Ouvrons la boîte noire : reconstruction de réseaux de neurones parcimonieux par échantillonnage

Valérie CASTIN[1] Rémi GRIBONVAL[2]

[1]CNRS, Ecole Normale Supérieure PSL, 45 rue d'Ulm, 75005 Paris, France

[2]Inria, CNRS, ENS de Lyon, Université Claude Bernard Lyon 1, LIP, UMR 5668, 69342, Lyon cedex 07, France

Résumé – Dans le contexte de la compression de réseaux de neurones, la distillation consiste à entraîner un petit réseau *élève* (si possible parcimonieux) à partir d'échantillons générés par un réseau *enseignant*. Malgré le succès empirique de ces approches on connaît mal leurs conditions de succès. En guise de preuve de concept nous introduisons une nouvelle famille de réseaux parcimonieux structurés dits *perceptrons multiarbre*, et un algorithme d'estimation des paramètres de tels réseaux à partir d'échantillons de la fonction correspondante et de sa Jacobienne. L'algorithme en question n'est pas basé sur la descente de gradient.

Abstract – In the context of neural network compression by distillation, one wishes to estimate the parameters of a (small and possibly sparse) *student* network given samples of a *teacher* network. Despite empirical successes, little is known on the conditions of success of such approaches. As a proof of concept, we introduce a new family of sparse structured ReLU networks –called multitree perceptrons– endowed with a provably good algorithm to learn their parameters from samples of the corresponding function and its Jacobian. The algorithm completely bypasses usual gradient descent.

1 Introduction

L'apprentissage par réseaux de neurones profonds fournit des résultats spectaculaires dans une grande variété de domaines. Cependant, les réseaux profonds actuels ont énormément de paramètres (de l'ordre du milliard, voire du trillion), ce qui nécessite de collecter des jeux de données d'entraînement gigantesques et rend ces modèles très coûteux à entraîner et à stocker. Une façon de contourner ce problème est de considérer des réseaux profonds *parcimonieux*, c'est-à-dire ayant peu de paramètres non nuls, et de procéder par distillation [7], c'est-à-dire d'entraîner un petit réseau parcimonieux en ayant accès à un grand réseau entraîné.

Un enjeu est de comprendre les conditions de succès des méthodes de distillation, et notamment de maîtriser le nombre d'appels à la fonction correspondant au grand réseau pour estimer les paramètres du petit. En guise de preuve de concept cet article s'intéresse à une famille particulière de réseaux parcimonieux, que nous nommerons perceptrons multiarbre (en anglais multitree perceptron – MTP). Pour différents types de MTP, nous étudions la reconstruction des paramètres à partir d'échantillons de la réalisation du réseau et de sa Jacobienne.

Soit un MTP d'activation $\rho(t) = \mathrm{ReLU}(t) = \max(t, 0)$, avec des matrices de poids $W_1, \dots, W_L$ inconnues mais de supports connus, et des vecteurs de biais $b_1, \dots, b_L$ inconnus. On connaît donc son architecture, mais pas la valeur de ses paramètres. Notons f sa réalisation :

$$f(x) = W_L \rho(\dots W_2 \rho(W_1 x + b_1) + b_2 \dots) + b_L, \qquad (1)$$

et supposons que l'on peut échantillonner librement f et sa Jacobienne ∂f. Comment déterminer des poids $\hat{W}_1, \dots, \hat{W}_L$ et des biais $\hat{b}_1, \dots, \hat{b}_L$ dont la réalisation associée $\hat{f}$ est égale à f, avec des garanties théoriques de reconstruction ?

Nos résultats pourraient servir de base à des travaux futurs pour élaborer des méthodes de distillation *sans descente de gradient*, l'idée étant d'apprendre un MTP à partir d'échantillons de la réalisation et de la Jacobienne d'un réseau plein. Au-delà de la distillation, ce problème est relié à la robustesse aux attaques et à la protection de données sensibles. En effet, retrouver les paramètres d'un réseau peut aider à construire des attaques adversariales [3] ou fournir des informations sur les données ayant servi à entraîner ce réseau [6].

2 Perceptrons multi-arbres ReLU

Pour fixer les notations, nous formalisons la notion de perceptron multi-arbre (MTP), discutons ses liens avec la structure papillon et les algorithmes de factorisation associés, et rappelons quelques propriétés utiles des réseaux ReLU.

Perceptron multiarbre. Un perceptron multiarbre (MTP) est un perceptron multicouche (MLP) parcimonieux dont le graphe acyclique orienté sous-jacent [5] est un multiarbre (en anglais multitree [4]), c'est-à-dire que tout couple de sommets de ce graphe est relié par au plus un chemin orienté. Cette propriété dépend uniquement du *support* des matrices de poids $(W_1, \dots, W_L)$ du MTP, et on dira aussi dans ce cas que $(W_1, \dots, W_L)$ est un MTP. Si w et x sont deux vecteurs regroupant tous les coefficients non nuls de $(W_1, \dots, W_L)$ et $X \coloneqq W_L \dots W_1$ respectivement, dans un ordre fixé (arbitraire), alors la propriété multiarbre se traduit par une dépendance linéaire entre $\log x$ et $\log w$, où $\log$ est la détermination complexe du logarithme ayant pour ligne de coupure $-\mathrm{i}\mathbb{R}$: il existe une matrice P dépendant uniquement de l'ordre de vectorisation pour w et x telle que

$$\log x = P \log w. \qquad (2)$$

Une famille particulière de MTP est celle des MLP papillon, dont les matrices de poids ont une structure *papillon dyadique* (Fig. 1) : pour $1 \leq \ell \leq L$,

$$\operatorname{Supp} W_\ell \subseteq S_\ell \coloneqq I_{2^{L-\ell}} \otimes \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix} \otimes I_{2^{\ell-1}}. \qquad (3)$$

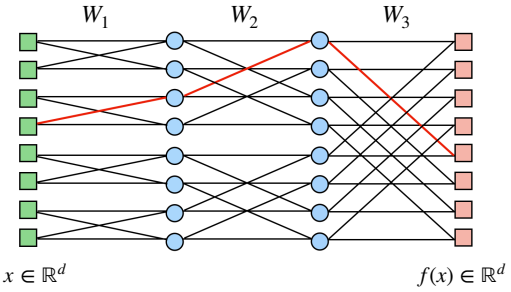

FIGURE 1 : Graphe d'un MLP papillon à trois couches. En rouge, le chemin $4 \to 5$.

Le support d'une matrice (les indices de ses entrées non nulles) est ici identifié à une matrice binaire et $\otimes$ est le produit de Kronecker. Ce type de structure apparaît dans les transformées rapides de type FFT [2], et des algorithmes efficaces permettent d'identifier les facteurs d'une matrice A quand elle admet une factorisation papillon $A = W_L \dots W_1$. On s'appuie sur ces algorithmes, munis de garanties [8, 12, 9], pour identifier des matrices de poids à partir de f et de sa Jacobienne ∂f.

Factorisation de produits MTP ou papillon. Notre méthode de reconstruction repose sur un algorithme de factorisation introduit dans [8]. Soient (W_1, W_2) les poids (inconnus) d'un MT[1] à une couche cachée *linéaire*. Etant donnés les supports (MTP) de W_1 et W_2 et le produit W_2W_1, on peut reconstruire [8] des matrices $(\hat{W}_1, \hat{W}_2)$ équivalentes à (W_1, W_2) modulo homothéties, i.e. telles qu'il existe une matrice diagonale inversible telle que $\hat{W}_1 = DW_1$ et $\hat{W}_2 = W_2D^{-1}$. En outre, ce résultat permet de factoriser itérativement un produit de matrices papillon [12], car les supports papillon $S_1, \dots, S_L$ définis dans l'équation (3) vérifient que pour tous $p \le \ell < q$, le couple $(S_q \dots S_{\ell+1}, S_\ell \dots S_p)$ est un MTP à deux couches. Cette factorisation itérative, modulo quelques adaptations, est applicable à des matrices arbitraires qu'elle approche de façon quasi-optimale par un produit de matrices papillon [9].

Jacobienne d'un MLP, activations et chemins. Pour tout vecteur d'entrée x et tout indice $1 \le \ell \le L$, notons $z_\ell(x) := W_\ell\rho(\dots W_2\rho(W_1x + b_1) + b_2 \dots) + b_\ell$ la pré-activation de la couche ℓ. La Jacobienne de la réalisation f définie en (1) s'écrit alors (voir par exemple [11])

$$\partial f(x) = W_L A_{L-1} W_{L-1} \dots A_1 W_1, \tag{4}$$

où $A_\ell = \mathrm{diag}(\mathbf{1}_{z_\ell(x)>0})$ est la matrice diagonale binaire d'activation des neurones de la couche cachée ℓ. On dit que le vecteur x active le neurone i de la couche cachée ℓ si $z_\ell(x)_i > 0$. On désigne par *chemin* de f tout chemin dans le graphe de f reliant un neurone d'entrée à un neurone de sortie (voir Figure 1). Un chemin est dit *activé* par un vecteur d'entrée x si x active tous les neurones cachés contenus dans ce chemin, et *activable* s'il existe un vecteur d'entrée qui active ce chemin. On note $i_0 \to i_1 \to \dots \to i_L$ le chemin passant par le neurone i_ℓ de la couche ℓ pour $0 \le \ell \le L$, où les couches 0 et L sont respectivement les couches d'entrée et de sortie. Une propriété clé des MTP est la suivante : *si f est un MTP, ce chemin est uniquement déterminé par i_0 et i_L, on le notera $i_0 \to i_L$.*

[1] La structure MTP à deux couches est équivalente aux hypothèses utilisées dans la factorisation à deux couches de [8], voir le Lemme 4.1.

Invariances de la réalisation. La réalisation f introduite dans l'équation (1) est invariante par la transformation

$$\tilde{W}_\ell \leftarrow D_\ell W_\ell D_{\ell-1}^{-1}, \quad \tilde{b}_\ell \leftarrow D_\ell b_\ell \quad \text{pour } 1 \le \ell \le L, \tag{5}$$

où $D_1, \dots, D_{L-1}$ sont des matrices diagonales strictement positives. Si $\tilde{W}_\ell, \tilde{b}_\ell$ se déduisent de la sorte des W_ℓ, b_ℓ pour $1 \le \ell \le L$, on dira que ces deux collections de paramètres sont *équivalentes modulo homothéties positives*. De plus, si le neurone i de la couche cachée ℓ n'est pas activable, mettre à zéro le biais $(b_\ell)_i$ et tous les coefficients de la ligne $L_i(W_\ell)$ et de la colonne $C_i(W_{\ell+1})$ ne change pas la réalisation f.

Identification de paramètres à invariances près. La reconstruction des paramètres de réseaux ReLU a été étudiée pour des MLP denses, i.e. sans structure de parcimonie, par [10]. Les auteurs proposent un algorithme permettant une reconstruction approchée des poids et biais, mais dont la complexité d'échantillonnage devient importante lorsque la profondeur du réseau à reconstruire augmente.

3 Aperçu des résultats

L'idée de notre approche, spécifique aux perceptrons multiarbre, est de tirer profit de la structure parcimonieuse des matrices de poids pour développer un algorithme de reconstruction plus efficace, *en supposant avoir accès à des échantillons de la Jacobienne du réseau* (ce qui est bien le cas dans un cadre de distillation, par auto-différentiation). Les échantillons de la Jacobienne de f sont en effet, pour des MTP, une version "masquée" du produit $W_L \dots W_1$. Nous adaptons les algorithmes de factorisation de [8, 12] au cas d'un produit masqué pour retrouver des poids $(\hat{W}_1, \dots, \hat{W}_L)$ proches de $(W_1, \dots, W_L)$. La reconstruction des biais fait l'objet d'une analyse séparée.

Contributions. Nous établissons les résultats suivants.

- Pour un MTP f *à une couche cachée*, les poids peuvent être retrouvés modulo homothéties positives *à partir de seulement deux échantillons de la Jacobienne* de f. Nous proposons un algorithme reconstruisant des paramètres $(\hat{W}_1, \hat{W}_2, \hat{b})$ induisant une réalisation égale à f (Section 4).
- Pour un réseau papillon f sans biais à L couches, nous proposons un algorithme qui, si suffisamment de chemins sont activés au cours de l'échantillonnage, reconstruit des paramètres $(\hat{W}_1, \dots, \hat{W}_L)$ induisant une réalisation égale à f (Section 5).
- Nous implémentons nos algorithmes sur différents MTP à deux couches, et sur des réseaux papillon jusqu'à la dimension 16. Notre code est en libre accès `https://github.com/vcastin/2025-reverse-engineering` [1].

4 Réseaux MTP à une couche cachée

Il est facile de montrer que les perceptrons multiarbre à une couche cachée sont caractérisés de la façon suivante.

Lemme 4.1. *Soient $W_1 \in \mathbb{R}^{d_{\text{in}} \times H}$ et $W_2 \in \mathbb{R}^{H \times d_{\text{out}}}$. On peut décomposer le produit W_2W_1 comme la somme de ses composantes de rang 1 : $W_2W_1 = \sum_{h=1}^{H} C_h(W_2)L_h(W_1)^\top$. Les poids (W_1, W_2) sont un MTP si et seulement les supports des composantes de rang 1 de W_2W_1 sont deux à deux disjoints.*

Dans la suite nous notons S_1 et S_2 les supports respectifs de W_1 et W_2, et $\Sigma_h \coloneqq C_h(S_2)L_h(S_1)^\top$ le support de la h-ème composante de rang 1 de W_2W_1. Des exemples de supports de MTP à deux couches sont donnés dans [8]. Une conséquence importante du Lemme 4.1 est la suivante, où l'on note $\odot$ le produit de Hadamard (coordonnée par coordonnée).

Lemme 4.2. *Soit f un MTP à une couche cachée, et h un neurone caché tel que $\Sigma_h \neq 0$. Un vecteur d'entrée $x \in \mathbb{R}^{d_{\text{in}}}$ active h si et seulement si $\partial f(x) \odot \Sigma_h \neq 0$: observer $\partial f(x)$ détermine quels neurones cachés sont activés par x.*

Démonstration. Puisque f est un MTP, les Σ_h sont deux à deux disjoints, d'après le Lemme 4.1. D'après (4) on a :

$$\partial f(x) = \sum_{h=1}^{H} \mathbf{1}_{\{x \text{ active } h\}} C_h(W_2)L_h(W_1)^\top, \qquad (6)$$

donc $\partial f(x) \odot \Sigma_h = \mathbf{1}_{\{x \text{ active } h\}} C_h(W_2)L_h(W_1)^\top$, qui est non nul si et seulement si x active h. □

S'ensuit une caractérisation des vecteurs d'entrée permettant d'identifier les paramètres d'un MTP à une couche cachée.

Théorème 4.1. *Soit f un MTP à une couche cachée de paramètres inconnus (W_1, W_2, b), tel que les supports de W_1 et W_2 sont connus. Soit $\mathcal{X} \subset \mathbb{R}^{d_{\text{in}}}$ un ensemble de vecteurs d'entrée. L'observation de $(f(x))_{x\in\mathcal{X}}$ et $(\partial f(x))_{x\in\mathcal{X}}$ permet la reconstruction de paramètres $(\hat{W}_1, \hat{W}_2, \hat{b})$ associés à une réalisation $\hat{f}$ égale à f si, et seulement si, les deux conditions suivantes sont satisfaites pour tout neurone caché $1 \leq h \leq H$:*

(i) il existe $x, y \in \mathcal{X}$ tels que x active h et y ne l'active pas,

(ii) il existe $x, y \in \mathcal{X}$ tels que seul le neurone h change d'état d'activation entre x et y.

De plus, la condition (i) est nécessaire et suffisante pour reconstruire des poids $(\hat{W}_1, \hat{W}_2)$ équivalents à (W_1, W_2) modulo homothéties positives.

On en déduit une procédure de reconstruction de paramètres $(\hat{W}_1, \hat{W}_2, \hat{b})$ associés à une réalisation $\hat{f}$ *égale à* f. On peut supposer sans perte de généralité que les supports Σ_h sont non nuls, car si $\Sigma_h = 0$, il suffit de fixer $C_h(\hat{W}_2) = 0$, et une valeur arbitraire pour $L_h(\hat{W}_1)$.

Etape 1 : construction d'échantillons $\mathcal{X}$ vérifiant (i). Soit x un vecteur aléatoire de coordonnées i.i.d. gaussiennes. On a presque sûrement $\langle L_h(W_1), x\rangle \neq 0$ pour tout h, et l'on peut multiplier x par 2 itérativement jusqu'à ce que x et $-x$ activent deux ensembles de neurones exactement complémentaires, ce qui se produit dès que $|\langle L_h(W_1), x\rangle| > |b_h|$ pour tout h. Alors, $\mathcal{X} = \{x, -x\}$ convient. En effet, si $\langle L_h(W_1), x\rangle > 0$, alors $\langle L_h(W_1), x\rangle + b_h > 0$ donc x active h, et $-\langle L_h(W_1), x\rangle + b_h < 0$ donc $-x$ n'active pas h. De même, si $\langle L_h(W_1), x\rangle < 0$, alors $-x$ active h et x n'active pas h.

Etape 2 : reconstruction et factorisation du produit W_2W_1. Soit $\mathcal{X} = \{x, -x\}$ l'ensemble d'échantillons construit ci-dessus. Par construction, les neurones activés par x sont exactement les neurones non activés par $-x$. On a donc $W_2W_1 = \partial f(x) + \partial f(-x)$ d'après l'équation (6). On calcule ainsi le produit W_2W_1 et on utilise ensuite l'algorithme de factorisation introduit dans [8] pour déterminer des matrices $\tilde{W}_1, \tilde{W}_2$ telles que $\tilde{W}_1 = DW_1$ et $\tilde{W}_2 = W_2D^{-1}$ pour D une matrice diagonale inversible inconnue.

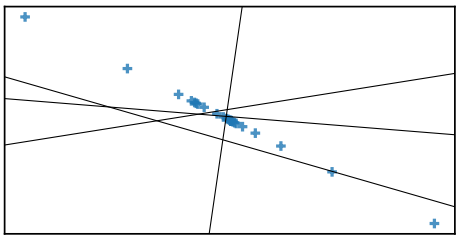

FIGURE 2 : Ensemble $\mathcal{X}$ exploité par notre algorithme pour un MTP à une couche cachée avec $d_{\text{in}}, H, d_{\text{out}} = 16$ (projection sur les deux premières coordonnées). Les points extrémaux correspondent à $\{x, -x\}$ vérifiant (i). Les droites marquent le lieu d'annulation de la préactivation de chaque neurone caché.

Etape 3 : redressement de $\tilde{W}_1, \tilde{W}_2$. Avec le x de la première étape, $\mathcal{X} = \{x, -x\}$. Pour tout $1 \leq h \leq H$, si $\partial f(x) \odot \Sigma_h \neq 0$, on pose[2] $\varepsilon_h \coloneqq \text{sgn}\langle \tilde{W}_1, x\rangle$, et si $\partial f(x) \odot \Sigma_h = 0$, alors $\varepsilon_h \coloneqq -\text{sgn}\langle \tilde{W}_1, x\rangle$. On définit $\hat{W}_1 \coloneqq \text{diag}(\varepsilon)\tilde{W}_1$ et $\hat{W}_2 = \tilde{W}_2 \,\text{diag}(\varepsilon)$. Par construction, $\hat{W}_1, \hat{W}_2$ sont équivalentes à W_1, W_2 modulo homothéties *positives* (cf (5)).

Etape 4 : reconstruction des biais. Pour reconstruire les biais $\hat{b}_h$, on cherche $H-1$ points $(y_h)_{1\leq h\leq H-1}$ appartenant au segment $[-x, x]$ et tels qu'il existe une permutation σ de $\{1, \ldots, h\}$ pour laquelle, pour tout $1 \leq h \leq H$, seul le neurone $\sigma(h)$ change d'état d'activation entre y_{h-1} et y_h, en notant $y_0 \coloneqq -x$ et $y_H \coloneqq x$. L'ensemble $\{y_h : 0 \leq h \leq H\}$ vérifie alors la condition (ii) du Théorème 4.1 (voir Figure 2). Les y_h sont calculés par dichotomie. Ensuite, chaque biais $\hat{b}_{\sigma(h)}$ est déterminé par la formule $\hat{b}_{\sigma(h)}C_{\sigma(h)}(\hat{W}_2) = \eta_h(\partial f(y_h)y_h - \partial f(y_{h-1})y_{h-1} + f(y_{h-1}) - f(y_h))$ où $\eta_h = 1$ si y_{h-1} active $\sigma(h)$, et -1 sinon. Cette formule est une réécriture de l'égalité $f(y_h) + \partial f(y_h)(z - y_h) = f(y_{h-1}) + \partial f(y_{h-1})(z - y_{h-1})$ pour tout z tel que $\langle L_i(W_1), z\rangle + b_i = 0$, en utilisant que $\partial f(y_{h-1}) - \partial f(y_h) = \eta_h C_{\sigma(h)}(\hat{W}_2)L_{\sigma(h)}(\hat{W}_1)^\top$ d'après (6).

5 Réseaux papillon multicouche

Les MLP papillon à L couches (voir (3)) sont un cas particulier de MTP. Leur Jacobienne prend donc la forme suivante.

Lemme 5.1. *Soit f un MTP papillon* ReLU *à L couches. Soit $1 \leq i_0, i_L \leq d$ avec $d \coloneqq 2^L$. Soient $(i_0, i_1, \ldots, i_L)$ les neurones contenus dans l'unique chemin $i_0 \to i_L$ de f reliant i_0 à i_L. Pour tout vecteur d'entrée $x \in \mathbb{R}^d$, on a :*

$$\partial f(x)_{i_L i_0} = \mathbf{1}_{\{x \text{ active } i_0 \to i_L\}} \prod_{\ell=1}^{L} (W_\ell)_{i_\ell i_{\ell-1}}.$$

Une conséquence importante du Lemme 5.1 est que, comme dans le cas à une couche cachée, on peut déterminer à partir de $\partial f(x)$ les chemins activés par x : le chemin $i_0 \to i_L$ est activé par x si et seulement si $\partial f(x)_{i_L i_0} \neq 0$.

Comme dans le cas à une couche cachée, on souhaite caractériser les ensembles d'entrée $\mathcal{X}$ qui permettent la reconstruction d'un réseau ReLU papillon multicouche. On supposera dans la suite que le réseau est sans biais, le cas avec biais présentant des difficultés techniques encore non résolues.

[2] On note $\text{sgn}(a) = 1$ si $a \geq 0$ et $\text{sgn}(a) = -1$ sinon.

Soit P la matrice définie par l'équation (2) pour $(W_1, \dots, W_L)$ des poids papillon. Chacune des $2Ld$ colonnes de P correspond à un élément i, j du support papillon de l'une des matrices $W_1, \dots, W_L$, ou encore à une arête du graphe papillon associé à $(W_1, \dots, W_L)$ (Figure 1). Avec ce point de vue, on associe à chaque ligne de P un chemin dans ce graphe, dont les arêtes sont données par les colonnes contenant un 1 sur cette ligne. Nous identifierons donc dans la suite les lignes de P à des chemins. Nous avons alors le résultat suivant.

Théorème 5.1. *Soit f un MLP papillon* ReLU *sans biais, de paramètres $(W_1, \dots, W_L) \in \mathbb{R}^d$. Soient $\mathcal{X} \subset \mathbb{R}^d$ un ensemble de vecteurs d'entrée, P définie par l'équation 2, et P' la matrice contenant les lignes de P correspondant à des chemins activables*[3]*. L'observation de $(f(x))_{x\in\mathcal{X}}$ et $(\partial f(x))_{x\in\mathcal{X}}$ permet la reconstruction de paramètres papillon $(\hat{W}_1, \dots, \hat{W}_L)$ associés à une réalisation $\hat{f}$ égale à f si, et seulement si, l'ensemble des chemins activés par $\mathcal{X}$ correspond à un sous-ensemble de lignes de P' de rang égal à* $\operatorname{rg} P'$.

La procédure suivante renvoie des paramètres $(\hat{W}_1, \dots, \hat{W}_L)$ induisant une réalisation $\hat{f}$ égale à f, sous réserve d'avoir activé un sous-ensemble de rang plein des lignes de P' au cours de l'Étape 1.

Etape 1 : échantillonnage. Trouver un ensemble $\mathcal{X}$ vérifiant la condition du Théorème 5.1 est un problème plus difficile que dans le cas à une couche cachée. En particulier, les vecteurs x et $-x$ pour x quelconque activent deux ensembles disjoints de chemins, mais pas forcément complémentaires : certains chemins peuvent être désactivés à la fois par x et $-x$. Nous choisissons donc $\mathcal{X}$ comme un ensemble de n vecteurs i.i.d. de loi gaussienne $\mathcal{N}(0, I_d)$, avec n fixé a priori.

Etape 2 : reconstruction partielle du produit $W_L \dots W_1$. Soit $\mathcal{X}$ l'ensemble construit au paragraphe précédent. On estime le produit $X \coloneqq W_L \dots W_1$ de la façon suivante. Soit $1 \le i, j \le d$. S'il existe $x \in \mathcal{X}$ qui active le chemin $i \to j$, on pose $\tilde{X}_{ij} \coloneqq \partial f(x)_{ij}$, cette valeur étant indépendante du choix de x d'après le Lemme 5.1. Sinon, on pose $\tilde{X}_{ij} \coloneqq 0$. Alors $\tilde{X}$ est une version masquée du produit X : il existe une matrice $\tilde{M} \in \{0,1\}^{d\times d}$ telle que $\tilde{X} = X \odot \tilde{M}$. L'algorithme de factorisation papillon [12] ne peut donc pas être appliqué tel quel, il nous faut un algorithme adapté au cas papillon *masqué*.

Etape 3 : factorisation de X via $\tilde{X}$. On voudrait trouver à partir de $\tilde{X}$ des matrices papillon $(\tilde{W}_1, \dots, \tilde{W}_L)$ telles que $\tilde{W}_L \dots \tilde{W}_1$ soit aussi proche que possible de X. Pour ce faire, nous introduisons un algorithme de factorisation MTP adapté au cas où la matrice à factoriser a des valeurs manquantes.

Algorithme 5.1. *Soit $T \coloneqq T_2T_1$ avec (T_1, T_2) un MTP. Soit M une matrice binaire et supposons que l'on observe $\tilde{T} \coloneqq (T_2T_1) \odot M$. On va exploiter successivement toutes les contraintes $C_{ij} : \{T_{ij} = (T_2)_{i,h(i,j)}(T_1)_{h(i,j),j}\}$ pour $(i,j) \in \operatorname{Supp}\tilde{T}$, où $h(i,j)$ est l'unique indice tel que $(i, h(i,j)) \in \operatorname{Supp} T_2$ et $(h(i,j), j) \in \operatorname{Supp} T_1$, bien défini car (T_1, T_2) est un MTP. Tant qu'il existe une contrainte C_{ij} non exploitée, on tire $(\tilde{T}_1)_{h(i,j),j} \sim \mathcal{N}(0,1)$ indépendamment du passé. Puis on instancie récursivement tous les coefficients encore inconnus de $\tilde{T}_1$ et $\tilde{T}_2$ qui sont impliqués par les contraintes au vu de la valeur $(\tilde{T}_1)_{h(i,j),j}$. Pour chaque (i', j') tel que $(\tilde{T}_1)_{h(i',j'),j'}$ et $(\tilde{T}_2)_{i',h(i',j')}$ ont été instanciés, on retire $C_{i'j'}$ de la liste des contraintes non exploitées.*

Cet algorithme termine, car le nombre de contraintes non exploitées décroît strictement à chaque itération. On l'applique ensuite itérativement. Posons $\tilde{T}_1 \coloneqq \tilde{X}$. Pour $\ell = 1, \dots, L-1$, on factorise $\tilde{T}_\ell$ en $\tilde{T}_{\ell+1}\tilde{X}_\ell$, où $\operatorname{Supp}\tilde{T}_{\ell+1} = S_L \dots S_{\ell+1}$ et $\operatorname{Supp}\tilde{X}_\ell = S_\ell$. En posant $\tilde{X}_L \coloneqq \tilde{T}_L$, cela donne des facteurs papillon $(\tilde{X}_1, \dots, \tilde{X}_L)$ tels que $(\tilde{X}_L \dots \tilde{X}_1 - X) \odot \tilde{M} = 0$.

Etape 4 : redressement. Pour ℓ allant de 1 à $L-1$, et pour tout h à ℓ fixé, on change le signe de $L_h(\tilde{W}_\ell)$ et $C_h(\tilde{W}_{\ell+1})$ s'il existe $x \in \mathcal{X}$ vérifiant simultanément : (i) $\langle L_h(\tilde{W}_\ell), \rho(\tilde{W}_{\ell-1}\rho(\dots \tilde{W}_1 x))\rangle \le 0$; (ii) il existe un chemin activé par x contenant le neurone h de la couche cachée ℓ.

Conclusion. Pour des architectures neuronales parcimonieuses structurées, les MTP, nous avons exhibé des algorithmes de reconstruction munis de garanties qui court-circuitent la descente de gradient. Ces algorithmes pourraient servir de base à des techniques économes de distillation pour développer l'utilisation de réseaux parcimonieux. Pour cela, au-delà du passage à l'échelle des algorithmes de reconstruction et de la validation numérique de leur robustesse, un enjeu clé sera d'évaluer le degré d'expressivité de l'architecture MTP considérée ou de ses extensions naturelles à des réseaux modernes organisés selon des graphes acycliques dirigés [5].

[3] Quand $L \ge 3$, et contrairement au cas $L = 2$, certains neurones (et donc certains chemins) peuvent ne pas être activables, par exemple si W_ℓ a une ligne à coefficients tous négatifs pour un certain $2 \le \ell \le L-1$.

Références

[1] Valérie CASTIN et Rémi GRIBONVAL : Code pour une recherche reproductible. `https://hal.science/hal-05136299`, 2025.

[2] James W COOLEY et John W TUKEY : An algorithm for the machine calculation of complex fourier series. *Mathematics of computation*, 19(90):297–301, 1965.

[3] Joana C COSTA, Tiago ROXO, Hugo PROENÇA et Pedro RM INÁCIO : How deep learning sees the world : A survey on adversarial attacks & defenses. *IEEE Access*, 2024.

[4] George W FURNAS et Jeff ZACKS : Multitrees : enriching and reusing hierarchical structure. *In Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 330–336, 1994.

[5] Antoine GONON, Nicolas BRISEBARRE, Elisa RICCIETTI et Rémi GRIBONVAL : A path-norm toolkit for modern networks : consequences, promises and challenges. *In ICLR*, 2024.

[6] Niv HAIM, Gal VARDI, Gilad YEHUDAI, Ohad SHAMIR et Michal IRANI : Reconstructing training data from trained neural networks. *Advances in Neural Information Processing Systems*, 35:22911–22924, 2022.

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

[8] Quoc-Tung LE, Léon ZHENG, Elisa RICCIETTI et Rémi GRIBONVAL : Fast learning of fast transforms, with guarantees. *In ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3348–3352. IEEE, 2022.

[9] Quoc-Tung LE, Léon ZHENG, Elisa RICCIETTI et Rémi GRIBONVAL : Butterfly factorization with error guarantees, novembre 2024. arXiv :2411.04506.

[10] David ROLNICK et Konrad KORDING : Reverse-engineering deep relu networks. *In International conference on machine learning*, pages 8178–8187. PMLR, 2020.

[11] Pierre STOCK et Rémi GRIBONVAL : An embedding of relu networks and an analysis of their identifiability. *Constructive Approximation*, 57(2):853–899, 2023.

[12] Léon ZHENG, Elisa RICCIETTI et Rémi GRIBONVAL : Efficient identification of butterfly sparse matrix factorizations. *SIAM Journal on Mathematics of Data Science*, 5(1):22–49, 2023.

Travail financé par les projets SHARP (ANR-23-PEIA-0008, financé par France 2030), et AllegroAssai (ANR-19-CHIA-0009).