Co-conception d'un réseau de neurones convolutif de segmentation sémantique et de son architecture matérielle

Hugo LE BLEVEC[1] Mathieu LÉONARDON[1] Hugo TESSIER[1] Matthieu ARZEL[1]

[1]IMT Atlantique, Lab-STICC, UMR CNRS 6285, F-29238 Brest, France

Résumé – De nombreuses tâches de vision machine, comme la compréhension des scènes urbaines, reposent sur l'apprentissage automatique, et plus particulièrement sur les réseaux de neurones profonds, afin de fournir des résultats suffisamment précis pour permettre des technologies telles que les véhicules autonomes. Les réseaux de neurones profonds convolutifs étant très gourmands en ressources informatiques, les GPU sont généralement la cible matérielle privilégiée pour l'accélération. Cependant, ces applications embarquées ont une consommation d'énergie élevée et des exigences de latence qui ne sont pas compatibles avec l'utilisation des GPU. Les FPGA se sont révélés être une excellente cible pour le déploiement d'applications hautement parallèles, à faible latence et à faible consommation d'énergie de toutes sortes. La conception d'architectures matérielles pour les réseaux de neurones peut être une tâche très fastidieuse, en particulier pour les architectures complexes telles que celles utilisées pour la segmentation sémantique, qui sont composées de couches et de raccourcis spécifiques à différents niveaux du modèle. Des *frameworks* dédiés comme FINN offrent une solution pour faciliter leur développement. Dans ce travail, nous avons identifié les problèmes potentiels pour l'implémentation d'un tel modèle en utilisant FINN, nous les avons résolus et nous avons évalué les performances ainsi atteintes. Notre modèle, un codeur-décodeur convolutif basé sur U-Net, atteint 62.9 % mIoU avec une quantification entière de 4 bits. Une fois déployée sur la carte FPGA Xilinx Alveo U250, l'architecture de réseau neuronal mise en œuvre est capable de produire près de 20 images par seconde. Le code de ce travail est open-source et a été rendu public.

Abstract – Many machine vision tasks like urban scene-understanding rely on machine learning, and more specifically deep neural networks to provide accurate enough results to make technology like autonomous vehicles possible. As convolutional deep neural networks are computationally intensive, GPUs are usually the go-to hardware platform for acceleration. However, these embedded applications have high power consumption and latency requirements that are not compatible with the use of GPUs. FPGAs have proven to be an excellent target for deploying highly-parallel, low-latency and low-power applications of all sort. Designing hardware architectures for neural networks can be a very tedious task, especially for complex architectures as the ones used for semantic segmentation, which are composed of specific layers and shortcuts at different levels of the model. Dedicated frameworks like FINN offer a solution to ease the development. In this work, we have identified potential problems for the implementation of such a model using FINN, solved them, and evaluated the achievable performances. Our model, a convolutional encoder-decoder based on U-Net, achieves 62.9 % mIoU with a 4-bit integer quantization. Once deployed on Xilinx Alveo U250 FPGA board, the implemented neural network architecture is able to output close to 20 images per second. The code of this work is open-source and was released publicly.

1 Introduction

La segmentation sémantique est une tâche de vision par ordinateur visant à identifier et à séparer des objets dans une image en attribuant une étiquette à chaque pixel. Elle est devenue très populaire en raison de l'intérêt récent pour les applications nécessitant une compréhension de la scène, comme la conduite autonome ou la chirurgie assistée par ordinateur. Comme d'autres tâches de traitement d'images telles que la classification ou la détection d'objets, elle est implémentée par des réseaux de neurones profonds qui dépendent de l'accès à une puissance de calcul élevée, généralement sous la forme d'un ou de plusieurs GPU. Ce type de matériel gourmand en énergie n'est toutefois pas adapté aux applications embarquées dont le budget énergétique est limité et qui nécessitent une faible latence, comme les véhicules autonomes. Les FPGA, en revanche, représentent de très bonnes cibles pour déployer des réseaux de neurones profonds économes en énergie [13]. Par conséquent, de nombreux efforts ont été déployés pour surmonter les défis de la mise en œuvre de tels systèmes sur FPGA, qui sont particulièrement élevés pour les réseaux de segmentation sémantique, en raison de topologies plus complexes et d'une résolution plus élevée des représentations intermédiaires que celles des réseaux de classification. Dans cet article, nous nous concentrons sur la mise en œuvre d'un réseau neuronal personnalisé pour la segmentation sémantique, basé sur ResNet-18 et U-Net, en utilisant FINN[1], un *framework* expérimental open-source pour le déploiement de DNN sur FPGA [2],[12]. FINN s'appuie sur la synthèse de haut niveau (HLS) pour générer des blocs personnalisés organisés comme une architecture de flux de données en pipeline avec des poids stockés uniquement sur la mémoire sur puce et un contrôle du parallélisme de traitement. En tant que tel, il a le potentiel d'offrir un meilleur débit et une meilleure latence que les méthodes basées sur des unités de calcul programmables tels que le Deep Learning Processor Unit (DPU) de Xilinx Vitis AI. Cependant, la plupart des autres réseaux existants déployés avec ce *framework* ont une structure linéaire. Un travail antérieur a traité avec succès certaines des difficultés de mise en œuvre d'un réseau résiduel sur FPGA avec FINN [8], mais la complexité supplémentaire donnée par le décodeur U-Net de notre réseau a nécessité

[1] `xilinx.github.io/finn`

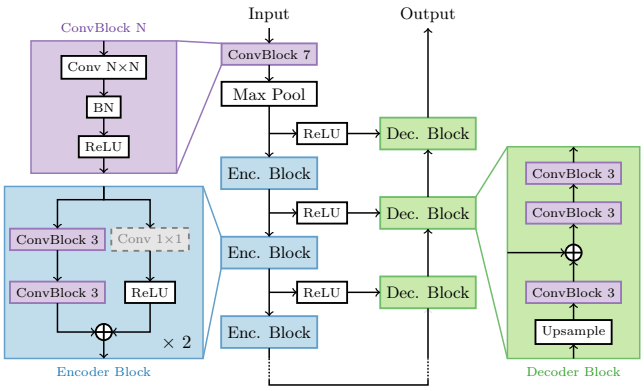


FIGURE 1 : Représentation de l'architecture de notre réseau de segmentation sémantique. "BN" signifie Batch Normalization. Le contenu du bloc encodeur est répété deux fois pour chaque bloc dans l'architecture. Comme dans l'architecture originale de ResNet-18, chaque raccourci peut contenir une couche de sous-échantillonnage sous la forme d'une convolution avec un noyau de 1x1, selon que la résolution de la carte des caractéristiques doit être réduite ou pas. Cela ne se produit généralement qu'une seule fois dans chaque bloc encodeur.

des évolutions supplémentaires à apporter au *framework* pour obtenir un accélérateur fonctionnel. L'intégralité du code est open-source[2].

2 Architecture

Les modèles à l'état de l'art basés sur HRNet [11] ont une structure trop complexe pour être mis en œuvre dans une architecture de flux de données en pipeline, et les exigences en matière de mémoire seraient sans doute trop élevées pour s'adapter à n'importe quel dispositif disponible sans l'utilisation d'une mémoire externe. À titre d'exemple, HRNet-18 (la version la plus légère) a 21 millions de paramètres, donc avec une quantification de 4 bits, les poids occuperaient 84 Mb de mémoire, ce qui est déjà assez conséquent pour un FPGA, et ce n'est pas la seule exigence en matière de mémoire, car l'architecture du réseau nécessite de stocker des représentations intermédiaires à haute résolution.

Des travaux récents, comme ENet [7] ou ERFNet [9] ont proposé des auto-encodeurs efficaces orientés vers la mise en œuvre matérielle qui seraient de bons candidats. Cependant, ils reposent sur l'utilisation de convolutions dilatées qui ne sont pas encore totalement prises en charge par FINN. Il nous reste donc des architectures d'encodeur-décodeurs convolutifs plus classiques, parmi lesquelles U-Net [10] est l'une des plus performantes, grâce à l'utilisation des sorties intermédiaires du codeur comme entrées secondaires supplémentaires dans le décodeur, ce qui s'est avéré très important pour améliorer la précision des réseaux de segmentation sémantique [4].

La figure 1 montre une représentation de l'architecture du réseau que nous proposons de mettre en œuvre avec FINN. Nous avons choisi d'utiliser la partie encodeur d'un ResNet-18 parce qu'il offre un bon rapport précision/nombre de paramètres par rapport à d'autres réseaux de classification populaires [5]. Nous avons ensuite procédé à l'architecture habituelle du décodeur d'un U-Net, composée de couches de suréchantillonnage et de convolutions reflétant l'encodeur. Nous utiliserions normalement la convolution transposée pour suréchantillonner les images, car il s'agit de la méthode habituelle et la meilleure pour suréchantillonner les images dans les réseaux neuronaux profonds. Cependant, FINN ne les prend pas en charge, et nous avons donc dû utiliser un suréchantillonnage par la méthodes des plus proches voisins, qui est moins précis que les convolutions transposées, lesquelles ont l'avantage d'avoir des paramètres entraînables qui garantissent que le suréchantillonnage est pertinent pour les données. Nous avons mesuré une perte d'environ 3 % en mIoU après modification.

Une modification essentielle qu'il a fallu appliquer à la topologie originale du réseau est le transfert des fonctions d'activation autour des raccourcis. Traditionnellement, la fonction d'activation Rectified Linear Unit (ReLU) est placée après l'addition. Cependant, l'existence de connexions vides rend difficile la conversion en couches HLS dans l'outil. Le fait de les déplacer avant les ajouts d'entrées latérales permets de résoudre ce problème et n'entraîne pas de baisse de précision. Enfin, nous avons modifié la topologie initiale du réseau en ajoutant une couche ReLU à la toute fin du réseau, après la dernière convolution qui produit une sortie avec 19 canaux, chacun correspondant à une classe de l'ensemble de données d'apprentissage. L'entraînement du réseau avec cet ajout ne modifie pas la précision de notre modèle. Cependant, il facilite la conversion vers les couches HLS de FINN.

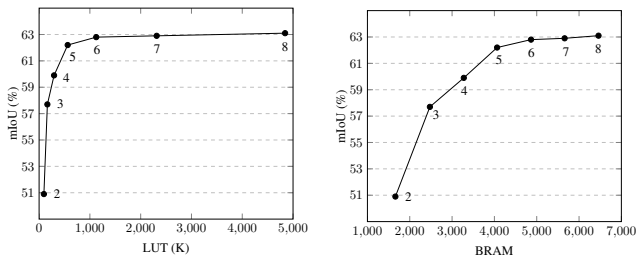
3 Exploration de l'espace de conception

La quantification est une méthode courante pour compresser les réseaux de neurones afin de réduire leur empreinte mémoire et l'utilisation des ressources de calcul. Pour l'implémentation sur FPGA, c'est même une exigence car ils ne sont pas adaptés aux opérations en virgule flottante, contrairement aux GPU. Des travaux antérieurs ont montré qu'il est possible d'utiliser une quantification sur 8 bits entiers sans perte de précision. Dans notre cas, une quantification sur 8 bits ferait que le réseau dépasserait largement les ressources disponibles sur la cible, nous avons donc dû trouver un niveau de quantification qui garantisse que le réseau puisse tenir sur un FPGA comme celui disponible sur une carte Xilinx Alveo U250 (typiquement utilisée dans les centres de données) sans utiliser de mémoire externe et sans trop sacrifier la précision du modèle. Dans cette sous-section, nous explorons les compromis entre la quantification et l'utilisation des ressources pour notre réseau afin de sélectionner la quantification appropriée. Nous avons utilisé Brevitas [6] pour effectuer la quantification du réseau. Il est paramétré pour utiliser une quantification uniforme avec un facteur d'échelle par couche appris au cours de l'apprentissage. Nous avons entraîné le réseau sur Cityscapes [3] pendant 200 époques en utilisant la descente de gradient stochastique comme optimiseur et la *Cross-Entropy* en fonction de coût, avec des niveaux de quantification de 2 à 8 bits, identiques pour les poids et les activations. Nous avons utilisé la "mean intersection-over-union" (mIoU) comme métrique de précision.

Pour évaluer et comparer l'utilisation des ressources as-

[2] https://anonymous.4open.science/r/finn-semseg-E2D7

sociée à chaque quantification, nous avons fixé les paramètres de parallélisme des couches du réseau dans FINN afin d'obtenir un débit estimé à environ 40 images par seconde (IPS) avec une horloge de 200 MHz.



(a) (b)

FIGURE 2 : Précision de segmentation de Cityscapes mesuré par la mIoU, en fonction de l'estimation du nombre de LUT (a) et de BRAM (b) utilisées.

Les figures 2a et 2b représentent respectivement la mIoU en fonction de l'utilisation estimée des LUTs et des BRAMs pour chaque quantification étudiée. Elles montrent qu'après 6 bits, il n'y a pratiquement pas de gain de précision, ce qui signifie que même si nous disposions de plus de ressources, nous n'aurions aucune raison d'aller au-delà de cette quantification. La figure 2a indique que la quantification sur 5 bits est le meilleur compromis entre l'utilisation de LUT et la précision du modèle, puisqu'il s'agit d'un point d'inflexion dans la courbe. Cependant, l'utilisation de BRAM est un peu trop proche de la quantité maximum disponible, et après avoir ajouté les autres éléments requis dans l'architecture spécifique au déploiement sur les cartes Alveo, nous n'avons pas pu mettre en œuvre la conception parce que l'utilisation de BRAM dépassait la quantité disponible. Nous avons donc privilégié une quantification en nombres entiers de 4 bits, qui constitue un meilleur compromis entre la précision du modèle et l'utilisation des ressources.

Nous avons encore amélioré la précision du modèle en ne mettant que la première et la dernière couche à 8 bits. Cette méthode s'est avérée efficace, car elle a permis d'atteindre un taux de précision de 62.9 %, avec un surcoût minime en termes d'utilisation des ressources. Ce faisant, les estimations de LUT et de BRAM sont respectivement de 367 322 et 3 327 unités. C'est donc ce schéma de quantification que nous avons choisi de mettre en œuvre.

4 Implémentation et résultats

Nous avons utilisé FINN pour convertir notre réseau en une architecture matérielle. Pour ce faire, nous avons dû apporté plusieurs modifications à certains éléments internes du *framework*, que nous ne détaillons pas ici. Les résultats de l'implémentation de cette architecture sur Alveo U250 sont présentés dans la Table 1.

Comme prévu, l'utilisation des LUT et BRAM sont plus élevées que les estimations faites dans la section 3. Toutes les FIFO importantes du réseau ont été placées sur UltraRAM (URAM) afin de tirer parti de cette ressource mémoire supplémentaire et de réduire l'utilisation de BRAM. Nous visions initialement une fréquence d'horloge de 200 MHz,

TABLE 1 : Utilisation des ressources de notre réseau sur Alveo U250

Resource	Utilisation	Disponible	Pourc. Util.
LUT	465,826	1,726,240	27%
LUTRAM	32,348	790,208	12%
FF	561,770	3,456,000	16%
BRAM	4,260	5,376	79%
URAM	388	1280	30%
DSP	1,043	12,288	8%

mais l'architecture n'a pas pu répondre à cette exigence et a abouti à une fréquence maximale de 134 MHz. Cela peut être dû à l'utilisation élevée de BRAM qui met trop de pression sur le placement, ainsi qu'aux spécificités du FPGA. Il est partitionné en 4 Super Logic Regions (SLR), ce qui rend les longues interconnexions difficilement compatibles avec une fréquence de fonctionnement élevée.

Nous avons utilisé le pilote basé sur la bibliothèque *Pynq* fourni par FINN pour évaluer le débit et la latence du système. Le réseau doit être alimenté en batchs d'entrées plutôt qu'en une seule image pour atteindre les meilleures performances possibles. Cela est lié au fonctionnement de l'architecture en pipeline de FINN [1] et aux caractéristiques des interfaces PCIe. Nous avons mesuré la latence d'exécution comme étant le temps écoulé entre le moment où nous envoyons le batch de données à la carte et le moment où nous le recevons. En tant que tel, il prend en compte non seulement le temps d'exécution, mais aussi le temps de transfert des données composé du temps de transfert PCIe et du temps de traitement des DMA d'entrée et de sortie. Nous avons effectué des mesures avec différentes tailles de batchs, de 1 à 10^4 par pas de puissances de dix. Les résultats sont présentés dans la figure 3.

Comme prévu, le débit augmente avec la taille du batch et atteint sa valeur maximale d'environ 20 IPS avec un batch d'au moins 100 images. Cette valeur est nettement inférieure à l'objectif initial de 40 IPS et s'explique principalement par deux facteurs. Premièrement, les FIFO que nous avons insérées manuellement dans la conception ne sont probablement pas parfaitement placées et dimensionnées. L'estimation de l'insertion de FIFO est assez difficile, car l'architecture du réseau et le comportement interne de chaque bloc matériel

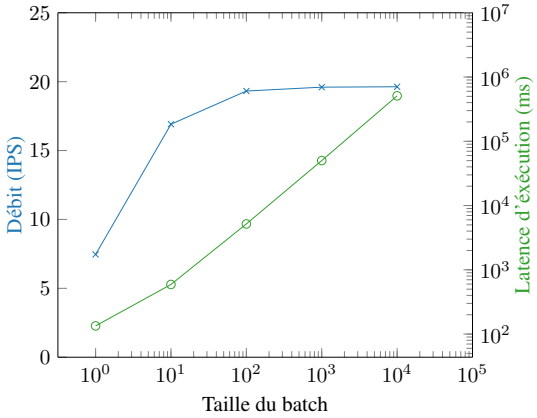


FIGURE 3 : Débit de segmentation en images par secondes (bleu, croix) and latence d'exécution en ms (vert, cercles) du réseau sur Alveo U250.

sont assez complexes. Les versions ultérieures de FINN pourraient également améliorer l'outil de dimensionnement automatique des FIFO pour les réseaux résiduels et faciliter cette tâche. Deuxièmement, notre conception n'a pu atteindre qu'une fréquence d'horloge de 134 MHz, par rapport aux 200 MHz visés et utilisés dans les premières estimations de débit. Si notre conception est capable de fonctionner à cette vitesse, elle produira près de 30 IPS. Nous pensons que d'autres optimisations de l'utilisation de la mémoire et la personnalisation du placement des SLR peuvent nous aider à réduire la longueur du chemin critique et à nous rapprocher des 200 MHz.

Ensuite, la latence évolue linéairement avec la taille du batch avec un surcoût correspondant à la latence de l'interface qui est visible pour les petites tailles de batch (1 ou 10) mais qui devient négligeable pour les tailles de batch plus élevées. Une régression linéaire donne une ordonnée à l'origine de 82 ms et une pente de 51 ms par entrée. Cette latence est un peu trop élevée pour être compatible avec la conduite autonome en temps réel qui vise une latence de l'ordre de la milliseconde. Comme pour le débit, cette latence peut être réduite sans changer le modèle en optimisant le placement des FIFO ainsi que la latence de chaque couche par un réglage plus fin des paramètres de repliage (*folding*).

5 Conclusion

Nous avons proposé une implémentation d'un réseau de segmentation sémantique quantifié à 4 bits sur FPGA en utilisant FINN. Nous avons adapté le réseau original, développé de nouveaux éléments dans l'outil, les avons rendus publics et open-source, et avons étudié les compromis entre la précision du modèle et l'utilisation des ressources pour pouvoir le déployer complètement sur Alveo U250. Notre conception atteint 62.9 % mIoU sur Cityscapes et peut atteindre un débit de 20 images par seconde. De plus, nous avons montré que FINN a le potentiel de générer des architectures matérielles à faible latence, même pour des réseaux de neurones de segmentation sémantique complexes et de grande taille.

Remerciements

Nous remercions les membres de l'équipe de recherche d'AMD Dublin, et particulièrement Thomas Preußer et Jakoba Petri-Koenig pour toute l'aide précieuse apportée pendant la réalisation de ces travaux.

Références

[1] Michaela BLOTT, Nicholas J. FRASER, Giulio GAMBARDELLA, Lisa HALDER, Johannes KATH, Zachary NEVEU, Yaman UMUROGLU, Alina VASILCIUC, Miriam LEESER et Linda DOYLE : Evaluation of Optimized CNNs on Heterogeneous Accelerators Using a Novel Benchmarking Approach. *IEEE Transactions on Computers*, 70(10):1654–1669, 2021.

[2] Michaela BLOTT, Thomas B PREUSSER, Nicholas J FRASER, Giulio GAMBARDELLA, Kenneth O'BRIEN, Yaman UMUROGLU, Miriam LEESER et Kees VISSERS : FINN-R : An end-to-end deep-learning framework for fast exploration of quantized neural networks. *ACM Transactions on Reconfigurable Technology and Systems (TRETS)*, 11(3):1–23, 2018.

[3] Marius CORDTS, Mohamed OMRAN, Sebastian RAMOS, Timo REHFELD, Markus ENZWEILER, Rodrigo BENENSON, Uwe FRANKE, Stefan ROTH et Bernt SCHIELE : The Cityscapes Dataset for Semantic Urban Scene Understanding. *In Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[4] Michal DROZDZAL, Eugene VORONTSOV, Gabriel CHARTRAND, Samuel KADOURY et Chris PAL : The Importance of Skip Connections in Biomedical Image Segmentation. *CoRR*, abs/1608.04117, 2016.

[5] Kaiming HE, Xiangyu ZHANG, Shaoqing REN et Jian SUN : Deep Residual Learning for Image Recognition. *In 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016.

[6] Alessandro PAPPALARDO : Xilinx/brevitas, 2022.

[7] Adam PASZKE, Abhishek CHAURASIA, Sangpil KIM et Eugenio CULURCIELLO : ENet : A Deep Neural Network Architecture for Real-Time Semantic Segmentation. *CoRR*, abs/1606.02147, 2016.

[8] Lucian PETRICA, Tobias ALONSO, Mairin KROES, Nicholas FRASER, Sorin COTOFANA et Michaela BLOTT : Memory-Efficient Dataflow Inference for Deep CNNs on FPGA. *In 2020 International Conference on Field-Programmable Technology (ICFPT)*, pages 48–55, 2020.

[9] Eduardo ROMERA, José M. ÁLVAREZ, Luis M. BERGASA et Roberto ARROYO : ERFNet : Efficient Residual Factorized ConvNet for Real-Time Semantic Segmentation. *IEEE Transactions on Intelligent Transportation Systems*, 19(1):263–272, 2018.

[10] Olaf RONNEBERGER, Philipp FISCHER et Thomas BROX : U-Net : Convolutional Networks for Biomedical Image Segmentation. *CoRR*, abs/1505.04597, 2015.

[11] Ke SUN, Bin XIAO, Dong LIU et Jingdong WANG : Deep High-Resolution Representation Learning for Human Pose Estimation. *In 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5686–5696, 2019.

[12] Yaman UMUROGLU, Nicholas J. FRASER, Giulio GAMBARDELLA, Michaela BLOTT, Philip LEONG, Magnus JAHRE et Kees VISSERS : FINN : A Framework for Fast, Scalable Binarized Neural Network Inference. *In Proceedings of the 2017 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays*, FPGA '17, pages 65–74. ACM, 2017.

[13] Xiaowei XU, Xinyi ZHANG, Bei YU, Xiaobo Sharon HU, Christopher ROWEN, Jingtong HU et Yiyu SHI : DAC-SDC Low Power Object Detection Challenge for UAV Applications. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 43(2):392–403, 2021.