Apprentissage incrémental sous contrainte mémoire : stratégie de mise à l'échelle de réseaux de neurones

Eva Feillet[1,2], Adrian Popescu[1], Céline Hudelot[2], et Marina Reyboz[3]
[1]Université Paris-Saclay, CEA, LIST F-91120, Palaiseau, France
[2]MICS, Université Paris-Saclay, CentraleSupélec, France
[3]Université Grenoble Alpes, CEA, LIST, F-38000 Grenoble, France
{eva.feillet,adrian.popescu,marina.reyboz}@cea.fr, celine.hudelot@centralesupelec.fr

1 Introduction

Les réseaux de neurones profonds excellent aujourd'hui dans les tâches d'apprentissage supervisé à distribution de données fixée. Cependant, il n'est pas toujours possible de disposer simultanément de toutes les données pour entraîner un modèle profond. Face à ce défi, l'apprentissage continu vise à construire des modèles capables d'apprendre de manière séquentielle et d'intégrer de nouvelles connaissances au cours du temps sans pour autant accéder aux données passées.
En particulier, l'apprentissage incrémental par classes (*class-incremental learning*, CIL) traite des problèmes de classification où l'apprentissage se fait par étapes. Une étape consiste à intégrer au modèle de classification un ensemble de nouvelles classes. Dans le scénario dit "sans mémoire", qui est le plus difficile, à chaque étape, seules les données relatives aux nouvelles classes sont accessibles.
Un champ d'application majeur de l'apprentissage incrémental est la vision par ordinateur pour les systèmes embarqués, lesquels sont sujets à des contraintes pratiques fortes, comme le budget mémoire que l'on peut attribuer au modèle. Les applications embarquées requièrent donc des architectures ayant un nombre de paramètres relativement limité. Or, à ce jour, la majorité de la littérature en CIL évalue ses méthodes sur la base de réseaux trop volumineux pour les dispositifs ciblés [1, 2]. Un fossé sépare donc les performances espérées et celles atteignables en pratique.
Ce travail met en évidence le besoin de stratégies pour améliorer les performances des modèles neuronaux profonds sous contraintes de mémoire. Après une brève revue des travaux existants (Section 2), nous présentons des expériences de mise à l'échelle de réseaux de neurones (Section 3). Celles-ci nous permettent de proposer une méthode de mise à l'échelle sous contrainte mémoire d'un réseau de neurones convolutionnel (Section 4). Enfin, nous concluons avec quelques perspectives.

2 Travaux connexes

Dans cette section, nous effectuons une brève revue des approches proposées visant à adapter l'architecture d'un réseau de neurones à des contraintes mémoires.

2.1 Recherche d'architecture sous contrainte

La recherche d'architecture neuronale (*neural architecture search*, NAS) est une procédure pour optimiser les performances des architectures profondes pour une tâche d'apprentissage [4]. En particulier, des contraintes comme le nombre de paramètres ou le temps de latence peuvent être prises en compte [20]. Dans le cas de l'apprentissage incrémental, plusieurs approches à base d'apprentissage par renforcement sont proposées [10, 11]. Cependant, ces procédures de recherche d'architecture sont généralement longues et coûteuses, car elles explorent un large espace de configurations et requièrent l'entraînement des modèles candidats pour évaluer une architecture. Ces approches supposent également l'accès aux données de l'ensemble des étapes d'apprentissage. Elles ne conviennent donc pas au cas pratique d'un modèle s'adaptant rapidement aux environnements dynamiques rencontrés en CIL, où les données arrivent en flux et les distributions peuvent changer.

2.2 Compression et élagage

À partir d'une architecture et d'une plateforme matérielle, on peut recourir à la quantification pour adapter le modèle à la taille mémoire disponible. Il existe d'une part des méthodes post-entraînement, quantifiant un modèle déjà entraîné [12], et d'autre part des méthodes permettant l'entraînement complet d'un réseau quantifié [5, 6]. Par ailleurs, dans le cadre de l'apprentissage par transfert, des approches par élagage de réseaux déjà entraînés (*pruning*) ont également été proposées afin de réduire la redondance des poids [17]. Comme dans le cas de la NAS, le problème de la disponibilité des données se pose. En apprentissage incrémental, la question de la taille des réseaux doit d'abord être étudiée sous l'angle de leur architecture plutôt qu'en quantifiant ou élaguant des réseaux trop volumineux.

2.3 Stratégies de mise à l'échelle

Pour une tâche de classification supervisée standard, les auteurs de MobileNets [9] ont proposé une procédure de mise à l'échelle de leur architecture. Ils introduisent ainsi des hyperparamètres contrôlant la taille du réseau afin de trouver la configuration la plus performante. Cette approche a été affinée par les auteurs de EfficientNet [19], qui ont mis à l'échelle leur réseau en jouant simultanément sur la profondeur (nombre de couches ou blocs), la largeur (nombre de filtres de convolution) et la résolution (taille d'entrée des images) selon une méthode appelée *compound scaling*. Toutefois, cette approche exige de calibrer des hyperparamètres, processus coûteux et approximatif dans le cas où l'utilisateur n'a accès qu'à une sous-partie des classes.

2.4 Influence de l'architecture sur les performances

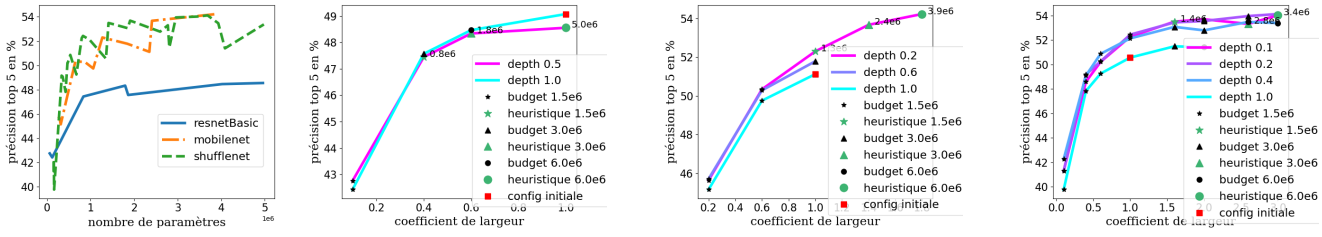
Les expériences de [15, 16] ont montré l'influence de l'architecture du réseau sur les performances des modèles d'apprentissage incrémental. L'apprentissage continu est confronté à l'oubli catastrophique qui consiste en la perte des informations apprises lors d'une tâche précédente après un entraînement sur des tâches ultérieures [14]. Les auteurs de [15] ont montré qu'augmenter la largeur des réseaux de neurones permet d'atténuer l'oubli catastrophique et d'augmenter la précision du modèle. De plus, ils ont observé qu'augmenter la profondeur a un effet nul voire négatif sur les performances du modèle. Ces observations ont été confirmées par [16] où, dans le cas d'un apprentissage incrémental, l'influence des composants architecturaux classiques comme la normalisation par batch est étudiée. Notons que les résultats rapportés dans ces deux articles sont obtenus avec des modèles surparamétrés et sans aucun mécanisme venant contrer explicitement l'oubli catastrophique. Dans ce qui suit, nous cherchons à déterminer si les améliorations de performance rapportées par [15, 16] se confirment dans le cas de petits réseaux pour lesquels nous ne savons pas s'ils sont surparamétrés ou non.

3 Expériences

Afin de mettre au point une méthode de mise à l'échelle d'architecture, nous avons mené une étude expérimentale répondant à deux questions, inspirées par les travaux de [15, 19]. Pour un budget de mémoire donné : (1) vaut-il mieux que l'architecture du réseau soit plus profonde ou plus large ? (2) vaut-il mieux mettre à l'échelle uniquement selon la largeur (resp. la profondeur) ou selon les deux à la fois ?
Nous avons considéré la mise à l'échelle de trois réseaux convolutionnels de référence, à savoir ResNet18 (11.7M paramètres) [7], MobileNetv2 (3.5M) [9, 18] et ShuffleNetv2 (2.3M) [13], pour trois budgets mémoires de petite taille (1.5, 3.0 et 6.0 millions de paramètres respectivement). Les modèles mis à l'échelle sont entraînés en utilisant une adaptation sans mémoire (i.e, ne stockant aucune image d'entraînement au cours de l'apprentissage) de l'algorithme Learning a Unifed Classifer Incrementally via Rebalancing (LUCIR) [8]. Les modèles sont évalués sur la base de leur précision (*accuracy*) incrémentale, calculée sur l'ensemble des classes à l'issue de la dernière étape incrémentale. Enfin, le problème de classification consiste à apprendre cent classes issues du jeu de données ImageNet [3], réparties équitablement sur dix états. Dans ce qui suit, nous proposons une approche de mise à l'échelle de réseaux qui s'appuie sur les observations de ces expériences.

4 Méthode proposée

La méthode proposée adopte le point de vue d'un utilisateur souhaitant mettre en place une méthode d'apprentissage incrémental pour un budget mémoire donné m et sans effectuer de calculs préliminaires. Soit une architecture de référence A_{ref} comportant $n_{ref} > m$ paramètres. Nous proposons l'heuristique suivante $\psi : (m, A_{ref}) \rightarrow (A, w, d)$, où $d \in]0, 1]$ (resp. $w \in \mathbb{R}^{*+}$) est un coefficient de profondeur (resp. largeur) appliqué à A_{ref} qui permet d'obtenir l'architecture A respectant la contrainte mémoire. L'architecture A comportant n paramètres est obtenue en multipliant uniformément le nombre de couches de A_{ref} par d, et le nombre de filtres de convolution par w. Le coefficient de profondeur d est choisi en premier de manière à rendre le réseau le moins profond possible tout en préservant la structure du réseau de départ. Le coefficient de largeur w est choisi de manière à maximiser le nombre de filtres de convolution tout en respectant la contrainte du budget mémoire, i.e, $n \leq m$. Pour tester notre méthode, nous avons mis à l'échelle trois réseaux de référence. Pour un réseau et un budget mémoire fixés, les performances de différentes configurations (w, d), dont celle recommandée par notre heuristique (en vert) sont présentées (Figures 1b, 1c, 1d). Les modèles sont évalués sur la base de leur précision en top 5.



(a) Précision selon #params (b) ResNet18 (c) MobileNetv2 (d) ShuffleNetv2

FIGURE 1 – Mise à l'échelle de trois réseaux de référence pour trois budgets mémoire (entraînement via LUCIR).

5 Conclusion et perspectives

Nous avons étudié le problème de mise à l'échelle de réseau de neurones dans le cas de l'apprentissage incrémental par classe. Nos premières expériences ont montré que, pour un budget mémoire donné, une mise à l'échelle ayant pour seul objectif de maximiser le nombre total de paramètres n'est pas optimale en termes de précision (Figure 1a). Nous avons observé que lorsque l'on réduit le réseau de manière unidimensionnelle, la précision est mieux préservée lorsqu'on diminue la profondeur plutôt que la largeur. Nous proposons une méthode de mise à l'échelle consistant à réduire la profondeur du réseau de manière à maximiser la largeur de celui-ci. En perspectives, nous souhaitons raffiner cette méthode et l'utiliser dans le cadre d'un système d'automatisation du processus d'apprentissage incrémental.

Références

[1] E. Belouadah, A. Popescu, and I. Kanellos. A comprehensive study of class incremental learning algorithms for visual tasks. *Neural Networks*, 135 :38–54, 2021. ISSN 0893-6080.

[2] M. Delange, R. Aljundi, M. Masana, S. Parisot, X. Jia, A. Leonardis, G. Slabaugh, and T. Tuytelaars. A continual learning survey : Defying forgetting in classification tasks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1–1, 2021.

[3] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet : A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. IEEE, 2009.

[4] T. Elsken, J. H. Metzen, and F. Hutter. Neural architecture search : A survey. *The Journal of Machine Learning Research*, 20(1) :1997–2017, 2019.

[5] S. Esser, J. L. McKinstry, D. Bablani, R. Appuswamy, and D. S. Modha. Learned step size quantization, Aug. 26 2021. US Patent App. 16/796,397.

[6] S. K. Esser, J. L. McKinstry, D. Bablani, R. Appuswamy, and D. S. Modha. Learned step size quantization. *arXiv preprint arXiv :1902.08153*, 2019.

[7] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016.

[8] S. Hou, X. Pan, C. C. Loy, Z. Wang, and D. Lin. Learning a Unified Classifier Incrementally via Rebalancing. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 831–839, Long Beach, CA, USA, June 2019. IEEE. ISBN 978-1-72813-293-8.

[9] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam. Mobilenets : Efficient convolutional neural networks for mobile vision applications. *CoRR*, abs/1704.04861, 2017.

[10] S. Huang, V. François-Lavet, and G. Rabusseau. Neural architecture search for class-incremental learning. *ArXiv*, abs/1909.06686, 2019.

[11] S. Huang, V. Francois-Lavet, and G. Rabusseau. Understanding capacity saturation in incremental learning. *Proceedings of the Canadian Conference on Artificial Intelligence*, 6 2021.

[12] Y. Li, R. Gong, X. Tan, Y. Yang, P. Hu, Q. Zhang, F. Yu, W. Wang, and S. Gu. Brecq : Pushing the limit of post-training quantization by block reconstruction. In *International Conference on Learning Representations*, 2021.

[13] N. Ma, X. Zhang, H.-T. Zheng, and J. Sun. Shufflenet v2 : Practical guidelines for efficient cnn architecture design. In *Proceedings of the European Conference on Computer Vision (ECCV)*, September 2018.

[14] M. McCloskey and N. J. Cohen. Catastrophic interference in connectionist networks : The sequential learning problem. In *Psychology of learning and motivation*, volume 24, pages 109–165. Elsevier, 1989.

[15] S. Mirzadeh, A. Chaudhry, H. Hu, R. Pascanu, D. Görür, and M. Farajtabar. Wide neural networks forget less catastrophically. *CoRR*, abs/2110.11526, 2021.

[16] S. Mirzadeh, A. Chaudhry, D. Yin, T. Nguyen, R. Pascanu, D. Görür, and M. Farajtabar. Architecture matters in continual learning. *CoRR*, abs/2202.00275, 2022.

[17] P. Molchanov, S. Tyree, T. Karras, T. Aila, and J. Kautz. Pruning convolutional neural networks for resource efficient inference. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*, 2017.

[18] M. Sandler, A. G. Howard, M. Zhu, A. Zhmoginov, and L. Chen. Mobilenetv2 : Inverted residuals and linear bottlenecks. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pages 4510–4520. Computer Vision Foundation / IEEE Computer Society, 2018.

[19] M. Tan and Q. V. Le. Efficientnet : Rethinking model scaling for convolutional neural networks. In K. Chaudhuri and R. Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pages 6105–6114. PMLR, 2019.

[20] M. Tan, B. Chen, R. Pang, V. Vasudevan, M. Sandler, A. Howard, and Q. V. Le. MnasNet : Platform-Aware Neural Architecture Search for Mobile. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2815–2823, Long Beach, CA, USA, June 2019. IEEE. ISBN 978-1-72813-293-8.