La parcimonie des réseaux de neurones peut améliorer leur confidentialité

Antoine GONON*[1] Léon ZHENG*[1, 2] Clément LALANNE[1] Quoc-Tung LE[1] Guillaume LAUGA†[1] Can POULIQUEN†[1]

[1]Univ Lyon, EnsL, UCBL, CNRS, Inria, LIP, F-69342, LYON Cedex 07, France

[2]valeo.ai, Paris, France

Résumé – L'utilisation des réseaux de neurones parcimonieux, qui par définition ont peu de paramètres non nuls, est d'abord motivée par l'obtention de performances similaires au réseau dense d'origine tout en économisant des ressources. Cet article montre par ailleurs empiriquement que la parcimonie peut améliorer la confidentialité des données utilisées à l'entraînement des réseaux.

Abstract – Sparse neural networks are mainly motivated by the ability to obtain the same accuracy as the original dense network while being more resource efficient, as they have by definition only a few nonzero parameters. This article shows empirically that sparsity can also improve the privacy of the data used to train the networks.

1 Introduction

Les réseaux de neurones profonds constituent l'état de l'art dans de nombreux problèmes d'apprentissage. En pratique, il est possible d'ajuster les paramètres du réseau considéré afin de parfaitement interpoler les données disponibles [21]. Cette situation de *sur-apprentissage* est intéressante car les modèles ont de bonnes performances dans ce régime [3]. Néanmoins, il présente un risque de confidentialité puisque le modèle mémorise des informations sur les données, au point de les interpoler. Parmi ces informations, certaines sont peut-être confidentielles, et il se pose la question de savoir lesquelles peuvent être retrouvées à partir de la seule connaissance du modèle appris.

Pour détecter une situation de sur-apprentissage, un indicateur est donné par le ratio nombre de paramètres sur nombre de données : plus il y a de paramètres, plus le modèle peut interpoler les données. Afin de contrôler la capacité du modèle à sur-apprendre, et donc à mémoriser des informations confidentielles, cet article étudie le rôle du nombre de paramètres non nuls utilisés. Existe-t-il un bon compromis entre précision du modèle et confidentialité en ajustant uniquement la parcimonie (nombre de paramètres non nuls) des réseaux de neurones ?

Via un type d'attaque nommé « Membership Inference Attack » (MIA), il est possible d'inférer l'appartenance de données au jeu d'entraînement [18]. Cette attaque ne requiert qu'un accès boîte-noire au modèle visé, et peut être problématique selon la confidentialité des données (médicales, etc.). Étant donné un réseau, comment réduire le risque d'une telle attaque, tout en préservant au mieux ses performances ?

De nombreuses procédures ont été proposées pour se défendre contre les MIAs [12]. Ici, l'approche étudiée consiste à diminuer le nombre de paramètres non nuls utilisés par le réseau afin de réduire sa capacité de mémorisation, en préservant autant que possible les performances. [1]

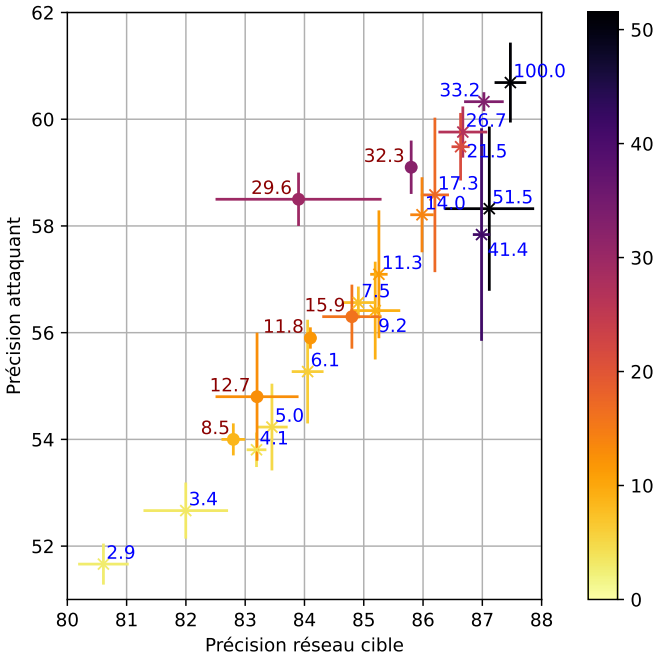

FIGURE 1 : Résultats moyens et écart-types obtenus pour la précision et la défense d'un réseau cible. Le pourcentage de poids non nuls est indiqué en bleu pour IMP ($*$ p%), en rouge pour Butterfly ($\bullet$ p%). La couleur des points indique le niveau de parcimonie (en noir de 50 à 100%).

Approches similaires. Les liens entre parcimonie des réseaux de neurones et confidentialité ont déjà été partiellement explorés, mais, à notre connaissance, il n'a pas encore été mis en évidence que la parcimonie permet d'améliorer la confidentialité *sans modification supplémentaire* de l'algorithme d'entraînement. Cet article propose ainsi d'étudier expérimentalement les compromis entre les performances d'un réseau de neurones parcimonieux et sa robustesse aux attaques de confidentialité, en ajustant uniquement son degré de parcimonie. Un positionnement est réalisé en section 4.

*,† : Contributions égales. Le travail est en partie soutenu par les projets AllegroAssai ANR-19-CHIA-0009, NuSCAP ANR-20-CE48-0014, SeqALO ANR-20-CHIA-0020-01, MOMIGS du GdR ISIS et la CIFRE N°2020/1643. Les auteurs remercient le Centre Blaise Pascal pour les moyens de calcul. La plateforme exploite SIDUS [16] développée par Emmanuel Quemener.

[1]Empiriquement, à un nombre de paramètres donné, il vaut mieux considérer un gros réseau avec beaucoup de poids mis à zéro qu'un petit réseau dense, le premier pouvant atteindre la performance du gros réseau dense [8].

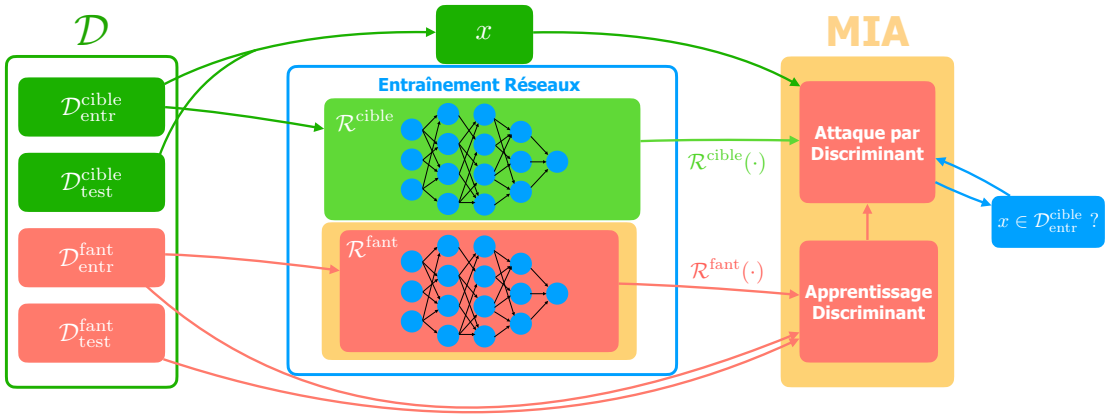

FIGURE 2 : Les expériences réalisées obéissent au même schéma général représenté ici : deux réseaux sont entraînés de la même manière sur $\mathcal{D}^{\text{cible}}_{\text{entr}}$ et $\mathcal{D}^{\text{fant}}_{\text{entr}}$ respectivement. $\mathcal{R}^{\text{fant}}$, $\mathcal{D}^{\text{fant}}_{\text{entr}}$ et $\mathcal{D}^{\text{fant}}_{\text{test}}$ sont ensuite utilisés pour entraîner un discriminant qui va attaquer $\mathcal{R}^{\text{cible}}$ en identifiant l'appartenance ou non de x à $\mathcal{D}^{\text{cible}}_{\text{entr}}$.

Contributions et résultats. Cet article étudie deux types de parcimonie détaillées en section 3 : l'une sans contrainte sur le support des poids non nuls, dite *non structurée* [8], et l'autre *structurée* [15, 6]. Les résultats en section 4 corroborent l'hypothèse que la parcimonie permet d'améliorer la défense contre une MIA tout en gardant des performances comparables sur la tâche d'apprentissage, voir Figure 1.

1. C'est le cas de la parcimonie non structurée, déjà explorée dans [20], mais en arrivant à des conclusions opposées. Une comparaison est faite en section 4.

2. Pour la parcimonie structurée, les expériences suggèrent un compromis confidentialité/performance similaire au cas non structuré. Ceci est remarquable car la structure est fixée indépendamment des données. De surcroît, cette dernière a l'avantage de permettre une implémentation efficace de la multiplication matrice-vecteur.

Limites Néanmoins, les écart-types observés nuancent les résultats et suggèrent qu'il est nécessaire de réaliser des expériences à plus grande échelle avant de pouvoir confirmer cette tendance. En outre, les expériences se contentent de montrer que la parcimonie peut améliorer la confidentialité, sans la comparer à d'autres mécanismes de défenses existants [12]. Enfin, et surtout, l'intérêt de la parcimonie doit être confirmée en la confrontant à des attaques plus pertinentes ayant de faibles taux de faux positifs [4] (mais plus coûteuses à mettre en place).

Plan La section 2 introduit le modèle d'attaque par réseau fantôme utilisé pour les expériences. La section 3 décrit les types de parcimonie utilisés pour se défendre contre les MIAs. Les expériences sont présentées en section 4.

2 Attaque par réseau fantôme

Étant donné un jeu de données $\mathcal{D}^{\text{cible}}$ et un réseau de neurones $\mathcal{R}^{\text{cible}}$ entraîné sur un sous-ensemble d'entraînement $\mathcal{D}^{\text{cible}}_{\text{entr}} \subset \mathcal{D}^{\text{cible}}$, une MIA consiste à retrouver la fonction d'appartenance

$$a_{\text{cible}} : x \in \mathcal{D}^{\text{cible}} \mapsto \begin{cases} 1 & \text{si } x \in \mathcal{D}^{\text{cible}}_{\text{entr}}, \\ 0 & \text{sinon,} \end{cases}$$

en ayant seulement un accès *boîte noire* à $x \mapsto \mathcal{R}^{\text{cible}}(x)$. La plupart des attaques cherchent à mesurer la confiance du modèle en ses prédictions réalisées localement autour de x [12]. Si la mesure de confiance est suffisamment élevée, alors l'attaquant répond oui à la question d'appartenance.

En pratique, l'attaque la plus performante [12] consiste à entraîner un modèle *discriminant* qui prend une décision en fonction d'informations locales sur $\mathcal{R}^{\text{cible}}$ autour de x. Ce discriminant est entraîné à partir d'un ou plusieurs réseau(x) *fantôme(s)* [12], comme expliqué ci-dessous (voir aussi Figure 2). Les expériences de la section 4 ne considèrent que le cas le moins coûteux où un seul réseau fantôme a été entraîné.

Réseau Fantôme. Supposons que l'attaquant ait accès à un jeu de données $\mathcal{D}^{\text{fant}}$ issu de la même distribution que $\mathcal{D}^{\text{cible}}$. Il entraîne alors son propre réseau fantôme $\mathcal{R}^{\text{fant}}$ sur un sous-ensemble $\mathcal{D}^{\text{fant}}_{\text{entr}} \subset \mathcal{D}^{\text{fant}}$ des données qu'il possède. Idéalement, $\mathcal{R}^{\text{fant}}$ est entraîné dans les mêmes conditions que $\mathcal{R}^{\text{cible}}$ (même structure et même algorithme d'optimisation). L'attaquant a alors un triplet $(\mathcal{R}^{\text{fant}}, \mathcal{D}^{\text{fant}}, \mathcal{D}^{\text{fant}}_{\text{entr}})$ ayant des similarités avec le triplet $(\mathcal{R}^{\text{cible}}, \mathcal{D}^{\text{cible}}, \mathcal{D}^{\text{cible}}_{\text{entr}})$, et la connaissance de la fonction d'appartenance $a_{\text{fant}} := a_{\mathcal{D}^{\text{fant}}_{\text{entr}}, \mathcal{D}^{\text{fant}}}$.

Discriminant. L'attaquant peut ensuite entraîner un discriminant afin d'approcher a_{fant} à partir du seul accès boîte noire de $\mathcal{R}^{\text{fant}}$. Ce discriminant peut alors servir à approcher a_{cible} à partir du seul accès boîte noire de $\mathcal{R}^{\text{cible}}$. Le modèle pour le discriminant peut être n'importe quel classificateur classique (régression logistique, réseau de neurones, etc.) [12].

3 Défense et parcimonie des réseaux

L'entraînement de réseaux de neurones parcimonieux est d'abord motivé par des besoins de frugalité en ressources [11] (mémoire, temps d'inférence, temps d'entraînement, etc.).

Ici, l'hypothèse suivante est étudiée : la parcimonie peut limiter le sur-apprentissage, et ainsi limiter la capacité du modèle à mémoriser des informations confidentielles sur les données qu'il a vues. Un réseau parfaitement confidentiel n'a rien appris de ses données et n'a donc pas d'intérêt en pratique. Un compromis entre confidentialité et précision du réseau est à réaliser en fonction de la tâche considérée.

3.1 Parcimonie non structurée via IMP

Dans le premier cas, aucune structure spécifique n'est imposée sur l'ensemble des poids non nuls. Les poids nuls sont

sélectionnés par un processus itératif d'élagage par amplitude (« Iterative Magnitude Pruning », IMP) [8] qui consiste à :

(i) entraîner un réseau de manière usuelle,

(ii) élaguer (mettre à zéro) $p\%$ des poids ayant l'amplitude la plus faible,

(iii) ajuster les poids restants en ré-entraînant le réseau (les poids ayant été élagués sont masqués et ne sont plus mis à jour), puis revenir à l'étape (ii) tant que la parcimonie souhaitée n'est pas atteinte.

Cette procédure permet de trouver des sous-réseaux ayant empiriquement de bonnes propriétés statistiques [8, 9].

3.2 Parcimonie structurée butterfly

Dans le second cas, la parcimonie dite « butterfly » est structurée : les matrices de poids des couches du réseau de neurones sont contraintes à s'écrire comme un produit de matrices creuses ayant des supports bien spécifiques [15, 6], voir Figure 3. Ce type de parcimonie est particulièrement intéressant car toute matrice de taille $N \times N$ ayant une telle factorisation a une complexité théorique pour la multiplication matrice-vecteur sous-quadratique, par exemple $\mathcal{O}(N \log N)$ pour certains choix de supports [7], contre $\mathcal{O}(N^2)$ en général.

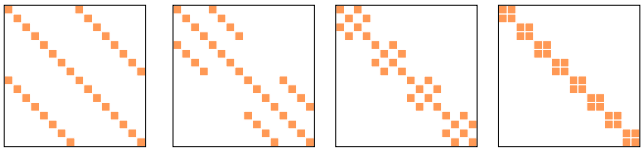

FIGURE 3 : Exemple de choix de supports imposés aux facteurs creux dans une décomposition butterfly.

Pour imposer la structure butterfly dans un réseau de neurones, les matrices de poids $\mathbf{W}$ sont paramétrisées sous la forme $\mathbf{W} = \mathbf{X}^{(1)} \dots \mathbf{X}^{(L)}$, et seuls les coefficients non nuls des facteurs $\mathbf{X}^{(1)}, \dots, \mathbf{X}^{(L)}$ sont ajustés pour minimiser la fonction de coût au cours de l'entraînement. L'initialisation de ces coefficients au début de l'entraînement est aléatoire. Dans le cas d'une couche de convolution, la matrice $\mathbf{W}$ pour laquelle on impose une telle structure correspond à la concaténation des noyaux de convolution [15]. Les réseaux butterfly obtenus atteignent des performances empiriques comparables à un réseau dense sur la classification d'images [6, 15].

4 Résultats expérimentaux

Tous les hyperparamètres (y compris l'architecture du discriminant) ont été déterminés suite à une recherche sur grille où l'aléa a été moyenné sur trois expériences.

Données. Les expériences sont réalisées sur le problème de classification CIFAR-10 (60000 images $32 \times 32 \times 3$, 10 classes). Les données sont aléatoirement (uniformément) partitionnées en 4 sous-ensembles $\mathcal{D}^{\text{cible}}_{\text{entr}}, \mathcal{D}^{\text{cible}}_{\text{test}}, \mathcal{D}^{\text{fant}}_{\text{entr}}, \mathcal{D}^{\text{fant}}_{\text{test}}$ de 15000 images, respectivement utilisés pour entraîner et tester les réseaux cibles et fantômes. Les questions d'appartenances se posent pour $\mathcal{D}^{\text{cible}} := \mathcal{D}^{\text{cible}}_{\text{entr}} \cup \mathcal{D}^{\text{cible}}_{\text{test}}$ et $\mathcal{D}^{\text{fant}} := \mathcal{D}^{\text{fant}}_{\text{entr}} \cup \mathcal{D}^{\text{fant}}_{\text{test}}$. Pour le réseau cible et fantôme, parmi leurs 15000 données d'entraînement, 1000 sont choisies aléatoirement et fixées pour toutes nos expériences comme ensemble de validation (utilisé pour choisir les hyperparamètres, et pour le critère d'arrêt).

TABLE 1 : Hyperparamètres pour l'entraînement des réseaux cibles et fantômes.

Réseau	% de paramètres	Pas initial	Weight decay
ResNet-20 dense	100 %	0.03	0.005
Butterfly ($S = 1, L = 2$)	32.3 %	0.3	0.0005
Butterfly ($S = 1, L = 3$)	29.6 %	0.3	0.0001
Butterfly ($S = 2, L = 2$)	15.9 %	0.3	0.0005
Butterfly ($S = 2, L = 3$)	12.9 %	0.1	0.001
Butterfly ($S = 3, L = 2$)	11.8 %	0.3	0.0005
Butterfly ($S = 3, L = 3$)	8.5 %	0.1	0.001
IMP avec k élagages	$\simeq 100 \times (0.8)^k\%$	0.03	0.005

Entraînement des réseaux cibles et fantômes. Les réseaux cibles et fantômes sont des ResNet-20 [10] (272474 paramètres) entraînés pour minimiser l'entropie croisée par descente de gradient stochastique (avec momentum 0.9 et sans accélération Nesterov) sur leur jeu d'entraînement respectif pendant 300 époques, avec des batchs de taille 256. Les données sont augmentées avec retournement horizontal aléatoire et recadrage aléatoire. Le pas d'apprentissage initial est divisé par 10 au bout de 150 époques, puis à nouveau par 10 au bout de 225 époques. Les poids des réseaux de neurones sont initialisés avec la méthode standard par défaut sur Pytorch selon une loi uniforme sur $(-1/\sqrt{n}, 1/\sqrt{n})$ où $n =$ dimension entrée pour une couche linéaire, et $n =$ dimension entrée $\times$ largeur noyau $\times$ hauteur noyau pour une convolution. Les valeurs du pas d'apprentissage initial et du « weight decay » sont reportées dans la Table 1. Elles permettent de reproduire les résultats de [10] lorsque les réseaux sont entraînés sur l'ensemble du jeu d'entraînement de CIFAR-10 avec ResNet-20.

Pour IMP, 24 élagages et ré-ajustements des poids sont réalisés. Chaque ré-ajustement consiste en un entraînement comme ci-dessus (300 époques, etc.). Avant chaque élagage, les poids sont rembobinés à leurs valeurs associées à l'époque ayant la précision maximale sur le jeu de validation lors des dernières 300 époques.

Pour l'entraînement de ResNet-20 avec structure butterfly, les matrices de poids originelles de certaines couches de convolution sont substituées par des matrices ayant une factorisation, avec un nombre $L = 2, 3$ de facteurs, suivant une chaîne monotone minimisant le nombre de paramètres dans la factorisation [15]. Les couches substituées sont celle des $S = 1, 2, 3$ derniers segments de ResNet-20.

Entraînement du discriminant. Un discriminant prend en entrée la classe i de x, la prédiction $\mathcal{R}(x)$ réalisée par un réseau $\mathcal{R}$ (cible ou fantôme), ainsi que $\frac{1}{\epsilon}\mathbb{E}\left(|\mathcal{R}(x) - \mathcal{R}(x + \epsilon\mathcal{N})|\right)$ ($\epsilon = 0,001$ et $\mathcal{N}$ un vecteur gaussien indépendant centré réduit) encodant des informations locales du premier ordre sur $\mathcal{R}$ autour de x. La moyenne est réalisée sur 5 échantillons. Ce sont les entrées les plus communément utilisées pour le discriminant dans la littérature [12]. Pour chaque couple de réseaux ($\mathcal{R}^{\text{cible}}$,$\mathcal{R}^{\text{fant}}$), sont entraînés trois discriminants (perceptrons) à respectivement 1, 2, 3 couche(s) cachée(s) avec respectivement 30, 30, 100 neurones sur chaque couche cachée. L'entropie croisée est minimisée avec Adam, sans weight decay avec des pas dans $\{0.01, 0.001, 0.0001\}$ sur 80 époques.

Précision du réseau et de l'attaquant La *précision* d'un réseau est le pourcentage de données dont la classe est celle prédite avec le plus de probabilité par le réseau. Dans notre cas, il y a autant de données vues que non vues par le réseau (cible

ou fantôme) durant l'entraînement. Idéalement, le discriminant ne peut pas faire mieux que deviner au hasard, correspondant à une précision pour l'attaque de 50%. Une précision plus élevée traduit une diminution de la confidentialité des données.

Résultats Les réseaux cibles et fantômes denses atteignent en moyenne 87.5% de précision sur l'ensemble test. Cette précision diminue avec la parcimonie, voir la Figure 1. Un gain (ou perte) en précision est significatif si l'intervalle donné par la moyenne plus ou moins l'écart-type est disjoint de l'intervalle correspondant au réseau dense entraîné. De manière significative, une diminution de la précision de l'attaquant est observée pour une proportion de poids entre 0% et 17.3%, ainsi que pour 41.4% et 51.5%. En moyenne, de 3.4% à 17.3%, une perte relative de $p\%$ en précision du réseau, par rapport au réseau dense entraîné, mène à une perte *relative* de $0.87 \times p\%$ en précision de l'attaquant : $0.87 \simeq \frac{|\text{préc. attaquant} - \text{préc. attaquant dense}|}{\text{préc. attaquant dense}} \frac{\text{préc. réseau dense}}{|\text{préc. réseau} - \text{préc. réseau dense}|}$.

Ces expériences suggèrent (i) que la parcimonie est un levier permettant d'améliorer la confidentialité des données d'entraînement en échange d'une diminution de la précision du réseau, et (ii) que ce compromis confidentialité/précision est le même pour la parcimonie non structurée via IMP et la parcimonie structurée butterfly. Ces expériences confirment l'intérêt de la parcimonie pour les questions de confidentialité. Cela ouvre la voie à la comparaison entre la parcimonie et les mécanismes de défenses existants.

Néanmoins, les écarts types observés nuancent les résultats et suggèrent que des expériences à plus grandes échelles sont nécessaires pour confirmer ces tendances.

Positionnement Les résultats expérimentaux de [20] suggèrent à l'inverse que l'entraînement d'un réseau avec régularisation parcimonieuse par IMP *dégrade* la confidentialité. Mais ces résultats n'ont pas été moyennés sur plusieurs expériences pour diminuer la variabilité de l'aléa. Les expériences de [20] sont réalisées sur un modèle ayant 40 fois plus de poids que ResNet-20, et pour des proportions de poids non nuls $> 50\%$. Étant donné les écart-types observés en Figure 1 pour des niveaux de parcimonie $> 20\%$ sur ResNet-20, il convient de rester prudent sur l'interprétabilité des résultats de [20].

L'article [19] fixe un niveau de parcimonie, et cherche les paramètres qui minimisent la fonction de perte du problème d'apprentissage, pénalisé par la plus grande précision d'attaque MIA atteignable contre ces paramètres. Néanmoins, ce terme de pénalisation n'est en général pas calculable explicitement, et difficile à minimiser. Sans comparaison avec le cas non pénalisé [19], on ne peut conclure sur la nécessité de cette pénalisation. Ici, la confidentialité est améliorée sans cette pénalisation. De plus, [19] ne présente pas quelle confidentialité est atteinte pour chaque niveau de parcimonie, mais seulement au niveau de parcimonie ayant la plus petite fonction de perte pénalisée. La Figure 1 montre quel est l'effet de la parcimonie sur la confidentialité.

Enfin, il a été observé qu'imposer la parcimonie durant l'entraînement des réseaux de neurones avec DP-SGD (« Differentialy Private Stochastic Gradient Descent ») [1, 2] améliore leurs performances, à garanties égales de « Differential Privacy » (donnant des garanties fortes de confidentialité) [13, 2]. L'utilisation de la DP-SGD a cependant un coût en performances et en ressources [17, 14] prohibitif pour des expériences à grandes échelles. Ici, l'amélioration de la confidentialité se fait à un coût moindre (en performance, en ressources car SGD est utilisé) mais n'apporte pas de garantie théorique.

5 Conclusion

Si la parcimonie est intéressante pour économiser des ressources (temps, mémoire), les résultats suggèrent qu'elle peut aussi améliorer la confidentialité des données d'entraînement, avec un coût relativement faible sur les performances des réseaux. C'est en particulier le cas pour la parcimonie structurée butterfly, à notre connaissance jamais explorée dans ce contexte dans la littérature.

Pour confirmer cet intérêt potentiel, il faudrait d'abord confronter la parcimonie à des attaques plus pertinentes [5] (et plus coûteuses à mettre en place), sur un ensemble plus riche de modèles et de données. De plus, une comparaison à des mécanismes de défense identifiés comme tels permettrait de voir le gain relatif de confidentialité induit par la parcimonie, comparé à ces derniers.

Références

[1] M. Abadi, A. Chu, I. Goodfellow, H. B. McMahan, I. Mironov, K. Talwar et Li. Zhang : Deep learning with differential privacy. *In SIGSAC*, 2016.

[2] K. Adamczewski et M. Park : Differential privacy meets neural network pruning. *Preprint*, 2023.

[3] M. Belkin, D. Hsu, S. Ma et S. Mandal : Reconciling modern machine-learning practice and the classical bias-variance trade-off. *National Academy of Sciences USA*, 2019.

[4] N. Carlini, C. Liu, U. Erlingsson, J. Kos et D. Song : The secret sharer : Evaluating and testing unintended memorization in neural networks. *In USENIX*, 2019.

[5] Nicholas Carlini, Steve Chien, Milad Nasr, Shuang Song, Andreas Terzis et Florian Tramèr : Membership inference attacks from first principles. *In 43rd IEEE Symposium on Security and Privacy, SP 2022, San Francisco, CA, USA, May 22-26, 2022*, pages 1897–1914. IEEE, 2022.

[6] T. Dao, B. Chen, N. S. Sohoni, A. Desai, M. Poli, J. Grogan, A. Liu, A. Rao, A. Rudra et C. Ré : Monarch : Expressive structured matrices for efficient and accurate training. *In ICML*, 2022.

[7] T. Dao, A. Gu, M. Eichhorn, A. Rudra et C. Ré : Learning fast algorithms for linear transforms using butterfly factorizations. *In ICML*, 2019.

[8] J. Frankle et M. Carbin : The lottery ticket hypothesis : Finding sparse, trainable neural networks. *In ICLR*, 2019.

[9] J. Frankle, G. K. Dziugaite, D. Roy et M. Carbin : Pruning neural networks at initialization : Why are we missing the mark ? *In ICLR*, 2021.

[10] K. He, X. Zhang, Sh. Ren et J. Sun : Deep residual learning for image recognition. *In CVPR*. IEEE, 2016.

[11] T. Hoefler, D. Alistarh, T. Ben-Nun, N. Dryden et A. Peste : Sparsity in deep learning : Pruning and growth for efficient inference and training in neural networks. *Journal of Machine Learning Research*, 2021.

[12] H. Hu, Z. Salcic, L. Sun, G. Dobbie, P. S. Yu et X. Zhang : Membership inference attacks on machine learning : A survey. *ACM Computing Surveys*, 2022.

[13] Y. Huang, Y. Su, S. Ravi, Z. Song, S. Arora et K. Li : Privacy-preserving learning via deep net pruning. *Preprint*, 2020.

[14] C. Lalanne, A. Garivier et R. Gribonval : On the statistical complexity of estimation and testing under privacy constraints. *Preprint*, 2022.

[15] R. Lin, J. Ran, K. H. Chiu, G. Chesi et N. Wong : Deformable butterfly : A highly structured and sparse linear transform. *In NeurIPS*, 2021.

[16] E. Quemener et M. Corvellec : SIDUS—the Solution for Extreme Deduplication of an Operating System. *Linux Journal*, 2013.

[17] T. Sander, P. Stock et A. Sablayrolles : Tan without a burn : Scaling laws of dp-sgd. *Preprint*, 2022.

[18] R. Shokri, M. Stronati, C. Song et V. Shmatikov : Membership inference attacks against machine learning models. *In SP*. IEEE, 2017.

[19] Y. Wang, C. Wang, Z. Wang, S. Zhou, H. Liu, J. Bi, C. Ding et S. Rajasekaran : Against membership inference attack : Pruning is all you need. *IJCAI*, 2021.

[20] X. Yuan et L. Zhang : Membership inference attacks and defenses in neural network pruning. *In USENIX*. IEEE, 2022.

[21] C. Zhang, S. Bengio, M. Hardt, B. Recht et O. Vinyals : Understanding deep learning (still) requires rethinking generalization. *ACM*, 2021.