a "veto". The CookieBox produces 200 GB/s of data. For the veto to be of any value to the downstream main experiment, it must be determined in less than 100 μs to limit buffer size.

Classical methods for shot reconstruction requires several iterations, making them ineligible for real-time implementation. To overcome this challenge, we train a NN to extract relevant features, such as the number of sub-spikes in each x-ray shot, using the shot energy spectrum [4]. EdgeML is a technique to process data on the fly by placing machine learning (ML) inference engine next to the detectors. The first iteration of EdgeML for the Cookiebox counted the number of sub-spikes within x-ray shots with a NN on FPGA with total inference time of 19.3 μs [4]. FPGAs are customizable devices which are growing in popularity for edgeML applications due to their inherent parallelism, low power usage and large number of I/O ports for high-throughput communication. This makes FPGAs well suited for high rate edgeML application such as ours [75].

In this work, we also optimally quantize the x-ray shot energy spectrum. Quantization is the process of mapping a large set of values to a smaller set of determined ones (i.e. quantized values). Uniform quantization is often chosen for its design simplicity. However, PDF-optimized nonuniform quantization are trained to minimizes the mean-square quantization error (MSQE). Thus, optimally quantizing the x-ray shot energy spectrum allows to reduce the number of energy bins in the spectrum while preserving its relevant information as shown in previous work [74]. The next section describe the material and methods we used to combined optimized nonuniform quantization and NN inference model both on FPGA.

4.3 Material and methods

As raw Cookiebox signals are yet to be available, we generate a 500,000 shot event dataset via a Monte Carlo simulation based on the physics of attosecond angular streaking [3, 74]. 400,000 shot events are reserved to train the quantizer and the NNs and 100,000 shot events are used for testing. Each shot is labelled with its respective class representing the number of sub-spikes. These classes are : 0, 1, 2, 3 and "many". The "many" class corresponds to all shot events with four sub-spikes or more. The raw test data from the Monte Carlo simulation are streamed to the FPGA where they are quantized in respect to the pre-trained quantizer and converted into a heatmap.

We designed three quantizers : a 3-bits, a 5-bits and a 7-bits quantizer. We trained each bit depth specific quantizer offline on CPU with the simulation training set in respect to the desired number of quantized energy levels within the spectrum. We trained the quantizer using the Lloyd-Max (LM) algorithm which iteratively finds the optimal quantized energy levels within the spectrum in terms of MSQE [74, 15, 14]. The quantized spectrum are then converted into a per shot heatmap, as shown in Figure 4.2. Each heatmap contains the energy spectrum from the 16 TOF detectors as one finite size image. This 16×2^n heatmap is the input image for the NNs. 16 is the number of TOF detectors, 2^n is the number of quantized energy levels within the spectrum and n is the bit depth of the quantizer. This heatmap image is the input of all NNs.

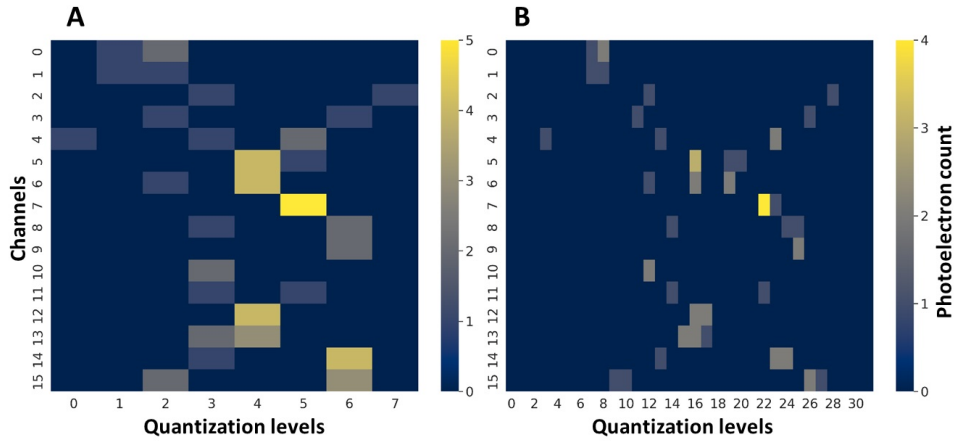

FIGURE 4.2 3 and 5 bits quantized data heatmaps example for a 2 sub-spikes event (respectively A and B). For a human eye, the 5 bits quantized data heatmaps (B) makes the two pulses distinction less ambiguous than 3 bits quantized data heatmaps (A). However, the 3 bits quantized data heatmaps (A) is four times smaller in term of number of pixels compare to the 5 bits quantized data heatmaps (B).

We designed two types of NNs for comparison : a fully connected neural network (FCNN) and a convolutional neural network (CNN) architectures. The FCNN has a flatten layer at its input, followed by four densely connected layers with respectively 4, 32, 32 and 5 neurons each. The CNN model has three convolution layers interleaved with maxpooling layers. The two last layers of the CNN are densely connected layer with five neurons each. For both architectures, the last layer uses a Softmax activation function for classification while the rest of the network uses rectified linear unit activation function (RELU). Except for the input size, all the other configurations are identical for all FCNNs and CNNs. The input shape depends of the number of energy bins in the quantized x-ray shot energy spectrum. A specific pair of FCNN and CNN is dedicated for each corresponding bit depth

because of the NNs input size changes according to the number of quantized energy levels. We trained the networks offline on GPU (RTX 3090) using qkeras [63]. For better FPGA resources usage, all network weights, bias and RELU activation functions are quantized as 8 bits integers. This is the NN quantization itself and it is not to be confused with the nonuniform quantization scheme presented in this work. During training, we also pruned the NN at 50 % which means that 50 % of the weights of the NN are forced to 0 to lower the NN footprint, computational complexity. Qkeras optimizes the pruning to minimizes the drop in accuracy the process might induce.

We implemented the trained quantizers on FPGA using Vivado High-Level-Synthesis (HLS) and deployed the trained NNs using the High-Level-Synthesis for Machine Learning (HLS4ML) python framework [65]. The device is a Zynq UltraScale+ MPSoC ZCU102 Evaluation Kit running at 100 MHz. For all NNs, we computed the latency using a clock counter on the *FPGA*. For the quantizers, the latency is obtained from cosimulations in Vivado HLS. For all NNs and quantizers, the resources usage is obtained from cosimulations in Vivado HLS.

4.4 Results

We first trained the quantizer. The tolerance for change in MSQE after which the LM algorithm converged is set to 10^{-5}. The initial estimate for the quantization levels are uniformly placed within the distribution. Table 4.1 shows the MSQE in respect to each bit depth obtained from the test set.

Tableau 4.1 Quantization Performances

Bit depth	**3**	**5**	**7**
MSQE (eV)	14.430	0.884	0.074
Latency (μs)	0.05	0.17	0.65
BRAM (%)	0	0	0
DSP (%)	0	0	0
FF (%)	∼0	∼0	1
LUT (%)	∼0	3	33
URAM (%)	0	0	0

We then trained the respective FCNN and CNN using the heatmap images to classify the number of sub-spikes. The training configurations for all NNs are presented in Table 4.2. For each NNs, we also measured the classification accuracy and the size the networks (i.e. number of parameters). Table 4.3 and table 4.4 show these results. The latency and the resources usage for all quantizers and NNs are also presented in Tables 4.1-4.4 Table 4.3 and Table 4.4.

TABLEAU 4.2 Neural Networks Training Configurations

	FCNN	**CNN**
Loss Function	SCC*	SCC*
Optimizer	Adam	Adam
Pruning (%)	50	50
Learning Rate	0.001	0.001
Batch Size	256	2042
Validation Split	0.2	0.2

*Sparse Categorical Crossentropy (SCC)

Table 4.1 shows that the MSQE decreases rapidly as we increase the bit depth. Low MSQE is generally synonym of better and less noisy quantized signals. On the other hand, table 4.4 shows the CNN accuracy dropping from 5 to 7-bits quantization. This drop in accuracy correlates with a significant input size growth ; from a 16×32 to a 16×128 input size. This limits the learning potential of the network since the input is larger and more complex, but the number of filters and kernel size are constant. A straightforward solution could be to use a bigger network. However, this larger CNN would be slower and would also require more training data to prevent overfitting. For these reasons, we do not consider going bigger and deeper a viable, sustainable and elegant solution for edgeML.

TABLEAU 4.3 Fully Connected Neural Network Performances

Bit depth	**3**	**5**	**7**
Accuracy (%)	81	84	N/A
Size (Params.)	1897	3433	9577
Latency (μs)	1.93	5.78	N/A
BRAM (%)	∼0	∼0	N/A
DSP (%)	∼0	∼0	N/A
FF (%)	1	2	N/A
LUT (%)	18	32	N/A
URAM (%)	0	0	N/A

TABLEAU 4.4 Convolutional Neural Network Performances

Bit depth	**3**	**5**	**7**
Accuracy (%)	82	86	84
Size (Params.)	3065	3665	6065
Latency (μs)	11.47	21.14	91.07
BRAM (%)	2	5	10
DSP (%)	8	13	17
FF (%)	5	7	8
LUT (%)	24	37	52
URAM (%)	0	0	0

Table 4.3 and Table 4.4 show that the FCNN is faster and uses less resources than the CNN even for similar size networks. Nevertheless, the FCNN architecture can grow very

large if its input size also grows large. Note that the results for the FCNN with 7-bits bit depth are not available because of a software limitation in HLS4ML. The framework limits the number of parameters per layer at 4096. We exceed that limit when using a 16×128 flattened image input size.

Finally, the usage of NNs for ultra-low latency edgeML applications requires trade-offs between the NN size and the latency. FCNNs are faster than CNNs, but are more sensitive to a large input dimensionality compare to CNNs. Thus, for the use of NNs in ultra-low latency edgeML applications, the problem should be reduced to the lowest possible dimensionality (with known feature extraction by example) before feeding it to a FCNN. If the dimensionality is still too large, a CNN may be a better choice. In this work we optimized the problem dimensionality by using optimized nonuniform quantization.

4.5 Conclusion

In this work, we implemented optimized data quantization and deployed NNs, both on FPGA, for real time edgeML. We used optimized quantization to maximize the information representation with lower dimensionality. The resulting NN are smaller, thus faster. This scheme allows for ultra-low latency event discrimination which leads to real time data reduction from the LCLS-II experiments. Finally we showed that edgeML is a powerful tool for extracting relevant information at the source for the large physics and light source experiments. This information extraction lowers the data acquisition complexity and overall bandwidth, power and storage requirements.

4.6 Acknowledgment

This research was undertaken, in part, thanks to funding from the Canada Research Chairs Program and the Mitacs Globalink Research Award. Corbeil Therrien holds the Canada Research Chair in Real-Time Intelligence Embedded for High-Speed Sensors, which funded the current work, materials and personnel, including Gouin-Ferland. Additional funding for Gouin-Ferland and funding for Coffee was provided by the Department of Energy, Office of Science, Office of Basic Energy Sciences under Field Work Proposal 100643 “Actionable Information from Sensor to Data Center.” The development of the algorithm for simulation was funded by the Department of Energy, Office of Science, Office of Basic Energy Sciences under Field Work Proposal 100498 “Enabling long wavelength Streaking for Attosecond X-ray Science”. We also thank James J. Russell. and Abhilasha Dave for their contribution to the deployment of this method at SLAC National Accelerator Laboratory.

CHAPITRE 5

CONCLUSION

5.1 Sommaire de la recherche

Ces travaux montrent que la combinaison de la quantification nonuniforme optimisée et des RNA est une solution pour caractériser les impulsions de rayons X du LCLS-II en moins de 100 µs tout en minimisant la puissance consommée. La quantification des données spectrales de la « CookieBox » permet de réduire la taille du spectre. L'optimisation de la quantification permet de maximiser la représentation de l'information à travers ces données. C'est cette réduction de dimensionnalité et maximisation de l'information qui permet l'utilisation de petits RNA pour compter les sous-impulsions de rayons X. Le problème (le compte des sous-impulsions) est plus simple à résoudre et ce qui évite d'utiliser des algorithmes d'AP qui requièrent plus de ressources computationnelles en plus de complexifier la méthodologie d'acquisition. À l'inverse, les RNA développés dans ces travaux sont très petits, ce qui les rend très rapides.

Plus en détail, on quantifie optimalement le spectre en énergie du rayon X pour modéliser les données et en diminuer la dimensionnalité. Le RNA utilise ce spectre de taille réduite pour compter le nombre de sous-impulsions dans une impulsion principale de rayons X. Premièrement, les résultats montrent que le RNA performe marginalement mieux lorsque les données sont optimalement quantifiées plutôt qu'uniformément quantifiées. C'est ce qui est montré à la figure 3.5. Ceci dit, ces résultats ont été obtenus à l'aide de données simulées. L'écart de complexité entre les données simulées et les vraies données de la « CookieBox » est assez grand pour motiver l'utilisation de la quantification nonuniforme optimisée plutôt que la quantification uniforme seulement. Deuxièmement, la réduction de dimensionnalité du spectre en énergie a bien pour effet de diminuer la taille du RNA comment montré à la figure 3.5. Puisque c'est le spectre qui est présenté à l'entrée du RNA, la réduction de sa taille cause directement la réduction de la première couche du RNA. Ceci a pour effet de diminuer la taille globale du RNA. Finalement, lorsque déployé sur un *FPGA*, le RNA comptes le nombre de sous-impulsions en $\sim$2 µs avec une précision de $\sim$ 85 % comme montré aux tableaux 4.3 et 4.4. La latence de la quantification se fait sous la centaine de nanosecondes comme montré au tableau 4.1.

Ceci répond à la question de recherche ; la combinaison de la quantification nonuniforme optimisée et des RNA est implémentable sur un *FPGA* et offre des résultats d'inférence fiables, en moins de 100 µs tout en minimisant la puissance consommée.

5.2 Retour sur les contributions originales

Pendant ma maîtrise, j'ai contribué de manière significative au domaine de l'IA embarquée en utilisant une approche combinant la quantification nonuniforme optimisée et les RNA sur *FPGA*. Mon travail a consisté à démontrer que la quantification optimisée des données obtenues à partir de détecteurs scientifiques permettait une réduction importante de leur taille tout en conservant les informations utiles. Cette technique de quantification optimisée des données a simplifié la tâche des RNA pour cette application, leur permettant ainsi d'être plus petits, plus rapides et plus économes en énergie. En conséquence, elle a rendu possible le traitement en temps réel des données pendant leur acquisition, ce qui n'était pas possible auparavant avec les systèmes traditionnels.

J'ai mis au point un système qui est maintenant en phase de déploiement et qui est capable d'effectuer des inférences sur un *FPGA* en moins de 2 µs. Cette conception répond aux exigences en temps réel des applications embarquées en offrant une solution innovante basée sur les fondements théoriques de l'information et des techniques de pointe d'IA.

J'ai participé à la mise en place d'un environnement de développement pour les RNA sur *FPGA* ce qui a été bénéfique pour notre groupe de recherche ainsi que pour le *SLAC National Accelerator Laboratory*. Ces efforts ont pavé la voie pour des travaux futurs dans ce domaine.

Mon travail a permis d'établir de bonnes pratiques pour la conception et la mise en œuvre de solutions d'IA embarquées en science. Les travaux futurs pourront capitaliser sur les avantages techniques et méthodologiques que j'ai mis en place pour améliorer encore plus la performance et la flexibilité des systèmes d'AMP.

Ces travaux permettront à la communauté scientifique d'acquérir plus rapidement des données de meilleure qualité, accélérant ainsi la recherche. Le détecteur visé bénéficiera directement de ces avancées, améliorant la recherche fondamentale en structure des matériaux et des molécules biologiques. De plus, les outils de compression développés pourront être transférables à d'autres applications, telles que l'imagerie médicale et l'informatique quantique.

5.3 Nouvelles perspectives de recherche

Malgré les résultats prometteurs obtenus dans ces travaux, il reste encore plusieurs défis à relever pour améliorer l'utilisation des systèmes *FPGA* pour la quantification et le RNA des impulsions de rayons X.

Tout d'abord, il est important de souligner que la combinaison de la quantification et du RNA sur un système *FPGA* reste à finaliser. Bien que les deux méthodes aient été démontrées indépendamment, un système *FPGA* complet qui combine ces deux approches reste à déployer. Cependant, des efforts sont actuellement en cours pour intégrer ces deux méthodes et déployer le banc d'essai combiné.

De plus, à l'heure actuelle, il n'existe pas de base de données étiquetée des impulsions de rayons X enregistrées par la « CookieBox ». De ce fait, l'équipe du Dr Ryan Coffee travaille activement sur deux méthodes pour pallier à cette situation. La première en développant un système de simulation complet du spectre en énergie, qui comprend notamment le spectre d'émission des électrons diffus et l'effet Auger. La seconde méthode est d'utiliser un détecteur substitue et complémentaire à la « CookieBox » afin de générer des bases de données réelles et étiquetées. Ces méthodes amélioront la précision des systèmes *FPGA* en fournissant des données plus représentative de la réalité.

Aussi, le traitement neuromorphique asynchrone des données événementielles est un domaine de recherche en plein essor. Les données événementielles d'origine temporelle sont courantes dans de nombreuses disciplines scientifiques, telles que la physique des particules et l'imagerie médicale, et même la physique quantique et l'astrophysique. Bien que les résultats liés à cet objectif n'aient pas été suffisamment approfondis pour être intégrés à ce mémoire, il est important de souligner son potentiel prometteur pour améliorer les performances des systèmes d'AMP pour les détecteurs scientifiques. Les avantages de cette approche incluent son efficacité énergétique, ainsi que sa capacité à résoudre des problèmes complexes. Pour soutenir l'utilisation de cette technologie, il serait important de rendre disponible une plateforme de développement qui comprendrait un traitement neuromorphique reprogrammable, ainsi que des modules d'interfaçage standard. Un tel outil permettrait de tester des configurations neuromorphiques physiques dans divers cas d'utilisation concrets, permettant ainsi de tirer parti du potentiel de l'IA neuromorphique pour la communauté scientifique.

LISTE DES RÉFÉRENCES

[1] P. Abbamonte *et al.*, "New science opportunities enabled by LCLS-II X-ray lasers," 6 2015.

[2] J. B. Thayer, G. Carini, W. Kroeger, C. O'Grady, A. Perazzo, M. Shankar, and M. Weaver, "Building a Data System for LCLS-II," in *Proceedings of the IEEE Nuclear Science Symposium and Medical Imaging Conference*, 2017, pp. 1–4.

[3] N. Hartmann, G. Hartmann, R. Heider, M. S. Wagner, M. Ilchen, J. Buck, A. Lindahl, C. Benko, C. Bostedt, J. Gruenert, J. Krzywinski, J. Liu, A. Lutman, A. Marinelli, T. Maxwell, A. Miahnahri, S. Moeller, M. Planas, J. Robinson, J. Viefhaus, T. Feurer, R. Kienberger, R. N. Coffee, and W. Helml, "Attosecond time–energy structure of X-ray free-electron laser pulses," *Nat. Photon.*, vol. 12, p. 215, 2018.

[4] A. C. Therrien, R. Herbst, O. Quijano, A. Gatton, and R. Coffee, "Machine Learning at the Edge for Ultra High Rate Detectors," in *Proceedings of the IEEE Nuclear Science Symposium and Medical Imaging Conference*, 2019, pp. 1–4.

[5] E. Allaria *et al.*, "Control of the polarization of a vacuum-ultraviolet, high-gain, free-electron laser," *Phys. Rev. X*, vol. 4, p. 041040, Dec 2014.

[6] A. A. Lutman *et al.*, "Polarization control in an x-ray free-electron laser," *Nature Photonics*, vol. 10, no. 7, pp. 468–472, Jul 2016.

[7] S. Li, Z. Guo, R. N. Coffee, K. Hegazy, Z. Huang, A. Natan, T. Osipov, D. Ray, A. Marinelli, and J. P. Cryan, "Characterizing isolated attosecond pulses with angular streaking," *Optics Express*, vol. 26, p. 4531, 2 2018.

[8] S. V. Milton *et al.*, "Exponential gain and saturation of a self-amplified spontaneous emission free-electron Laser," *Science*, vol. 292, no. 5524, pp. 2037–2041, jun 2001.

[9] R. Coffee. (2022) CookieSimSlim : A simple simulation and data generator that approximates attosecond X-ray angular streaking results for LCSL-II algorithm development. Accessed at 2023-02-16. [Online]. Available : https ://github.com/ryancoffee/CookieSimSlim.git

[10] D. P. Kroese, T. Brereton, T. Taimre, and Z. I. Botev, "Why the Monte Carlo method is so important today," *WIREs Computational Statistics*, vol. 6, no. 6, pp. 386–392, 2014.

[11] N. Sudar, R. Coffee, and E. Hemsing, "Coherent X-rays with tunable time-dependent polarization," *Phys. Rev. Accel. Beams*, vol. 23, p. 120701, Dec 2020.

[12] K. Sayood, *Introduction to Data Compression, Fourth Edition*, 4th ed. San Francisco, CA, USA : Morgan Kaufmann Publishers Inc., 2012.

[13] C. E. Shannon, "A mathematical theory of communication," *The Bell System Technical Journal*, vol. 27, no. 3, pp. 379–423, 1948.

[14] J. Max, "Quantizing for minimum distortion," *IRE Transactions on Information Theory*, vol. 6, no. 1, pp. 7–12, 1960.

[15] S. Lloyd, "Least squares quantization in PCM," *IEEE Transactions on Information Theory*, vol. 28, no. 2, pp. 129–137, 1982.

[16] S. Russell, *Artificial intelligence : a modern approach*. Hoboken, NJ : Pearson, 2021.

[17] M. Negnevitsky, *Artificial intelligence : A guide to intelligent systems*. Harlow, England New York : Addison Wesley/Pearson, 2011.

[18] G. M. Shepherd, *The synaptic organization of the brain*. Oxford University Press, 01 2004.

[19] F. Rosenblatt, "The perceptron - A perceiving and recognizing automaton," Cornell Aeronautical Laboratory, Ithaca, New York, Tech. Rep. 85-460-1, January 1957.

[20] S. Russell, *Artificial intelligence : a modern approach*. Upper Saddle River, New Jersey : Prentice Hall, 2010.

[21] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, no. 6088, pp. 533–536, Oct 1986.

[22] Y. Bengio, P. Frasconi, and P. Simard, "The problem of learning long-term dependencies in recurrent networks," in *Proceedings of the IEEE International Conference on Neural Networks*, 1993, pp. 1183–1188 vol.3.

[23] R. Pascanu, T. Mikolov, and Y. Bengio, "On the difficulty of training recurrent neural networks," in *Proceedings of the International Conference on International Conference on Machine Learning*, 2013, p. 1310–1318.

[24] S. Yang, X. Yu, and Y. Zhou, "LSTM and GRU neural network performance comparison study : Taking yelp review dataset as an example," in *International Workshop on Electronic Communication and Artificial Intelligence*, 2020, pp. 98–101.

[25] L. Li, K. Jamieson, A. Rostamizadeh, and A. Talwalkar, "Hyperband : a novel bandit-based approach to hyperparameter optimization," Tech. Rep., 2018.

[26] Z. Zhang, "Improved adam optimizer for deep neural networks," in *Proceedings of the IEEE/ACM International Symposium on Quality of Service*. Institute of Electrical and Electronics Engineers Inc., jan 2019, pp. 1–2.

[27] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, "ImageNet large scale visual recognition challenge," pp. 211–252, dec 2015, ArXiv :1409.0575.

[28] Z. Lu, H. Pu, F. Wang, Z. Hu, and L. Wang, "The Expressive Power of Neural Networks : A View from the Width," in *Advances in Neural Information Processing Systems*, vol. 30. Curran Associates, Inc., 2017.

[29] G. Cybenko, "Approximation by superpositions of a sigmoidal function," *Mathematics of Control, Signals and Systems*, vol. 2, no. 4, pp. 303–314, Dec 1989.

[30] P. Thanapol, K. Lavangnananda, P. Bouvry, F. Pinel, and F. Leprévost, "Reducing overfitting and improving generalization in training convolutional neural network (CNN) under limited sample sizes in image recognition," in *Proceedings of the International Conference on Information Technology*, 2020, pp. 300–305.

[31] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems*, vol. 25. Curran Associates, Inc., 2012, pp. 84–90.

[32] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," 2013, ArXiv :1311.2901.

[33] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014, ArXiv :1409.1556.

[34] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 1–9.

[35] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 2818–2826.

[36] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. A. Alemi, "Inception-v4, Inception-ResNet and the impact of residual connections on learning," in *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*. AAAI Press, 2017, p. 4278–4284.

[37] F. Wang, M. Jiang, C. Qian, S. Yang, C. Li, H. Zhang, X. Wang, and X. Tang, "Residual attention network for image classification," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 6450–6458.

[38] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He, "Aggregated residual transformations for deep neural networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 5987–5995.

[39] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018, pp. 7132–7141.

[40] X. Zhang, Z. Li, C. C. Loy, and D. Lin, "PolyNet : A pursuit of structural diversity in very deep networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 3900–3908.

[41] D. Han, J. Kim, and J. Kim, "Deep pyramidal residual networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 6307–6315.

[42] S. Woo, J. Park, J.-Y. Lee, and I. S. Kweon, "CBAM : Convolutional block attention module," in *Computer Vision – ECCV*. Cham : Springer International Publishing, 2018, pp. 3–19.

[43] A. Khan, A. Sohail, U. Zahoora, and A. S. Qureshi, "A survey of the recent architectures of deep convolutional neural networks," *Artificial Intelligence Review*, vol. 53, no. 8, pp. 5455–5516, Dec 2020.

[44] V. Sze, Y. H. Chen, T. J. Yang, and J. S. Emer, "Efficient processing of deep neural networks : A tutorial and survey," *Proceedings of the IEEE*, vol. 105, no. 12, pp. 2295–2329, dec 2017.

[45] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 770–778.

[46] F. N. Iandola, S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally, and K. Keutzer, "SqueezeNet : AlexNet-level accuracy with 50x fewer parameters and <0.5 MB model size," Tech. Rep., 2017, ArXiv :1602.07360v4.

[47] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, and M. Andreetto, "MobileNets : Efficient convolutional neural networks for mobile vision applications," 2017, ArXiv :1704.04861v1.

[48] F. Karim, S. Majumdar, H. Darabi, and S. Chen, "LSTM fully convolutional networks for time series classification," *IEEE Access*, vol. 6, pp. 1662–1669, dec 2017.

[49] R. Pascanu, C. Gulcehre, K. Cho, and Y. Bengio, "How to construct deep recurrent neural networks," Tech. Rep., 2014, ArXiv :1312.6026v5.

[50] L. Mou, P. Ghamisi, and X. X. Zhu, "Deep recurrent neural networks for hyperspectral image classification," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 55, no. 7, pp. 3639–3655, jul 2017.

[51] L. Wei and E. Keogh, "Semi-supervised time series classification," in *Proceedings of the ACM International Conference on Knowledge Discovery and Data Mining*, 2006, pp. 748–753.

[52] R. Guidotti, A. Monreale, F. Spinnato, D. Pedreschi, and F. Giannotti, "Explaining any time series classifier," in *Proceedings of the International Conference on Cognitive Machine Intelligence*. Institute of Electrical and Electronics Engineers Inc., oct 2020, pp. 167–176.

[53] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, nov 1997.

[54] R. Jozefowicz, W. Zaremba, and I. Sutskever, "An empirical exploration of recurrent network architectures," in *Proceedings of the International Conference on International Conference on Machine Learning*. JMLR.org, 2015, p. 2342–2350.

[55] F. Gers, J. Schmidhuber, and F. Cummins, "Learning to forget : continual prediction with LSTM," in *Proceedings of the International Conference on Artificial Neural Networks*, no. , 1999, pp. 850–855 vol.2.

[56] D. Parra and C. Camargo, "A systematic literature review of hardware neural networks," in *Proceedings of the IEEE Colombian Conference on Applications in Computational Intelligence*, 2018, pp. 1–6.

[57] B. Li, J. Gu, and W. Jiang, "Artificial intelligence chip technology review," in *Proceedings of the International Conference on Machine Learning, Big Data and Business Intelligence*, 2019, pp. 114–117.

[58] A. Dundar, J. Jin, B. Martini, and E. Culurciello, "Embedded streaming deep neural networks accelerator with applications," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 28, no. 7, pp. 1572–1583, jul 2017.

[59] A. Reuther, P. Michaleas, M. Jones, V. Gadepally, S. Samsi, and J. Kepner, "Survey and benchmarking of machine learning accelerators," in *Proceedings of the High Performance Extreme Computing Conference*, 2019, pp. 1–9.

[60] C. Ding, S. Liao, Y. Wang, Z. Li, N. Liu, Y. Zhuo, C. Wang, X. Qian, Y. Bai, G. Yuan, X. Ma, Y. Zhang, J. Tang, Q. Qiu, X. Lin, and B. Yuan, "CIRCNN : Accelerating and compressing deep neural networks using block-circulant weight matrices," in *Proceedings of the Annual IEEE/ACM International Symposium on Microarchitecture*, 2017, pp. 395–408.

[61] J. Cheng, J. Wu, C. Leng, Y. Wang, and Q. Hu, "Quantized CNN : A unified approach to accelerate and compress convolutional networks," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 29, no. 10, pp. 4730–4743, oct 2018.

[62] C. Zhang, P. Li, G. Sun, Y. Guan, B. Xiao, and J. Cong, "Optimizing FPGA-based accelerator design for deep convolutional neural networks," in *Proceedings of the ACM/-SIGDA International Symposium on Field-Programmable Gate Arrays*. New York, NY, USA : Association for Computing Machinery, 2015, p. 161–170.

[63] GitHub - google/qkeras : QKeras : a quantization deep learning library for Tensorflow Keras. Accessed at 2023-02-16. [Online]. Available : https ://github.com/google/qkeras

[64] Han, Song and Pool, Jeff and Tran, John and Dally, William J., "Learning both weights and connections for efficient neural networks," in *Proceedings of the International Conference on Neural Information Processing Systems*. Cambridge, MA, USA : MIT Press, 2015, p. 1135–1143.

[65] F. F. *et al.*, "hls4ml : An open-source codesign workflow to empower scientific low-power machine learning devices," 2021, ArXiv :2103.05579.

[66] M. Blott, T. B. Preußer, N. J. Fraser, G. Gambardella, K. O'brien, Y. Umuroglu, M. Leeser, and K. Vissers, "FINN-R : An end-to-end deep-learning framework for fast exploration of quantized neural networks," *ACM Trans. Reconfigurable Technol. Syst.*, vol. 11, no. 3, dec 2018. [Online]. Available : https ://doi.org/10.1145/3242897

[67] M. M. Rahimifar, C.-É. Granger, Q. Wingering, B. Gouin-Ferland, H. Ezzaoui Rahali, and A. C. Therrien, "A survey of machine learning to FPGA tool-flows for instrumentation," in *Proceedings of the IEEE Nuclear Science Symposium and Medical Imaging Conference*, 2022.

[68] G. D. Guglielmo, F. Fahim, C. Herwig, M. B. Valentin, J. Duarte, C. Gingu, P. Harris, J. Hirschauer, M. Kwok, V. Loncar, Y. Luo, L. Miranda, J. Ngadiuba, D. Noonan, S. Ogrenci-Memik, M. Pierini, S. Summers, and N. Tran, "A reconfigurable neural network ASIC for detector front-end data compression at the HL-LHC," *IEEE Transactions on Nuclear Science*, vol. 68, no. 8, pp. 2179–2186, 2021.

[69] J. Duarte, S. Han, P. Harris, S. Jindariani, E. Kreinar, B. Kreis, J. Ngadiuba, M. Pierini, R. Rivera, N. Tran, and Z. Wu, "Fast inference of deep neural networks in FPGAs for particle physics," *Journal of Instrumentation*, vol. 13, p. 7027, 7 2018.

[70] P. Walter, A. Kamalov, A. Gatton, T. Driver, D. Bhogadi, J.-C. Castagna, X. Cheng, H. Shi, R. Obaid, J. Cryan, W. Helml, M. Ilchen, and R. N. Coffee, "Multi-resolution electron spectrometer array for future free-electron laser experiments," *Journal of Synchrotron Radiation*, vol. 28, no. 5, pp. 1364–1376, Sep 2021.

[71] A. Gholami, K. Kwon, B. Wu, Z. Tai, X. Yue, P. Jin, S. Zhao, and K. Keutzer, "SqueezeNext : Hardware-aware neural network design," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 2018, pp. 1638–1647.

[72] Y. Linde, A. Buzo, and R. Gray, "An algorithm for vector quantizer design," *Proceedings of the IEEE Transactions on Communications*, vol. 28, no. 1, pp. 84–95, 1980.

[73] H.-T. Liao and K.-S. Chen, "Mapping the landscape of green communications and green computing : A review based on bibliometric analysis," in *Proceedings of the IEEE International Conference on Communication Technology*, 2021, pp. 565–569.

[74] B. Gouin-Ferland, R. Coffee, and A. C. Therrien, "Data reduction through optimized scalar quantization for More compact neural networks," *Frontiers in Physics*, p. 887, sep 2022.

[75] A. C. Therrien, B. Gouin-Ferland, and M. M. Rahimifar, "Potential of edge machine learning for instrumentation," *Appl. Opt.*, vol. 61, no. 8, pp. 1930–1937, Mar 2022.