Réduction de la Résolution comme Technique de Compression pour les Systèmes d'Apprentissage Profond en Vision

Jérémy MORLIER[1] Mathieu LÉONARDON[1] Vincent GRIPON[1]
[1]IMT Atlantique, Lab-STICC, UMR CNRS 6285, F-29238 Brest, France

Résumé – Dans cet article, nous évaluons et proposons de nouvelles méthodes de compression de réseaux de neurones pour des tâches de classification d'images qui utilisent la résolution des images utilisées. Les résultats montrent que ces méthodes permettent d'atteindre de nouveaux compromis entre mémoire et nombre de calculs en gardant de bonnes performances.

Abstract – In this paper, we evaluate and propose new neural network compression methods for image classification tasks based on input image resolution. The results show that these methods enable new trade-offs between memory and computation while maintaining strong performance.

1 Introduction

La réduction de la taille des modèles d'apprentissage profond est essentielle, notamment en vision par ordinateur où les grands modèles nécessitent beaucoup de ressources. Cette compression est cruciale pour des cibles ayant de faibles ressources matérielles comme les appareils mobiles et les systèmes embarqués mais également pour des cibles moins contraintes comme les datacenters.

Plusieurs méthodes de compression de modèles ont été proposées, telles que l'élagage [5], la quantification [10], la distillation de connaissances, et l'approximation de rang faible.

Dans ce travail, nous explorons la réduction de la résolution des images en entrée comme technique de compression post-entraînement. Cette approche, moins utilisée, pourrait réduire significativement les coûts calculatoires et la mémoire requise, en affectant directement la taille des activations dans les réseaux de neurones convolutionels (CNNs) et en réduisant la taille de la séquence dans les architectures basées sur les *Vision transformers* (ViTs) [2].

Nous cherchons à déterminer si la réduction de la résolution d'entrée peut rivaliser ou surpasser les méthodes traditionnelles de compression de modèles.

2 Travaux connexes

Plusieurs études ont exploré l'utilisation de la résolution comme un facteur d'échelle pour la compression des réseaux de neurones.

Pour les réseaux de neurones convolutifs, la résolution des images est un élément important lors de la conception des réseaux, avec des tailles variant de 224x224 pour la classification d'images à 1024x1024 pour la segmentation sémantique. Les architectures MobileNet [8] ont introduit la résolution comme un facteur de compression lors de la conception de l'architecture. EfficientNet [14] combine largeur, profondeur et résolution dans une architecture basée sur MobileNetv2 pour optimiser les performances. D'autres études [18, 1] analysent les impacts variés de la résolution sur la conception, l'entraînement et l'évaluation des systèmes d'apprentissage profond.

Pour les réseaux de neurones transformers, l'utilisation de tailles de *patch* plus petites, notamment dans l'architecture de transformers Swin [13], permet d'améliorer les performances pour les tâches nécessitant plus de détails. VITAR [3] et UniVit [12] se concentrent sur l'amélioration des encodages des positions pour améliorer la performance des transformers à différentes résolutions d'images.

3 Méthodologie

La compression des réseaux de neurones, au sens de la réduction du nombre de calculs et de la mémoire nécessaire à leurs exécutions, a connu de nombreuses contributions [7, 10, 6, 4] avec des techniques comme l'élagage, la quantification ou la distillation de connaissances. En particulier, la mise à l'échelle des modèles (*model scaling*), introduite par EfficientNet [14], combinait initialement largeur, profondeur et résolution d'entrée. En dehors des architectures EfficientNets, il est plus courant de modifier à la fois la largeur et la profondeur pour concevoir une architecture adaptée à un problème spécifique [11, 19, 16].

Notre étude vise à explorer systématiquement les compromis entre performance et complexité pour les architectures de réseaux de neurones (CNNs et ViTs). Nous montrons que les compromis entre le coût calculatoire et les besoins en mémoire obtenus par les méthodes de *model scaling* et de compression par la résolution (*resolution scaling*) varient différemment et que les méthodes de *resolution scaling* permettent d'atteindre de nouveaux compromis.

3.1 Évaluation des coûts calculatoires et de mémoire

L'évaluation des coûts calculatoires et des besoins en mémoire d'un réseau de neurones est complexe [9, 17]. En effet, l'évaluation de la compression d'un réseau de neurones avec des métriques réelles comme la latence, le débit ou la consommation énergétique dépendent fortement de la cible matérielle considérée et de la compilation du réseau de neurones. Dans un but de généralisation, nous considérons, dans ce papier,

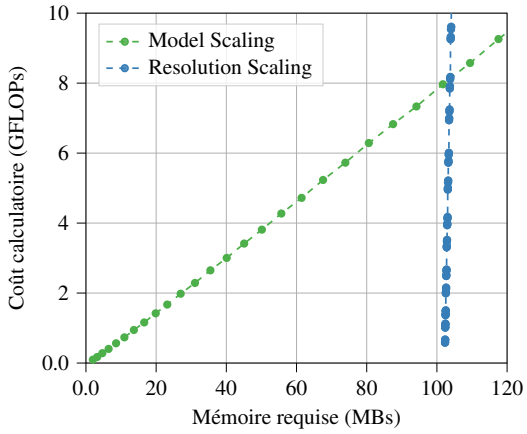

FIGURE 1 : Comparaison des coûts calculatoires et des besoins en mémoires pour des méthodes de *model scaling* et de *resolution scaling* pour un ResNet50.

deux métriques simplifiées, une pour le coût calculatoire et une pour les besoins en mémoire. Pour le coût calculatoire, nous estimons ce dernier à partir du nombre d'opérations en virgule flottante (FLOPs), une métrique usuelle en compression [15]. Pour la mémoire, nous considérons la somme de la taille des poids du modèle avec la taille maximale pour stocker les entrées et sorties d'un opérateur du réseau de neurones.

3.2 Dimensionnement des CNNs et des ViTs

Les architectures modernes de réseaux de neurones pour la vision comme les CNNs et les ViTs ont des résolutions d'entrées différentes. Les CNNs utilisent directement les images avec une résolution d'entrée qui correspond au nombres de pixels par ligne ou colonne. Pour les ViTs, l'image est d'abord découpée en sous-parties appelées *patch* qui sont ensuite encodées en une séquence d'éléments appelés *tokens* pour finalement être utilisée dans le réseau de neurones. La résolution vue par un ViT est donc le nombre de *tokens* par ligne ou colonne de l'image initiale.

Les CNNs sont principalement constitués de couches convolutives avec un nombre de FLOPs dans une couche convolutive proportionnel au nombre de canaux entrants et sortants, et au carré de la taille du noyau et de la résolution entrante. La taille des activations dépendent du carré de la résolution entrante et de la somme des canaux entrants et sortants. Modifier la résolution d'entrée a donc un impact quadratique sur les calculs et la mémoire. Cette différence est présentée dans la figure 1.

Pour les ViTs, les FLOPs d'un bloc ViT dépendent de la dimension interne (*hidden dim*), de la dimension cachée des MLPs (*MLP dim*) et de la taille de la séquence de tokens (*sequence scaling*), avec une complexité en $\mathcal{O}(N^4)$ pour le nombre d'opérations et en $\mathcal{O}(N^2)$ pour la mémoire. Nous ne considérons pas les couches d'encodages et de classification pour le calcul de la taille du modèle car ces dernières sont relativement de faible taille par rapport au reste du réseau dans un ViT. L'empreinte mémoire d'un ViT est donc principalement définie par la dimension interne et les couches MLP. Nous présentons les différents compromis mémoire/calcul possibles pour un ViT sur la figure 2.

3.3 Méthodes

Les CNNs et les ViTs utilisent des traitements différents pour les images, nous envisageons donc différentes méthodes de *resolution scaling* propres à ces architectures.

CNNs : Les CNNs emploient généralement un pipeline de prétraitement d'image différent en entraînement et en évaluation. Le pipeline d'entraînement est principalement composé d'un découpage aléatoire redimensionné de taille K_{train} et le pipeline d'évaluation est principalement composé d'un redimensionnement aléatoire de taille K suivi d'un découpage central de taille K_{eval}. Comme montré dans [18], cette différence de pipeline implique une disparité pour le réseau de neurones entre la phase d'entraînement et la phase d'évaluation, résultant en une résolution K_{eval} en évaluation plus élevée que celle utilisée pendant l'entraînement K_{train}. En utilisant cette disparité, nous proposons de réduire la taille du découpage aléatoire utilisé pendant l'entraînement afin d'entraîner un CNN qui peut être plus efficace pour une résolution d'évaluation plus basse et ainsi réduire ses coûts en calcul et en mémoire.

Pour notre deuxième méthode, nous proposons d'amplifier cet effet de disparité en employant une résolution d'entraînement plus élevée, puis en réduisant la résolution en sous-échantillonnant les activations dans les couches plus profondes.

ViTs : Les ViTs emploient généralement un pipeline de prétraitement similaire à celui des CNNs, mais avec la taille de découpage d'évaluation K égale à la taille de découpage d'entraînement K_{train}. Nous entraînons les ViTs pour une résolution spécifique en utilisant une taille de séquence fixe, contrôlant ainsi efficacement le coût calculatoire et les besoins en mémoire. Comme les images sont divisées en *patchs* pour créer la séquence de *tokens*, la taille des *patchs* est un autre facteur important dans la conception d'un ViT pour une résolution spécifique (c'est-à-dire la taille de la séquence). Dans ce travail, nous évaluons également la relation entre la taille des *patchs* et la longueur de la séquence.

La méthodologie proposée consiste à évaluer systématiquement la relation entre la résolution d'entrée et la taille des *patchs* afin de réduire les calculs et la mémoire.

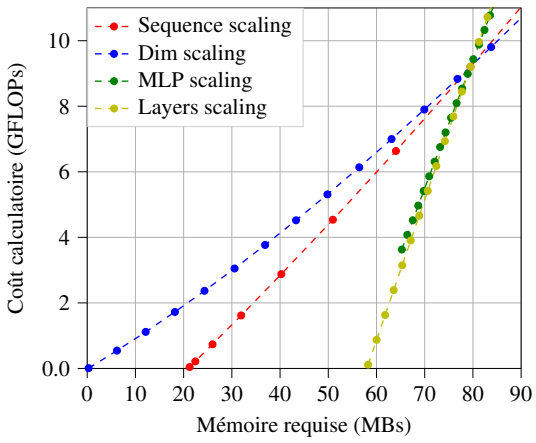

FIGURE 2 : Comparaison des coûts calculatoires et des besoins en mémoires pour des méthodes de *model scaling* et de *resolution scaling* pour un ViT-Small.

4 Expériences

4.1 CNNs

Dans cette section, nous évaluons nos méthodes de *resolution scaling* sur un réseau de neurones ResNet-50 pour une tâche très étudiée de classification d'images sur le jeu de données ImageNet. Chaque ResNet50 utilisé est entraîné avec le processus d'entraînement de torchvision qui applique une augmentation de données incluant un découpage aléatoire de taille K_{train}, avec 176 comme valeur de base. L'évaluation standard sur ImageNet consiste à normaliser l'image, la redimensionner à une taille K, puis effectuer un découpage central de taille K_{eval}, avec comme valeurs par défaut 232 et 224 respectivement.

Nous comparons deux méthodes. La première consiste à entraîner un ResNet-50 avec différentes tailles $K_{train} \in 64, 128, 160, 176$, et à l'évaluer avec plusieurs couples (K, K_{eval}). La seconde méthode entraîne avec $K_{train} = 224$ puis réduit la résolution après la première convolution, en adaptant les activations aux résolutions utilisées dans la première méthode. Cela permet d'avoir des coûts d'entraînement comparables entre les deux approches.

Les résultats présentés sur la Figure 3 confirment, comme dans [18], que la meilleure précision est atteinte à une résolution d'évaluation plus élevée que celle d'entraînement. La deuxième méthode surpasse la première ainsi que le modèle de base pour un même coût d'entraînement.

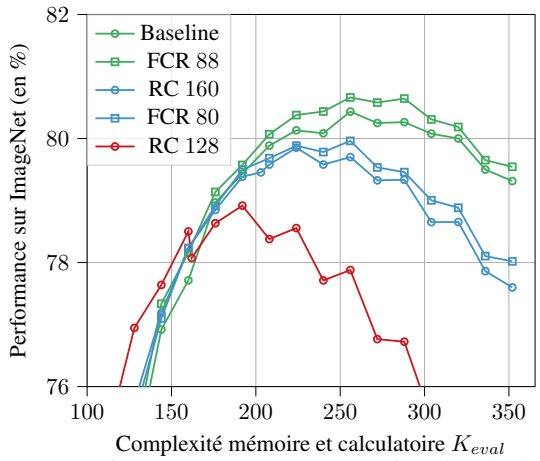

FIGURE 3 : Performance sur Imagenet en fonction de la résolution d'inférence K_{eval} pour des résolutions d'entrainement (RC) K_{train} et pour des résolution réduites après la première couche convolutive (FCR).

Nous comparons également nos approches avec une méthode de *model scaling*. En appliquant une réduction uniforme du nombre de filtres dans les couches convolutives (facteurs 0.5 et 0.25), nous constatons sur la Figure 4 que les méthodes de *resolution scaling* offrent des compromis intéressants entre performance et complexité, inaccessibles pour le *model scaling*.

4.2 ViTs

Nous explorons les méthodes de *resolution scaling* sur les ViTs en entraînant ViT-S à différentes résolutions 8, 11, 12, 13, 14, 15 avec une taille de *patch* fixe de 16x16. Les résultats montrent qu'une réduction de la résolution entraîne une

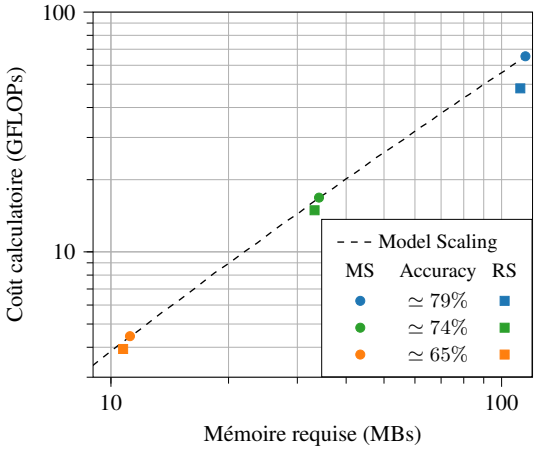

FIGURE 4 : Comparaison des méthodes de *model scaling* (MS) et de *resolution scaling* (RS) pour un ResNet-50. Chaque couleur correspond à une performance sur ImageNet équivalente.

baisse limitée de précision (-1%) tout en réduisant les FLOPs de 28%.

Nous étudions ensuite la relation entre taille des *patchs* et résolution d'architecture (taille de séquence par ligne/colonne) en testant deux résolutions (9 et 11) avec différentes tailles de *patchs* 8, 12, 16, 24 et 32. Les meilleurs résultats sont obtenus avec des tailles de *patch* de 16x16 (pour 9) et 12x12 (pour 11), correspondant à des résolutions d'image proches (128x128 et 132x132), ce qui souligne l'importance d'adapter la résolution d'image au jeu de données et à la tâche considérés.

Enfin, nous comparons les méthodes de *resolution scaling* à celles de *model scaling* en ajustant les paramètres internes des ViTs d'un modèle de base afin d'obtenir un nombre de FLOPs égal à chaque résolution. Le tableau 1 montre que la réduction de la résolution est plus efficace que la réduction du modèle en cas de contraintes calculatoires, sauf pour le *model scaling* des couches MLPs qui peut être meilleur à haute résolution, mais moins performant par rapport aux méthodes de *resolution scaling* en basses résolutions.

5 Conclusion

Dans ce travail, nous avons proposé de nouvelles méthodes de compression de réseaux de neurones, convolutionels ou transformers, se basant sur la résolution. En évaluant de manière systématique, nous avons montré que le changement de résolution peut être une méthode de compression efficace, seule ou en complément d'autres méthodes comme le *model scaling*. L'utilisation de la résolution dans les méthodes de compression ouvre des voies pour de nouvelles méthodes de compression et d'optimisation afin de concevoir des réseaux de neurones plus efficaces et performants.

Références

[1] Piotr DOLLÁR, Mannat SINGH et Ross GIRSHICK : Fast and accurate model scaling. *In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 924–932, 2021.

[2] Alexey DOSOVITSKIY, Lucas BEYER, Alexander KOLESNIKOV, Dirk WEISSENBORN, Xiaohua ZHAI, Tho-

Résolution 9

Méthode		**Mémoire Requise** (MBs)	**FLOPs** (GFLOPs)	**Perf.**
Hidden Size Scaling		25.9	4.5	47.54
MLP Scaling		66.6	4.7	<u>68.89</u>
Depth Scaling		65.3	5.0	62.14
Hybrid Scaling		39.4	4.8	67.64
Resolution Scaling	8	26.1	4.7	69.16
	12	32.0	4.7	69.77
	16	40.3	4.7	**70.64**
	24	63.9	4.8	68.55
	32	97.0	4.8	69.31

Résolution 11

Method		**Mémoire Requise** (MBs)	**FLOPs** (GFLOPs)	**Perf.**
Hidden Size Scaling		50.7	9.1	71.02
MLP Scaling		72.1	9.0	**73.68**
Depth Scaling		70.6	8.7	70.92
Resolution Scaling	8	30.4	8.9	73.21
	12	41.6	9.0	<u>73.48</u>
	16	57.2	9.0	73.23
	24	101.8	9.1	72.89
	32	164.4	9.2	72.98

TABLE 1 : Comparaison des méthodes de *model scaling* et de *resolution scaling* pour une architecture ViT évaluée sur ImageNet. Pour les méthodes de *resolution scaling*, la taille de la séquence est égale à la résolution et chaque taille de *patch* est évalué. Les méthodes de *model scaling* utilisent les dimensions internes des ViTs, MLP, dimension cachée (hidden dim), nombre de couches (depth). Une méthode intermédiaire (hybrid scaling) correspond à une modification similaire aux différences de tailles entre un ViT-B et un ViT-S.

mas UNTERTHINER, Mostafa DEHGHANI, Matthias MINDERER, Georg HEIGOLD, Sylvain GELLY, Jakob USZKOREIT et Neil HOULSBY : An Image is Worth 16x16 Words : Transformers for Image Recognition at Scale, juin 2021.

[3] Qihang FAN, Quanzeng YOU, Xiaotian HAN, Yongfei LIU, Yunzhe TAO, Huaibo HUANG, Ran HE et Hongxia YANG : Vitar : Vision transformer with any resolution. *arXiv preprint arXiv :2403.18361*, 2024.

[4] Song HAN, Huizi MAO et William J DALLY : Deep compression : Compressing deep neural networks with pruning, trained quantization and huffman coding. *arXiv preprint arXiv :1510.00149*, 2015.

[5] Song HAN, Jeff POOL, John TRAN et William DALLY : Learning both weights and connections for efficient neural network. *Advances in neural information processing systems*, 28, 2015.

[6] Yihui HE, Xiangyu ZHANG et Jian SUN : Channel pruning for accelerating very deep neural networks. *In Proceedings of the IEEE international conference on computer vision*, pages 1389–1397, 2017.

[7] Geoffrey HINTON : Distilling the knowledge in a neural network. *arXiv preprint arXiv :1503.02531*, 2015.

[8] Andrew G HOWARD : Mobilenets : Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv :1704.04861*, 2017.

[9] Andrei IVANOV, Nikoli DRYDEN, Tal BEN-NUN, Shigang LI et Torsten HOEFLER : Data movement is all you need : A case study on optimizing transformers. *Proceedings of Machine Learning and Systems*, 3:711–732, 2021.

[10] Benoit JACOB, Skirmantas KLIGYS, Bo CHEN, Menglong ZHU, Matthew TANG, Andrew HOWARD, Hartwig ADAM et Dmitry KALENICHENKO : Quantization and training of neural networks for efficient integer-arithmetic-only inference. *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[11] Eugene LEE et Chen-Yi LEE : Neuralscale : Efficient scaling of neurons for resource-constrained deep neural networks. *In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1478–1487, 2020.

[12] Tatiana LIKHOMANENKO, Qiantong XU, Gabriel SYNNAEVE, Ronan COLLOBERT et Alex ROGOZHNIKOV : Cape : Encoding relative positions with continuous augmented positional embeddings. *Advances in Neural Information Processing Systems*, 34:16079–16092, 2021.

[13] Ze LIU, Yutong LIN, Yue CAO, Han HU, Yixuan WEI, Zheng ZHANG, Stephen LIN et Baining GUO : Swin transformer : Hierarchical vision transformer using shifted windows. *In Proceedings of the IEEE/CVF international conference on computer vision*, pages 10012–10022, 2021.

[14] Yury PISARCHYK et Juhyun LEE : Efficient memory management for deep neural net inference. *arXiv preprint arXiv :2001.03288*, 2020.

[15] Mingxing TAN et Quoc V. LE : EfficientNetV2 : Smaller Models and Faster Training, juin 2021.

[16] Yehui TANG, Yunhe WANG, Jianyuan GUO, Zhijun TU, Kai HAN, Hailin HU et Dacheng TAO : A survey on transformer compression. *arXiv preprint arXiv :2402.05964*, 2024.

[17] Neil C THOMPSON, Kristjan GREENEWALD, Keeheon LEE et Gabriel F MANSO : The computational limits of deep learning. *arXiv preprint arXiv :2007.05558*, 10, 2020.

[18] Hugo TOUVRON, Andrea VEDALDI, Matthijs DOUZE et Hervé JÉGOU : Fixing the train-test resolution discrepancy, janvier 2022.

[19] Xiaohua ZHAI, Alexander KOLESNIKOV, Neil HOULSBY et Lucas BEYER : Scaling vision transformers. *In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12104–12113, 2022.