Quantification d'incertitudes au sein des réseaux de neurones : Application à la mesure automatisée de la taille de particules de TiO2

Paul Monchot

INSTITUT POLYTECHNIQUE DE PARIS

NNT : 2023IPPAX163

Thèse de doctorat

ÉCOLE POLYTECHNIQUE

IP PARIS

Quantification d'incertitudes au sein des réseaux de neurones: Application à la mesure automatisée de la taille de particules de TiO_2.

Thèse de doctorat de l'Institut Polytechnique de Paris
préparée à École polytechnique

École doctorale n°574 École doctorale de l'Institut Polytechnique de Paris (EDMH)
Spécialité de doctorat : Mathématiques appliquées

Thèse présentée et soutenue à Palaiseau, le 16 octobre 2023, par

Paul Monchot

Composition du Jury :

Karim Lounici Professeur, École Polytechnique, IPP	Président
Sébastien Da Veiga Professeur associé, CREST, ENSAI	Rapporteur
Pierre-Marc Jodoin Professeur, Université de Sherbrooke	Rapporteur
Agnès Lagnoux Maître de Conférence, Université Toulouse Jean Jaurès	Examinateur
Mathilde Mougeot Professeur, ENSIIE	Examinateur
Erwan Le Pennec Professeur, École Polytechnique, IPP	Directeur de thèse
Nicolas Fischer Responsable du département science des données et incertitudes, LNE	Invité
Loïc Coquelin Ingénieur de recherche, LNE	Invité

À mon père.

Remerciements

Je souhaite exprimer ma sincère gratitude au Laboratoire National de métrologie et D'Essais (LNE) pour m'avoir accueilli pendant cette thèse CIFRE, financée par l'Association nationale de la recherche et de la technologie (ANRT), ainsi qu'au Centre de Mathématique Appliqué (CMAP) de l'école Polytechnique.

J'adresse mes remerciements à toutes les personnes qui ont contribué, échangé et collaboré avec moi au cours de ces trois années de thèse, en particulier mes collègues : (par ordre alphabétique) Dr Séverine Demeyer, Dr Sébastien Marmin, Dr Sébastien Petit et Dr Jabran Zaouali.

Je tiens tout particulièrement à exprimer ma reconnaissance envers mon encadrant au sein du LNE, Dr Loïc Coquelin, avec qui j'ai travaillé quotidiennement et échangé des idées enrichissantes. Je souhaite également remercier mon directeur de thèse, M. Erwan Le Pennec, qui m'a guidé dans mes développements en laissant libre cours à mes idées et à mon cheminement scientifique. Enfin, mes remerciements vont à mon responsable de département, Dr Nicolas Fischer, qui a su créer un environnement propice et serein pour mener à bien mes travaux. Leur soutien constant, leur expertise et leurs conseils ont grandement contribué à la réussite de cette thèse. Je suis reconnaissant d'avoir eu la chance de travailler avec une équipe aussi talentueuse et bienveillante.

Je tiens également à exprimer ma reconnaissance envers tous mes proches, tout particulièrement ma compagne, Mme Joanna Montowska. Son soutien moral et ses encouragements ont été d'une importance capitale.

Mes sincères remerciements vont à tous ceux qui ont contribué de près ou de loin à la réalisation de cette thèse. Votre précieuse aide a été un pilier essentiel de mon parcours doctoral. À ce titre, j'aimerai citer Dr Nicolas Feltin, Dr Alexanra Delvallée, Dr Loïc Crouzier, Mr Antoine Bartoli et Mlle Maureen Bouilloux avec qui j'ai collaboré afin de mener à bien le développement d'ingénierie de notre cas d'application.

Abstract

L'utilisation croissante de solutions technologiques fondées sur des algorithmes d'apprentissage profond a connu une explosion ces dernières années en raison de leurs performances sur des tâches de détection d'objets, de segmentation d'images et de vidéos ou encore de classification, et ce dans de nombreux domaines tels que la médecine, la finance, la conduite autonome ... Dans ce contexte, la recherche en apprentissage profond se concentre de plus en plus sur l'amélioration des performances et une meilleure compréhension des algorithmes utilisés en essayant de quantifier l'incertitude associée à leurs prédictions. Fournir cette incertitude est clé pour une dissémination massive de ces nouveaux outils dans l'industrie et lever les freins actuels pour des systèmes critiques notamment. En effet, fournir l'information de l'incertitude peut revêtir une importance réglementaire dans certains secteurs d'activité.

Ce manuscrit expose nos travaux menés sur la quantification de l'incertitude au sein des réseaux de neurones. Pour commencer, nous proposons un état des lieux approfondi en explicitant les concepts clés impliqués dans un cadre métrologique. Ensuite, nous avons fait le choix de nous concentrer sur la propagation de l'incertitude des entrées à travers un réseau de neurones d'ores-et-déjà entraîné afin de répondre à un besoin industriel pressant. La méthode de propagation de l'incertitude des entrées proposée, nommée WGMprop, modélise les sorties du réseau comme des mixtures de gaussiennes dont la propagation de l'incertitude est assurée par un algorithme Split&Merge muni d'une mesure de divergence choisie comme la distance de Wasserstein. Nous nous sommes ensuite focalisés sur la quantification de l'incertitude inhérente aux paramètres du réseau. Dans ce cadre, une étude comparative des méthodes à l'état de l'art a été réalisée. Cette étude nous a notamment conduit à proposer une méthode de caractérisation locale des ensembles profonds, méthode faisant office de référence à l'heure actuelle. Notre méthodologie, nommée WEUQ, permet une exploration des bassins d'attraction du paysage des paramètres des réseaux de neurones en prenant en compte la diversité des prédicteurs. Enfin, nous présentons notre cas d'application, consistant en la mesure automatisée de la distribution des tailles de nanoparticules de dioxyde de titane (TiO_2) à partir d'images acquises par microscopie électronique à balayage (MEB). Nous décrivons à cette occasion le développement de la brique technologique utilisée ainsi que les choix méthodologiques de quantification d'incertitudes découlant de nos recherches.

Table des matières

Chapitre 1

Introduction

La notion d'incertitude, étroitement liée à la notion de doute, émerge dès lors qu'une quantité n'est pas connue avec certitude. Elle peut, par exemple, être présente au sein d'un processus de mesure, où l'incertitude peut se manifester à travers l'utilisation d'un outil de mesure dont la précision est limitée. On peut imaginer l'exemple d'un entraîneur d'athlétisme chronométrant la course de sportifs à l'aide d'un chronomètre manuel. Il est facile de comprendre que le résultat obtenu par l'entraîneur n'est qu'une *estimation* plus ou moins précise de la *vraie* durée de la course. Dans cet exemple, l'incertitude du résultat final dépend de différents facteurs, que l'on appelle *sources d'incertitude*, tels que le temps de réaction de l'entraîneur lors du départ, et de l'arrivée, ou encore la précision du chronomètre. Maintenant, imaginons que pour se prémunir d'un imprévu, un second entraîneur chronomètre simultanément la même course. Il apparaît peu probable que les durées fournies par les deux entraîneurs soient identiques. Dans ce cas, quel résultat doit-on garder ? Comment comprendre, expliquer, ou encore expliciter cette disparité de résultats ? Nous verrons tout au long de ce rapport que le domaine de la métrologie, des statistiques et des probabilités offre un large éventail de méthodes afin de répondre à ces questions.

Dans ce contexte, il est aisé de sentir la potentielle complexité des notions d'incertitudes et son vaste champ d'application. Ainsi, quantifier l'incertitude permet avant tout de comprendre l'influence des sources d'incertitude sur notre résultat final. Elle traduit directement la **plage des valeurs probables** et donc la **dispersion** du résultat de mesure. Une bonne quantification de l'incertitude peut alors permettre :

— **Accroître la confiance dans les résultats obtenus :** Lorsque l'incertitude est correctement quantifiée, cela renforce la confiance que l'on peut accorder aux résultats. En fournissant une estimation réaliste de la marge d'erreur associée à une mesure ou à un modèle, la quantification de l'incertitude permet de mieux évaluer la fiabilité et la validité des résultats. Cela est particulièrement important dans les domaines dans lesquels les décisions sont prises en se fondant sur ces résultats, tels que la recherche scientifique, l'ingénierie ou la prise de décision.

— **Améliorer la fiabilité et la sécurité des systèmes complexes :** La quantification de l'incertitude est essentielle pour évaluer la fiabilité des systèmes complexes. En identifiant les sources d'incertitude et en évaluant leur impact sur les performances du système, on peut prendre des mesures pour atténuer les risques et améliorer la sécurité. Cela est crucial dans des domaines tels que l'aérospatiale, l'ingénierie des infrastructures critiques, les systèmes de transport ou encore la gestion des réseaux électriques par exemple.
— **Augmenter la qualité des procédés industriels et de production :** La quantification de l'incertitude peut jouer un rôle clé dans l'optimisation des procédés industriels. En comprenant les sources d'incertitude et leur impact sur les performances des procédés, on peut identifier les domaines nécessitant une amélioration et mettre en place des mesures correctives pour réduire l'incertitude. Cela peut alors conduire à une augmentation de la qualité des produits ou services, à une réduction des déchets et à une meilleure maîtrise des coûts de production.
— **Favoriser une meilleure compréhension des systèmes étudiés :** La quantification de l'incertitude permet d'obtenir une compréhension plus approfondie des systèmes étudiés. En analysant les différentes sources d'incertitude et en évaluant leur contribution à la variabilité des résultats, on peut identifier les zones de connaissances limitées et les besoins de recherche supplémentaires. Cela contribue à une meilleure modélisation, une meilleure prédiction et une meilleure compréhension des phénomènes complexes.
— **Optimiser la prise de décision :** La quantification de l'incertitude fournit une base solide pour la prise de décision éclairée. En comprenant les limites et la variabilité associées aux données et aux résultats, on peut évaluer les risques et les avantages potentiels de différentes options. Cela permet de prendre des décisions plus informées et de choisir la meilleure stratégie ou la meilleure solution en fonction des contraintes et des objectifs spécifiques.
— **Renforcer la communication scientifique et la transparence :** La quantification de l'incertitude permet une communication scientifique plus rigoureuse et transparente. En explicitant les sources d'incertitude et en fournissant des estimations précises, on améliore la compréhension et l'interprétation des résultats par les pairs, les décideurs et le grand public. Cela favorise une communication plus claire, honnête et nuancée, en évitant les extrapolations inappropriées ou les conclusions simplistes. En présentant les résultats accompagnés de mesures d'incertitude appropriées, on renforce la confiance dans les travaux scientifiques et on facilite la reproduction des résultats par d'autres chercheurs, contribuant ainsi à l'avancement global des connaissances.

À l'ère du numérique, où les avancées algorithmiques substituent de plus en plus l'intervention humaine, la nécessité de quantifier l'incertitude au sein de ces systèmes a connu une croissance exponentielle ces dernières décennies. Grâce à l'augmentation considérable de la puissance de calcul et des capacités de stockage des dispositifs informatiques, le déploiement massif de solutions basées

sur les réseaux de neurones est désormais réalisable. Ces avancées technologiques trouvent des applications potentielles dans divers domaines à risques tels que la conduite automobile (Muhammad et al., 2020), l'aide au diagnostic médical (Razzak et al., 2018), la finance (Huang et al., 2020), ainsi que le secteur militaire (Das et al., 2018). Dans ce contexte, il devient essentiel de pouvoir quantifier de manière adéquate l'incertitude de ces méthodes afin de garantir un déploiement sûr et maîtrisé.

Dans la continuité de ce chapitre, nous exposons les principes fondamentaux et les méthodes fondatrices de la quantification de l'incertitude, à la fois dans le domaine de l'apprentissage statistique que dans un cadre métrologique. Nous mettons en évidence les avancées significatives réalisées, tout en soulignant les limitations de certaines approches et les lacunes éventuelles présentes dans les connaissances actuellement disponibles dans la littérature scientifique.

1.1 Cadre mathématique général

Avant de présenter l'état de l'art ainsi que nos recherches effectuées, il est nécessaire de fournir un cadre mathématique rigoureux sur lequel nous nous appuierons tout au long de ce rapport, sauf indication contraire explicite.

Nous dénotons par $\mathbf{X} \in \mathbb{R}^d$ les variables aléatoires d'entrée d'un modèle, et par $\mathbf{x}$ une réalisation de cette variable aléatoire. De manière similaire, nous notons par $\mathbf{Y} \in \mathbb{R}^{d'}$ les variables aléatoires de sortie, et par $\mathbf{y}$ une réalisation de cette variable aléatoire. Nous désignons par f le modèle considéré tel que $\mathbf{y} = f(\mathbf{x})$.

La densité de probabilité de la variable aléatoire $\mathbf{X}$ est notée $p_{\mathbf{X}}$ ou $p(\mathbf{X})$, où p est une fonction non négative et intégrable au sens de Lebesgue. De la même manière, nous notons $P_{\mathbf{X}}$ ou $P(\mathbf{X})$ la distribution de la variable aléatoire $\mathbf{X}$. Enfin, on note $F_{\mathbf{X}}$ la fonction de répartition de la variable aléatoire $\mathbf{X}$, définie comme $F_{\mathbf{X}}(x) = \mathbb{P}(\mathbf{X} \leq x)$, où $\mathbb{P}$ est une mesure de probabilité.

L'espérance mathématique de $\mathbf{X}$ est notée $\mu_{\mathbf{X}} = \mathbb{E}[\mathbf{X}]$. Pour spécifier la distribution considérée pour l'espérance, nous pouvons utiliser la notation $\mathbb{E}_{\mathbf{X}\sim p_{\mathbf{X}}}[\cdot]$. De même, nous notons $\Sigma_{\mathbf{X}} = \mathbb{V}[\mathbf{X}]$ la variance de $\mathbf{X}$ (et $\sigma_{\mathbf{X}}$ représente l'écart type de $\mathbf{X}$ dans le cas uni-dimensionnel).

Enfin, pour deux variables aléatoires A et B, nous notons $p(A, B)$ la densité de probabilité jointe de A et B, et nous notons $p(A \mid B)$ la densité de probabilité de A conditionnellement à B.

Notre travail se focalise sur une famille de modèle $f\colon \mathbb{R}^d \to \mathbb{R}^{d'}$, appelée réseaux de neurones, représentant des fonctions paramétriques, dont les paramètres sont estimés à l'aide d'une base de

données d'exemples, appelée base d'entraînement ou base d'apprentissage.

Dans ce contexte, nous introduisons les notations suivantes. Nous définissons $\mathcal{D}$ comme étant la distribution idéale (X_{true}, Y_{true}), où X_{true} est la variable aléatoire des entrées et Y_{true} est la variable aléatoire des sorties. De plus, nous définissons f_{true} comme étant le prédicteur idéal qui associe à chaque valeur $\mathbf{x}_{true}$ de X_{true} la valeur $\mathbf{y}_{true}$ correspondante.

L'ensemble de données d'apprentissage est noté $\mathcal{D}_N = \{\mathbf{x}_i, \mathbf{y}_i\}_{0 \leq i < N}$, où N représente le nombre d'échantillons. Cet ensemble de données est construit de manière à être le plus proche possible de $\mathcal{D}$.

Un algorithme d'apprentissage profond cherche à approximer la distribution conditionnelle $p(\mathbf{y}|\mathbf{x})$ en utilisant une fonction paramétrique $f_{model}(\cdot, \theta) = f_{\mathcal{M}}(\cdot, \theta)$. Les paramètres de cette fonction, notés θ, sont estimés à l'aide d'une fonction de coût $\mathcal{L}$ et d'un algorithme basé sur une descente de gradient stochastique (SGD). Les poids et biais du modèle sont inclus dans le vecteur de paramètres θ. L'apprentissage étant implémenté de façon itérative, on pourra notamment noter θ_t pour rendre compte de la valeur des paramètres du réseau à l'itération t. La fonction de coût permet de quantifier la performance courante du modèle sur la tâche apprise et spécifiée par $\mathcal{D}_N$.

En pratique, le nombre de données dans la base pouvant être très élevée, l'optimisation s'effectue séquentiellement en observant qu'une sous-partie de $\mathcal{D}_N$ à la fois, que l'on appelle *mini-batch*. Les données d'entrée composant ces mini-batchs peuvent également faire l'objet de transformations telles que des rotations, des retournements, des modifications de luminosité et de contraste, permettant d'augmenter virtuellement le nombre de données dans la base d'entraînement et ainsi rendre le réseau final plus robuste (aux *shifts* dans la distribution d'entrée).

Comme tout algorithme d'optimisation reposant sur une base d'exemple, les réseaux de neurones sont prompts au sur-apprentissage. Les notions de sous et sur-apprentissage sont étroitement liées au compromis biais-variance. En général, un réseau de neurones présentant un biais élevé et une variance faible est considéré dans un état de sous-apprentissage. Au contraire, un réseau présentant un faible biais et une variance élevée est considéré dans un état de sur-apprentissage. Afin de juger de l'état du réseau de neurones durant l'entraînement, le jeu de données original $\mathcal{D}_N$ peut être *découpé* en un jeu d'entraînement (sur lequel les paramètres seront optimisés), un jeu de validation (sur lequel on juge du sous/sur-apprentissage du réseau) et un jeu de test permettant de quantifier les performances globales du réseau final.

Remarque 1.

La stochasticité dans l'optimisation provient de l'initialisation aléatoire des paramètres θ du réseau de neurones ainsi que de la nature stochastique des techniques d'augmentation des données et de l'ordre de présentation des lots (mini-batchs) de ces mêmes données.

L'initialisation des paramètres du réseau de neurones joue un rôle crucial dans l'obtention d'une convergence rapide. Plusieurs méthodes ont été proposées dans la littérature, les deux principales étant l'initialisation uniforme de Xavier/Glorot (Glorot and Bengio, 2010) et l'initialisation uniforme de He (He et al., 2015).

Dans les approches de l'état de l'art, les optimiseurs les plus couramment utilisés sont des variantes de l'algorithme de descente stochastique du gradient (SGD) :

$$\theta_{t+1} := \theta_t - \eta \nabla \mathcal{L}(\theta_t) = \theta_t - \frac{\eta}{n} \sum_{i=1}^{n} \nabla \mathcal{L}_i(\theta_t), \tag{1.1}$$

où η représente le taux d'apprentissage, $\mathcal{L}$ représente la fonction de coût, $\mathcal{L}_i$ est la fonction de coût évaluée sur le mini-batch i, et n est le nombre total de mini-batchs.

Parmi les variantes d'optimiseurs les plus répandues, citons RMSProp (Tieleman et al., 2012), AdaGrad (Duchi et al., 2011) et Adam (Kingma and Ba, 2014).

La famille de fonctions paramétriques $f_{\mathcal{M}}$ est déterminée par l'architecture du modèle d'apprentissage profond sélectionné, représentée par un ensemble d'hyperparamètres noté $\mathcal{M}$.

Lors de la phase d'entraînement, à l'aide de l'algorithme de descente de gradient stochastique et des hyperparamètres correspondants notés *opt*, nous obtenons une estimation des paramètres du modèle, notée $\hat{\theta}_{\mathcal{D}_N,opt}$, utilisant l'ensemble de données $\mathcal{D}_N$. Le prédicteur estimé correspondant est noté $\hat{f}_{\mathcal{M}}(\cdot, \hat{\theta}_{\mathcal{D}_N,opt})$.

Lors de l'inférence, pour une nouvelle entrée non observée $\mathbf{x}^*$ (estimation de $\mathbf{x}^*_{true}$), nous effectuons une prédiction $\mathbf{y}^* = \hat{f}_{\mathcal{M}}(\mathbf{x}^*, \hat{\theta}_{\mathcal{D}_m,opt})$, estimation de la valeur $\mathbf{y}^*_{true}$ correspondante. En général, le superscript * désigne un nouvel élément encore jamais observé, alors que le superscript $\hat{\cdot}$ dénote un estimé d'une grandeur d'intérêt.

En accord avec les normes de métrologie, nous utilisons la notation $u(s)$ pour représenter l'incertitude-type de la source s, et $u_s(\mathbf{x})$ pour représenter l'incertitude sur la valeur $\mathbf{x}$ engendrée par l'incertitude sur s.

La figure 1.1 présente fonctionnellement l'ensemble de ces éléments :

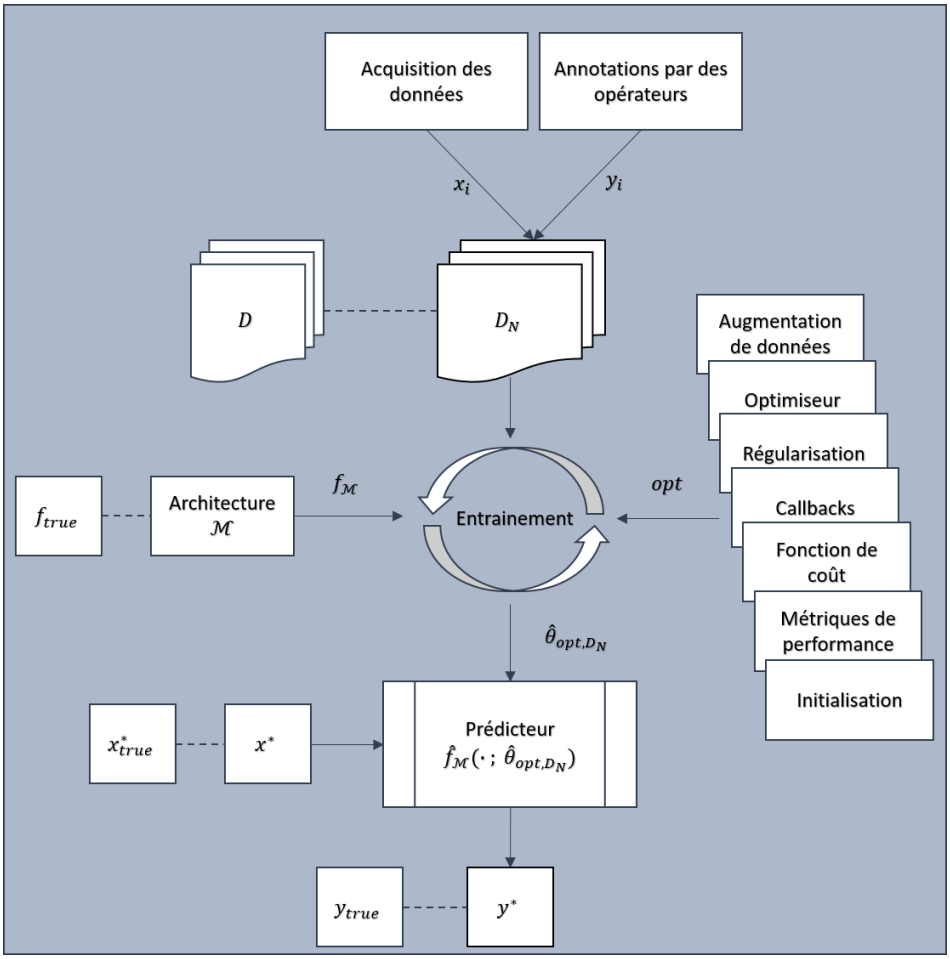


FIGURE 1.1 – Vue d'ensemble des éléments constitutifs d'une prédiction d'un réseau de neurones.

À la vue de cette figure, la question à laquelle nous désirons répondre est la suivante :

Quelle est la ***plage*** *des* ***valeurs probables*** *de* $\mathbf{y}^*$ *?*

La section suivante présente comment la communauté de l'apprentissage statistique essaye de répondre à cette question.

1.2 Incertitude en apprentissage profond : état de l'art et méthodes

La quantification des incertitudes au sein des algorithmes d'apprentissage profond est un domaine largement étudié depuis les premiers travaux fondateurs des années 90, qui ont notamment exploré le concept de réseaux de neurones bayésiens (MacKay, 1992; Neal, 2012). Comme souligné

précédemment, la quantification des incertitudes revêt une importance cruciale pour déployer de manière fiable et contrôlée les algorithmes d'apprentissage profond dans l'industrie et pour des applications grand public. Cette problématique devient d'autant plus pertinente que les réseaux de neurones ont tendance à manifester une confiance excessive dans leurs prédictions (Guo et al., 2017). Un exemple notable en est la difficulté pour ces algorithmes à détecter les échantillons situés en dehors de leur plage de fonctionnement, communément appelés échantillons "out-of-distribution" (OOD). De plus, avec l'évolution des algorithmes et de la puissance de calcul disponible (la mémoire matérielle, les capacités des GPU, ...), les réseaux de neurones sont de plus en plus utilisés dans des secteurs où les prises de décision revêtent un risque (humain, matériel, ...), telles que l'analyse de données médicales (Razzak et al., 2018; Shen et al., 2017; Chan et al., 2020), la conduite autonome (Muhammad et al., 2020; Grigorescu et al., 2020), ainsi que le secteur bancaire (Huang et al., 2020; Heaton et al., 2016). Dans ce contexte, il est essentiel de pouvoir fournir une estimation de l'incertitude prédictive afin d'améliorer la sécurité et d'établir un système de secours fiable.

Der Kiureghian and Ditlevsen (2009) fait office de référence et propose de répertorier les sources d'incertitude au sein de modèles statistiques. Dans cet article, les auteurs dressent une liste exhaustive des différentes sources d'incertitude à considérer lors de la modélisation statistique, tout en argumentant la possibilité de les classer a posteriori selon une catégorisation binaire entre les incertitudes "épistémiques" et "aléatoires". Cette classification permet en outre d'optimiser l'allocation des ressources en vue de réduire l'incertitude prédictive.

En effet, dans la littérature, le terme "épistémique", dérivé du grec "episteme" signifiant connaissance, est utilisé pour décrire les sources d'incertitude résultant d'un manque de connaissance du problème, pouvant se traduire par un déficit de données dans notre ensemble d'apprentissage. Par conséquent, l'augmentation du nombre de données dans cet ensemble permettrait de réduire l'incertitude prédictive liée à l'ensemble de ces sources.

D'autre part, le terme "aléatoire", également d'origine grecque ("alea"), désigne les sources d'incertitude correspondant à un phénomène intrinsèquement aléatoire du problème. Ces sources d'incertitude ne peuvent donc pas être atténuées par l'augmentation du nombre de données dans notre base de données.

La décomposition de l'incertitude en ces deux catégories est traduite mathématiquement par la relation suivante :

$$p\left(\mathbf{y}^* \mid \mathbf{x}^*, \mathcal{D}_N\right) = \int \underbrace{p(\mathbf{y}^* \mid \theta, \mathbf{x}^*)}_{\text{Aléatoire}} \underbrace{p(\theta \mid \mathcal{D}_N)}_{\text{Epistémique}} d\theta. \tag{1.2}$$

Remarque 2.

Pour une tâche de classification, où la couche de sortie est munie d'une fonction softmax, on peut mesurer l'incertitude comme l'entropie $\mathcal{H}[\mathbf{y}^* \mid \mathbf{x}^*, \mathcal{D}_N]$, décomposée de la façon suivante (Depeweg et al., 2017) :

$$\mathcal{H}[\mathbf{y}^* \mid \mathbf{x}^*, \mathcal{D}_N] = \mathcal{I}[\mathbf{y}^*, \theta \mid \mathbf{x}^*, \mathcal{D}_N] + E_{\theta \sim p(\theta \mid \mathcal{D}_N)}[\mathcal{H}[\mathbf{y}^* \mid \mathbf{x}^*, \theta]] \tag{1.3}$$

où $\mathcal{H}$ est l'entropie, et $\mathcal{I}$ l'information mutuelle.

Cette dernière équation peut se traduire comme la décomposition de l'incertitude prédictive en fonction de l'incertitude aléatoire, par le biais du terme $E_{\theta \sim p(\theta \mid \mathcal{D}_N)}[\mathcal{H}[\mathbf{y}^* \mid \mathbf{x}^*, \theta]]$ qui prend l'espérance sur le postérieur des paramètres θ conditionnellement à la base de données $\mathcal{D}_N$; et de l'incertitude épistémique, par le biais du terme $\mathcal{I}[\mathbf{y}^*, \theta \mid \mathbf{x}^*, \mathcal{D}_N] = \mathcal{H}[\mathbf{y}^* \mid \mathbf{x}^*, \mathcal{D}_N] - E_{\theta \sim p(\theta \mid \mathcal{D}_N)}[\mathcal{H}[\mathbf{y}^* \mid \mathbf{x}^*, \theta]]$ qui est la composante de l'incertitude restante et capture l'information mutuelle entre $\mathbf{y}^*$ et θ.

Cette classification de l'incertitude a été largement adoptée au sein de la communauté de l'apprentissage statistique. Dans les sections suivantes, nous exposons les méthodes permettant de quantifier ces deux catégories d'incertitude, ainsi que les méthodes permettant de répondre à certaines sources d'incertitude définies lors de notre approche métrologique de l'établissement de la liste des ces sources (cf. section 1.4).

Les sections 1.2.1–1.2.6 dressent alors un état des lieux des notions d'incertitude telles qu'elles sont appréhendées par la communauté de l'apprentissage profond, en mettant l'accent sur les algorithmes d'apprentissage supervisé. Notre objectif, ici, n'est pas d'être exhaustif en termes de techniques de quantification d'incertitudes, mais plutôt de donner un aperçu de la façon dont cette communauté aborde ces questions. En effet, de nombreux articles de revue sont disponibles (Gawlikowski et al., 2021; Jospin et al., 2022; Abdar et al., 2021) et offrent une vue d'ensemble approfondie de cette problématique.

1.2.1 Incertitude aléatoire

L'incertitude aléatoire (également rencontrée sous la dénomination d'incertitude des données, des entrées ou des observations), est une catégorie de sources d'incertitude qui ne peuvent pas être réduites en augmentant le nombre de données. Ce type d'incertitudes peut présenter une composante systématique (appelée homoscédastique et dont la variance ne dépend pas de l'échantillon) et une composante variable (appelée hétéroscédastique et dont la variance dépend de l'échantillon considéré). Cette catégorie est peu fournie au sein de la littérature. Nous présentons néanmoins deux approches.

Estimation par le réseau

Kendall and Gal (2017) proposent de quantifier l'incertitude aléatoire hétéroscédastique en supposant la fonction de vraisemblance $p\left(\mathcal{D}_N \mid \theta\right)$ comme gaussienne :

$$-\log p\left(\mathbf{y}_i \mid f_{\mathcal{M}}\left(\mathbf{x}_i; \hat{\theta}_{\mathcal{D}_N,opt}\right)\right) \propto \frac{1}{2\sigma^2}\left\|\mathbf{y}_i - f_{\mathcal{M}}\left(\mathbf{x}_i; \hat{\theta}_{\mathcal{D}_N,opt}\right)\right\|^2 + \frac{1}{2}\log\sigma^2 \tag{1.4}$$

où σ représente l'écart type de la distribution de sortie.

Les auteurs proposent alors de faire apprendre, directement par le réseau, le paramètre du bruit d'observation en le rendant dépendant aux données. La fonction de coût devient alors :

$$\mathcal{L} = \frac{1}{N}\sum_{i=1}^{N} \frac{1}{2\sigma\left(\mathbf{x}_i\right)^2}\left\|\mathbf{y}_i - f_{\mathcal{M}}\left(\mathbf{x}_i; \hat{\theta}_{\mathcal{D}_N,opt}\right)\right\|^2 + \frac{1}{2}\log\sigma\left(\mathbf{x}_i\right)^2 \tag{1.5}$$

où N est le nombre de données dans la base d'apprentissage.

À l'inférence, pour une entrée courante i, le réseau fournit une prédiction moyenne $\mu_i = f\left(\mathbf{x}_i; \hat{\theta}_{\mathcal{D}_N,opt}\right)$ ainsi que l'estimation de l'incertitude aléatoire hétéroscédastique associée σ_i.

À retenir

Cette méthode facile d'implémentation a pour principal défaut de reposer sur une forte hypothèse de forme de la fonction de vraisemblance. De plus, cette méthode confit la tâche de quantification de l'incertitude directement au réseau de neurones. Ce type de méthodes est complexe à interpréter et semble difficile d'utilisation dans un contexte métrologique. La méthode est cependant intuitivement intéressante, dans le sens où l'incertitude aléatoire doit se traduire au plus bas niveau sur les cartes caractéristiques "générées" par le réseau.

Modèle d'erreur dans les variables

Martin and Elster (2022) abordent la même problématique en utilisant le paradigme du modèle d'erreur dans les variables (EiV)(Fuller, 2009). EiV propose la modélisation suivante :

$$\mathbf{x}_i = \zeta_i + \varepsilon_{\mathbf{x}_i}, \quad \mathbf{y} = f_{\mathcal{M}}(\zeta_i; \theta) + \varepsilon_{\mathbf{y}_i} \tag{1.6}$$

où ζ_i est la "vraie" entrée mais inconnue car soumise à un bruit $\varepsilon_{\mathbf{x}_i}$, et $\varepsilon_{\mathbf{x}_i} \sim \mathcal{N}\left(0, \sigma_{\mathbf{x}}^2 I_{n_{\mathbf{x}} \times n_{\mathbf{x}}}\right)$ avec $\sigma_{\mathbf{x}}$ fixé, et $\varepsilon_{\mathbf{y}_i} \sim \mathcal{N}\left(0, \sigma_{\mathbf{y}}^2 I_{n_{\mathbf{y}} \times n_{\mathbf{y}}}\right)$ avec $\sigma_{\mathbf{y}}$ mise à jour pendant l'entraînement.

À retenir

Ce modèle propose bien de modéliser l'incertitude aléatoire en explicitant la possibilité d'une incertitude résiduelle $\varepsilon_{\mathbf{y}_i}$ lorsque l'incertitude épistémique est nulle (i.e. quand $\mathbb{V}_\theta\left[f(\zeta_i;\theta)\right]$ est nulle). Elle repose néanmoins sur de fortes hypothèses concernant la forme des bruits d'entrée/sortie considérés. Cette méthode a cependant l'avantage d'être inscrite dans un cadre théorique plus "contrôlé" que la méthode précédente.

1.2.2 Incertitude épistémique

Dans la littérature, l'incertitude épistémique consiste à estimer la contribution de l'incertitude prédictive qui peut être réduite en augmentant le nombre d'exemples d'apprentissage fournit au réseau durant la phase d'entraînement. Cette notion d'incertitude semble, par définition, liée aux approches bayésiennes.

Attention, la quantification de l'incertitude épistémique répond avant tout à la question suivante : *Quelle est la part de l'incertitude pouvant être réduite en augmentant le nombre de données dans la base d'apprentissage ?*, alors que les méthodes bayésiennes permettent d'établir un cadre solide répondant à la question : *Quelle est la probabilité de l'hypothèse compte tenu des données observées ?*. Dans le cadre des réseaux de neurones, répondre à la seconde question à l'aide d'approches bayésiennes ne garantie pas une réponse adéquate à la première.

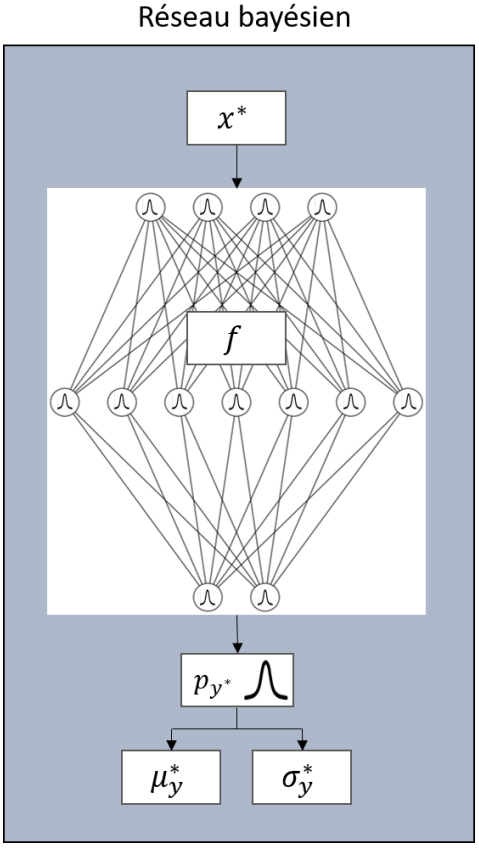


FIGURE 1.2 – Schéma d'un réseau bayésien.

La figure 1.2 présente une représentation d'un réseau de neurones bayésien, où les paramètres ne sont plus déterministes mais suivent une distribution donnée.

Nous rappelons que pour une base d'entraînement $\mathcal{D}_N$, la modélisation bayésienne tend à estimer le postérieur des paramètres du réseau conditionnellement à $\mathcal{D}_N$, par la règle de Bayes (Joyce, 2003) :

$$p(\boldsymbol{\theta} \mid \mathcal{D}_N) = \frac{p(\mathcal{D}_N \mid \boldsymbol{\theta}) p(\boldsymbol{\theta})}{p(\mathcal{D}_N)} \tag{1.7}$$

L'estimation de ce postérieur permet alors pour une nouvelle entrée $\mathbf{x}^*$ d'inférer en intégrant le postérieur des paramètres θ conditionnellement à $\mathcal{D}_N$:

$$p\left(\mathbf{y}^* \mid \mathbf{x}^*, \mathcal{D}_N\right) = \int p\left(\mathbf{y}^* \mid \mathbf{x}^*, \boldsymbol{\theta}\right) p(\boldsymbol{\theta} \mid \mathcal{D}_N) \mathrm{d}\boldsymbol{\theta} \tag{1.8}$$

Ce procédé est appelé Bayesian Model Averaging (BMA), où moyennage de modèles bayésiens en français. La grande difficulté de cette approche est l'estimation du postérieur $p(\boldsymbol{\theta} \mid \mathcal{D}_N)$ qui est intractable dans le cas des réseaux de neurones.

Dans la suite de cette partie, nous présentons les méthodes principales de l'état de l'art suivant cette méthodologie, que l'on peut regrouper en trois catégories distinctes :

- les méthodes d'**inférence variationelle**, consistant à approcher le postérieur intractable $p(\boldsymbol{\theta} \mid \mathcal{D}_N)$ par une distribution plus simple, souvent dénotée $q(\cdot \mid \mathcal{D}_N)$,
- les méthodes basées sur les méthodes de **Monte-Carlo par chaînes de Markov** (MCMC) permettant d'obtenir directement des échantillons de $p(\boldsymbol{\theta} \mid \mathcal{D}_N)$,
- les méthodes d'**ensembles**, consistant à obtenir plusieurs réseaux entraînés sur la même tâche et constituant ainsi un ensemble de prédicteurs indépendants.

Méthodes d'inférence variationelle

Monte-Carlo Dropout (MCD) Le Monte-Carlo Dropout, introduit par Gal and Ghahramani (2016), est largement reconnu et fréquemment utilisé comme méthode de quantification de l'incertitude épistémique (Milanés-Hermosilla et al., 2021; Padarian et al., 2022; Wei et al., 2021). Cette approche se distingue par sa simplicité et son efficacité, ne nécessitant ni augmentation de la mémoire requise ni temps d'entraînement supplémentaire.

La méthode consiste à entraîner un réseau de neurones en incorporant des couches de Dropout (Srivastava et al., 2014), puis à maintenir ces couches actives lors de l'inférence. Il convient de rappeler qu'une couche de Dropout fonctionne comme un interrupteur, où chaque neurone du réseau possède une probabilité p d'être désactivé. Par conséquent, chaque paramètre du réseau est associé à une variable aléatoire suivant une distribution binomiale, prenant la valeur 1 avec une probabilité de $(1-p)$ et 0 avec une probabilité p.

Mathématiquement, cette méthode propose d'approximer le postérieur par $q(\theta \mid \mathcal{D}_N)$ défini par :

$$\theta_i = \mathbf{M}_i \cdot \text{diag}\left([\mathbf{z}_{i,j}]_{j=1}^{K_i}\right) \tag{1.9}$$

$$\mathbf{z}_{i,j} \sim \text{Bernoulli}\,(p_i) \text{ for } i = 1, \ldots, L, j = 1, \ldots, K_{i-1} \tag{1.10}$$

où L est le nombre de couches dans le réseau et K_i le nombre de neurones de la couche i, et $\mathbf{M}_i$, les matrices de paramètres variationnels.

Effectuer N prédictions en tirant N fois dans chacune des distributions binomiales fournit N prédictions, desquelles on extrait une prédiction moyenne ainsi qu'un écart type associé :

$$\bar{\mathbf{y}}^* = \frac{1}{N}\sum_{i=1}^{N}\mathbf{y}_i^* \tag{1.11}$$

$$\sigma_{\mathbf{y}} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(\mathbf{y}_i^* - \bar{\mathbf{y}}^*)^2} \tag{1.12}$$

Cet écart-type est alors l'incertitude épistémique estimée, associée à l'échantillon courant.

Remarque 3.

Le Monte-Carlo Dropout fut originellement présenté comme modélisant l'incertitude de "modèle", puis classifié comme épistémique dans leur second article (Kendall and Gal, 2017).

À retenir

Un unique entraînement suffit afin de mettre en place cette méthode, et aucun coup mémoire n'est ajouté. Cependant, le fondement théorique du Monte-Carlo Dropout reste moins solide que d'autres méthodes bayésiennes, présentés ci-après. Le Monte-Carlo Dropout "injecte" par les couches du Dropout de la variabilité dans le prédicteur, difficilement interprétable comme une incertitude. Cette méthode pourrait également être interprétée comme quantifiant une part de l'incertitude de modèle (cf. section 1.5.3) : l'extinction de certains neurones engendre indirectement une nouvelle architecture.

Bayes-by-Backprop Blundell et al. (2015) proposent une méthodologie bayésienne d'optimisation par rétro-propagation en plaçant un prior gaussien sur les paramètres du réseau et en minimisant la Borne inférieure variationnelle (ELBO) entre le postérieur $p\,(\theta \mid \mathcal{D}_N)$ et la distribution variationnelle $q\,(\theta \mid \mathcal{D}_N)$.

Dans leur article, les auteurs choisissent $q\,(\theta \mid \mathcal{D}_N)$ comme une distribution gaussienne multivariée

munie d'une matrice de variance/covariance diagonale.

Remarque 4.

Un prior gaussien (diagonal, de moyenne $\mu = 0.0$, et d'écart-type $\sigma = 1.0$) sur les paramètres du réseau se traduit par une régularisation L_2 dans la fonction de coût. En effet, pour une optimisation du maximum a posteriori des paramètres du réseau, on a $p(\theta) = \prod_i \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}\theta_i^2} \approx \prod_i e^{-\frac{1}{2}\theta^2}$, donc :

$$\begin{aligned} \theta_{MAP} &= -\underset{\theta}{\operatorname{argmin}} \log\left(p\left(\theta \mid \mathcal{D}_N\right)\right) \\ &\propto -\underset{\theta}{\operatorname{argmin}} \log\left(p\left(\mathcal{D}_N \mid \theta\right)\right) + \log\left(p\left(\theta\right)\right) \\ &\propto -\underset{\theta}{\operatorname{argmin}} \log\left(p\left(\mathcal{D}_N \mid \theta\right)\right) - \frac{1}{2}\sum_i \theta_i^2 \end{aligned} \tag{1.13}$$

À retenir

Cette méthode "fortement" bayésienne a pour principal défaut d'augmenter le nombre de paramètres d'un facteur deux et de pouvoir sous- estimer l'incertitude épistémique (Blei et al., 2017). Elle repose également sur une forte hypothèse de forme du postérieur approché. On note notamment que le choix d'une matrice de variance/covariance diagonale revient à négliger l'intégralité des covariances entre les paramètres du modèle.

Il convient également de mentionner un autre paradigme d'apprentissage probabiliste, tel que présenté dans l'article de Hernández-Lobato and Adams (2015), qui repose sur l'établissement d'un modèle probabiliste du postérieur variationnel $q(\theta \mid \mathcal{D}_N)$ et utilise la divergence de Kullback-Leibler entre le vrai postérieur $p(\theta|\mathcal{D}_N)$ et $q(\theta \mid \mathcal{D}_N)$ comme fonction de coût.

Approximation de Laplace Une autre grande catégorie de méthode repose sur une approximation de Laplace. Ainsi, Ritter et al. (2018) proposent d'effectuer cette approximation autour du mode $\hat{\theta}_{\mathcal{D}_N,opt}$ estimé en fin d'apprentissage :

$$\log p(\theta \mid \mathcal{D}_N) \approx \log p\left(\hat{\theta}_{\mathcal{D}_N,opt} \mid \mathcal{D}_N\right) - \frac{1}{2}\left(\theta - \hat{\theta}_{\mathcal{D}_N,opt}\right)^\top \bar{H} \left(\theta - \hat{\theta}_{\mathcal{D}_N,opt}\right) \tag{1.14}$$

où $\bar{H}$ est l'espérance de la matrice Hessienne du négatif du logarithme du postérieur.

En appliquant une exponentielle, le postérieur des paramètres peut une nouvelle fois être approché par une distribution gaussienne multivariée $\theta \mid \mathcal{D}_N \sim \mathcal{N}\left(\hat{\theta}_{\mathcal{D}_N,opt}, \bar{H}^{-1}\right)$. Contrairement à la méthode "Bayes-by-Backprop" où les variances étaient "apprises" par le réseau à travers la fonction de coût,

ici le moment d'ordre 2 de $q(\theta|\mathcal{D}_N)$ est estimée de façon à incorporer la courbure locale du postérieur autour du mode $\hat{\theta}_{\mathcal{D}_N,opt}$.

À retenir

Cette méthodologie, facile d'implémentation, a pour principale frein la dimension de H, engendrant des difficultés pour estimer convenablement $\bar{H}^{-1}$. Enfin, cette méthode permet uniquement une représentation unimodale du postérieur approché. Nous pourrons voir notamment par la suite que cette hypothèse n'est en pratique pas vérifiée. En revanche, cette méthodologie présente l'avantage de pouvoir être implémentée sans modification de l'architecture ou de la boucle d'entraînement.

Méthodes de Monte-Carlo par chaînes de Markov (MCMC)

Cette catégorie de méthodes repose sur les méthodes de Monte-Carlo par chaînes de Markov, et permettent d'échantillonner directement dans le postérieur des paramètres conditionnellement à la base de données d'apprentissage. De nombreux travaux sont présents dans la littérature, nous présentons ici les deux méthodes principales.

SGLD La méthode SGLD (pour Stochastic Gradient Langevin Dynamic) (Welling and Teh, 2011) est une méthode de MCMC et propose de combiner la descente de gradient stochastique classique avec la dynamique de Langevin. Cela se traduit, en pratique, par l'introduction d'un terme de bruit η_t dans la mise à jour des paramètres du réseau :

$$\Delta\theta_t = \frac{\epsilon_t}{2}\left(\nabla \log p\left(\theta_t\right) + \frac{N}{n}\sum_{i=1}^{n} \nabla \log p\left(\mathbf{x}_{ti} \mid \theta_t\right)\right) + \eta_t \tag{1.15}$$

$$\eta_t \sim N\left(0, \epsilon_t\right) \tag{1.16}$$

où ϵ_t est la taille de pas à l'itération t, N le nombre de données dans la base d'entraînement, et n la taille d'un mini batch.

L'entraînement SGLD consiste alors en deux phases distinctes. Une première phase exploratoire où la dynamique est dominée par les fortes valeurs de gradients, puis la seconde phase, appelée phase de Langevin, commence lorsque le bruit injecté domine dans toutes les directions. Pendant cette phase, l'ensemble des paramètres "rencontrés" sont des échantillons de $p\left(\theta \mid \mathcal{D}_N\right)$.

À l'instar des approximations de Laplace, la méthode SGLD est unimodale, ce qui signifie qu'elle ne permet d'échantillonner le posterieur des paramètres qu'autour de la solution locale trouvée. Afin

de remédier à cela, Loshchilov and Hutter (2016) ont proposé une technique simple de réinitialisation de la valeur du pas, appelée "warm restart", qui suit une fonction sinus permettant d'explorer les différents modes du postérieur à échantillonner. Coupler la méthode SGLD avec le warm restart conduit à la méthode RECAST étudiée par Seedat and Kanan (2019).

En effet, à la fin de l'apprentissage, lorsque la valeur du pas devient faible, les valeurs successives des paramètres dans la trajectoire de la descente de gradient stochastique deviennent proches les unes des autres (du fait des faibles valeurs de pas et de gradient). On peut les considérer comme étant dans un même "bassin d'attraction". Ainsi, en augmentant brusquement la valeur du pas, il est possible de "s'échapper" de ce bassin, et en la diminuant progressivement, on permet une nouvelle convergence vers un autre bassin.

À retenir

Cette méthode est considérée comme fortement bayésienne du fait de son ancrage théorique. Cependant, la méthode est sensible au choix de ses hyperparamètres, notamment ceux du prior $p(\theta)$ choisi et est en pratique difficile d'implémentation afin d'obtenir des performances satisfaisantes. Elle doit également être couplée à des méthodes de "restart" afin de pouvoir capturer différents modes du postérieur.

SGHMC La méthode SGHMC (Chen et al., 2014), pour Stochastic Gradient Hamiltonian Monte Carlo, est l'adaptation pour l'apprentissage des réseaux de neurones de la méthode HMC (Hamiltonian Monte Carlo)(Neal et al., 2011). Cette dernière est également une méthode de MCMC et utilise la dynamique Hamiltonienne afin d'échantillonner une nouvelle fois dans une loi inconnue. Elle est considérée comme la référence des méthodes d'échantillonnage, dans un cadre plus standard.

La méthode HMC propose ainsi d'utiliser les équations de dynamique Hamiltonienne afin de proposer des échantillons (non-corrélés) de la distribution cible présentant une probabilité élevée d'acceptation. Dans ce paradigme, les paramètres θ représentent un vecteur de position (au sens physique), auquel on ajoute un vecteur de moment r (supposé indépendant de θ).

$$
\begin{aligned}
H(\mathbf{r}, \boldsymbol{\theta}) &= -\log p(\mathbf{r}, \boldsymbol{\theta} \mid \mathcal{D}_N)) && (1.17)\\
&= -\log p(\mathbf{r}) - \log p(\boldsymbol{\theta} \mid \mathcal{D}_N)) && (1.18)\\
&= T(\mathbf{r}) + U(\boldsymbol{\theta}) && (1.19)
\end{aligned}
$$

où U est assimilée à une énergie potentielle et T une énergie cinétique.

Pour chaque étape d'échantillonnage, on résout les équations de dynamiques suivantes :

$$\begin{cases} d\theta = M^{-1} r dt \\ dr = -\nabla U(\theta) dt. \end{cases} \tag{1.20}$$

offrant alors un nouvel échantillon (θ, r) de la distribution jointe recherchée. Ignorer les échantillons r conduit à échantillonner dans la loi marginale en θ de $p(\mathbf{r}, \boldsymbol{\theta} \mid \mathcal{D}_N)$ qui s'avère être le postérieur recherché $p(\theta|\mathcal{D}_N)$. D'autres subtilités existent au sein de cette méthode et ne sont pas couvertes ici.

La méthode SGHMC consiste alors en l'adaptation de ces équations due au bruit introduit dans la dynamique lors du calcul du gradient de U sur des mini-batchs au lieu de l'intégralité du jeu de données. Les auteurs introduisent alors un terme de friction dans la dynamique précédente.

À retenir

SGHMC est également une méthode fortement bayésienne et solide théoriquement. La méthode SGHMC est cependant coûteuse en temps, due notamment à la résolution des équations de dynamique, ainsi qu'en mémoire nécessaire. Il peut également s'avérer difficile d'obtenir des performances satisfaisantes en pratique.

Méthodes d'ensemble

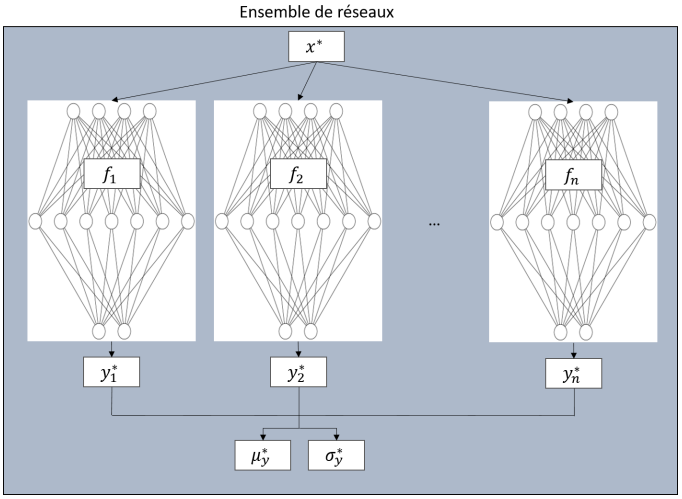


FIGURE 1.3 – Schéma d'un ensemble de réseaux de neurones.

La dernière catégorie d'approche repose sur les ensembles de prédicteurs, et constitue un paradigme largement utilisé en pratique en raison de sa simplicité d'implémentation. Ce groupe d'approches permet d'explorer facilement les différents modes du postérieur des paramètres, ce qui conduit à l'obtention de prédicteurs ayant des représentations différentes de la base de données d'entraînement.

Il convient de souligner que les méthodes d'ensemble ne sont pas initialement classifiées comme méthodes bayésiennes (Lakshminarayanan et al., 2017; Ovadia et al., 2019; Seedat and Kanan, 2019). Cependant, certains travaux discutent du caractère bayésien des méthodes d'ensemble, en justifiant leur validité pour effectuer du BMA (Bayesian Model Averaging) (Wilson and Izmailov,

2020; Hoeting et al., 1999) (l'annexe A présente une réflexion sur ces notions).

De nombreuses comparaisons disponibles dans la littérature mettent en évidence le fait que les méthodes d'ensemble surpassent les méthodes purement bayésiennes en termes de performances (Ovadia et al., 2019; Gustafsson et al., 2020). Par conséquent, ces méthodes peuvent être considérées comme étant à l'état de l'art pour la quantification de l'incertitude des paramètres d'un réseau de neurones.

Ensemble profond Avec le Monte-Carlo Dropout, la méthode d'ensemble profond (Lakshminarayanan et al., 2017) est sans doute la méthode la plus connue et la plus populaire des méthodes de quantification d'incertitude des paramètres d'un réseau de neurones.

Cette méthode consiste à effectuer M entraînements indépendants d'un même réseau de neurones afin d'obtenir M prédicteurs distincts, ayant convergé vers différents modes du "paysage" des paramètres. Cette exploration multimodale permet l'obtention d'une diversité de représentation de la tâche apprise à travers la base de données. Cette diversité de prédicteurs est facilement obtenable grâce à l'initialisation aléatoire des paramètres, ainsi qu'au caractère stochastique de la dynamique de descente de gradient. La force des ensembles profonds peut tenir de sa capacité de réduction de variance, de sa simplicité d'implémentation, mais également par la capacité d'avoir naturellement une diversité élevée. La diversité d'un ensemble est estimée par une métrique tentant d'évaluer la diversité de représentation de la tâche apprise. Ces notions seront discutées plus en détail dans la suite de ce rapport notamment au sein du chapitre 3 et de l'annexe I.

Plus pratiquement, pour un ensemble de prédicteurs ayant la même performance globale, l'erreur moyenne de l'ensemble diminue lorsque ses éléments constitutifs ne commettent pas les mêmes erreurs pour les mêmes données (dans le cas contraire, la variance de l'erreur serait réduite mais le biais resterait élevé). Augmenter la diversité d'un ensemble permet alors de réduire le biais d'erreur de cet ensemble.

À retenir

Les méthodes d'ensemble profond sont extrêmement simples d'utilisation car elles nécessitent uniquement le ré-entraînement du réseau. Cette technique nécessite donc un temps d'entraînement plus élevé et un stockage mémoire plus important. Cette méthode permet, par définition, de capter la part de l'incertitude issue de la stochasticité de la procédure d'optimisation, appelée incertitude de *répétabilité* au sein de la section 1.5.2.

Bagging Le Bagging (Breiman, 1996) ou Bootstrap (Friedman, 2001) consiste à effectuer plusieurs entraînements du même réseau en ne gardant, à chaque répétition, qu'une fraction du jeu de donnée total, tout en gardant un cardinal constant. Le bagging peut être considéré comme une variante des ensembles profonds.

Multi-SWAG Le Multi-SWAG (Wilson and Izmailov, 2020) (pour Stochastic Weight Averaging Gaussian) est une technique présentée comme non bayésienne mais permettant d'effectuer du BMA, et consiste à approcher le bassin d'attraction local par une distribution gaussienne dont les moments sont estimés à l'aide des différents jeux de paramètres rencontrés en fin d'entraînement.

$$\begin{aligned} \theta \mid \mathcal{D}_N &\sim \mathcal{N}\left(\mu_\theta, \Sigma_\theta\right) \\ \mu_\theta &= \frac{1}{T}\sum_{i=1}^{T}\theta_i \\ \Sigma_\theta &= \frac{1}{T-1}\sum_{i=1}^{T}\left(\theta_i - \theta_{\text{SWA}}\right)\left(\theta_i - \theta_{\text{SWA}}\right)^\top \end{aligned} \tag{1.21}$$

La dimension du problème étant très élevée, il est en pratique impossible d'estimer Σ_θ. Les auteurs proposent donc des approximations diagonales et de faible rang de Σ_θ.

À retenir

La méthode "SWAG" originale permet d'approcher le postérieur de façon unimodale. Le "Multi" invite donc à effectuer plusieurs entraînements SWAG à la façon d'un ensemble profond afin de pouvoir capter le caractère multimodale du paysage des paramètres. Cette approche propose alors une exploration locale par l'intermédiaire des distributions gaussiennes et globale en répétant cette méthodologie avec plusieurs initialisations. Elle permet également de réduire la mémoire nécessaire pour stocker les nombreuses copies du même réseau dans le cadre d'un ensemble profond par exemple.

Nous finissons cette présentation en fournissant un tableau de caractéristiques 1.1 des méthodes précédemment mises en avant.

TABLE 1.1 – Caractéristiques des méthodes de quantification de l'incertitude épistémique.

Méthodes	Caractère statistique	Représentation gaussienne	Multimodale	Temps d'apprentissage	Temps d'inférence	Mémoire d'apprentissage	Mémoire d'inférence	Implémentation	Modification d'architecture	Convergence	Sensible aux hyper-paramètres
DE	✗	✗	✓	Élevé	Élevé	Normale	Normale	Facile	✗	Facile	✗
BOOTSTRAP	✗	✗	✓	Élevé	Élevé	Normale	Normale	Facile	✗	Facile	✗
SGLD	✓	✗	✗	Moyen	Élevé	Élevé	Normale	Moyenne	✗	Difficile	✓
SGHMC	✓	✗	✓	Moyen	Élevé	Élevé	Normale	Moyenne	✗	Difficile	✓
MCD	✓	✗	✓	Moyen	Élevé	Normale	Normale	Facile	✓	Moyenne	✓
SWAG	✗	✓	✗	Moyen	Élevé	Normale	Normale	Facile	✗	Facile	✗
Laplace	✗	✓	✗	Faible	Élevé	Élevé	Normale	Facile	✗	Facile	✗
BBB	✓	✓	✓	Élevé	Élevé	Élevé	Élevé	Difficile	✓	Difficile	✓
PBP	✓	✓	✓	Élevé	Élevé	Élevé	Élevé	Difficile	✓	Difficile	✓

Ci-après, nous présentons deux catégories de méthodes permettant de quantifier deux autres types d'incertitude :

— L'incertitude des entrées (cf. section 1.5.1) du réseau de neurones au temps d'inférence au sein de la section 1.2.3,

— L'incertitude de distribution (cf. section 1.5.5) essayant de quantifier l'incertitude induite par la différence de distribution entre les données d'entraînement et les données de test au sein de la section 1.2.4.

1.2.3 Incertitude des entrées

Cette catégorie de méthodes tente de propager la distribution d'une variable aléatoire à travers un réseau de neurones entraîné, au temps de l'inférence. Ce pan de recherche est assez peu fourni et peu étudié par la communauté de l'apprentissage profond. Nous pouvons cependant noter deux grandes familles de méthodes que nous détaillons ci-après.

Propagation de l'incertitude sous hypothèse gaussienne

Ce type d'approches propage l'incertitude de l'entrée en modélisant les distributions d'entrée et de sortie comme étant gaussiennes. Cette classe de méthode se réduit alors à l'estimation des deux premiers moments de la distribution de sortie. Suivant cette approche, Astudillo and Neto (2011) propagent les deux premiers moments (à travers des couches non linéaires) couche par couches en utilisant l'échantillonnage UT (Julier and Uhlmann, 2004). Dans leur travail, Abdelaziz et al. (2015) utilisent une approche similaire en proposant une propagation directement à travers le réseau complet utilisant l'échantillonnage UT mais également en proposant une propagation analytique des moments à travers la fonction d'activation sigmoïde. Ensuite, Gast and Roth (2018) propagent les deux premiers moments analytiquement tout en supposant des matrices de covariance diagonales, négligeant ainsi les corrélations au sein des couches (appelé LPN pour Lightweight Probabilistic Network). On note également le travail de Titensky et al. (2018) propageant les deux premiers moments couche par couche à l'aide du filtrage de Kalman étendu (EKF).

À retenir

L'ensemble de ces méthodes repose sur l'hypothèse gaussienne des différents sorties considérées au détriment de la précision de l'estimation. En effet, lorsqu'il s'agit d'étudier un bruit d'entrée avec une structure de matrice de covariance complexe, l'hypothèse gaussienne unimodale sur la sortie du réseau de neurone n'est pas satisfaisante, la densité de sortie étant souvent multimodale ou asymétrique. Ainsi, la quantification de l'incertitude de la prédiction induite par la distribution d'entrée ne peut être résumée par une seule estimation (Cox and O'Hagan, 2022) (par exemple l'écart-type), mais plutôt par l'ensemble de sa PDF qui, lorsqu'elle est estimée avec précision, permet, entre autres, de dériver un intervalle de prédiction de $\alpha\%$, PI_α, à l'intérieur duquel tombe une nouvelle prédiction avec une probabilité $\alpha\%$.

Propagation de l'incertitude en utilisant des mixtures de gaussiennes.

La deuxième catégorie s'affranchit de l'hypothèse gaussienne en utilisant une distribution de mélange gaussien (MG). Une première tentative de propagation d'un MG dans un réseau de neurones $f\colon \mathbb{R}^d \to \mathbb{R}^{d'}$ a été réalisée par Zhang and Shin (2021). Cette méthodologie repose le paradigme Split&Merge (Sorenson and Alspach, 1971) et sur le calcul d'un critère de division basé sur la divergence de Kullback-Lieber (Kullback and Leibler, 1951). Ce travail, concurrent du nôtre, fait l'objet d'une analyse approfondie au sein du chapitre 2 exposant notre proposition alternative (Monchot et al., 2023).

1.2.4 Incertitude de distribution

Une dernière catégorie d'incertitude, beaucoup moins étudiée dans l'état de l'art, est celle de l'incertitude de distribution. Ce type d'incertitude essaye de quantifier la part de l'incertitude prédictive induite par un décalage de distribution entre les données d'entraînements et les données rencontrées lors de l'inférence.

La décomposition présentée au sein de l'équation 1.2 de l'incertitude prédictive peut se réécrire de la façon suivante :

$$p\left(\mathbf{y}^* \mid x^*, \mathcal{D}_N\right) = \int \underbrace{p(\mathbf{y}^* \mid \mu)}_{\text{Données}} \underbrace{p\left(\mu \mid \mathbf{x}^*, \theta\right)}_{\text{Distribution}} \underbrace{p(\theta \mid \mathcal{D}_N)}_{\text{Modèle}} d\mu d\theta. \tag{1.22}$$

Dans cette modélisation, l'incertitude prédictive est composée de l'incertitude de modèle (épistémique), influant sur l'incertitude de distribution, elle-même influant sur la quantification de l'incertitude des données (aléatoire) (Malinin and Gales, 2018). Ces méthodes permettent notamment

de détecter les données hors de la plage de fonctionnement (échantillons OOD). On note notamment le travail de Malinin and Gales (2018) (on peut également citer le travail concurrent et analogue (Sensoy et al., 2018)), qui développent le concept de "Réseaux priors".

L'équation 1.22 peut être réécrite de la façon suivante :

$$\int \mathrm{p}\left(\mathbf{y}^* \mid \boldsymbol{\mu}\right) \left[\int \mathrm{p}\left(\boldsymbol{\mu} \mid \mathbf{x}^*, \boldsymbol{\theta}\right) \mathrm{p}(\boldsymbol{\theta} \mid \mathcal{D}_N) d\boldsymbol{\theta}\right] d\boldsymbol{\mu} = \int \mathrm{p}\left(\mathbf{y}^* \mid \boldsymbol{\mu}\right) \mathrm{p}\left(\boldsymbol{\mu} \mid \mathbf{x}^*, \mathcal{D}_N\right) d\boldsymbol{\mu} \tag{1.23}$$

Ici, après marginalisation des paramètres θ, le modèle est redéfini par la distribution $p\left(\mathbf{y}^* \mid \mu\right)$ dont les paramètres suivent la distribution $\mathrm{p}\left(\boldsymbol{\mu} \mid \boldsymbol{x}^*, \mathcal{D}_N\right)$, conditionné par rapport à la base de données d'apprentissage $\mathcal{D}_N$ mais également de la donnée d'entrée $\mathbf{x}^*$. Les auteurs proposent alors de placer une distribution de Dirichlet sur les distributions paramétrisant un simplexe, $\mathrm{p}\left(\boldsymbol{\mu} \mid \boldsymbol{x}^*, \mathcal{D}_N\right)$:

$$\mathrm{p}\left(\boldsymbol{\mu} \mid \boldsymbol{x}^*, \mathcal{D}_N\right) = \mathrm{Dir}(\boldsymbol{\mu} \mid \boldsymbol{\alpha}) = \frac{\Gamma\left(\alpha_0\right)}{\prod_{c=1}^{K} \Gamma\left(\alpha_c\right)} \prod_{c=1}^{K} \mu_c^{\alpha_c - 1}, \quad \alpha_c > 0, \alpha_0 = \sum_{c=1}^{K} \alpha_c \tag{1.24}$$

où α_0 est appelé la précision de la distribution de Dirichlet.

Les distributions de Dirichlet, tout comme la fonction Softmax, paramétrisent une distribution sur un simplexe. Nous rappelons qu'un simplexe, $\mathcal{S}$, est l'ensemble entièrement défini par une collection de points $\{p_i\}_{0 \leq i \leq k}$ (linéairement indépendants) comme :

$$\mathcal{S} = \left\{ \mu_0 p_0 + \cdots + \mu_k p_k \mid \sum_{i=0}^{k} \mu_i = 1, \quad \mu_i \geq 0 \right\} \tag{1.25}$$

Ainsi, le simplexe élémentaire dans le plan est un triangle, et dans l'espace, le simplexe élémentaire est un tétraèdre. Le réseau de neurones fournit alors comme prédiction le vecteur α définissant les paramètres de concentration de la distribution de Dirichlet. La figure 1.4 (reprise de (Malinin and Gales, 2018)) représente le résultat attendu :

FIGURE 1.4 – : Comportement désiré d'une distribution sur les distributions (Figure reprise de Malinin and Gales (2018)). Cas d'une prédiction "confiante" (**gauche**), cas d'une entrée à haute incertitude aléatoire (**milieu**) et cas d'une entrée en dehors de la zone de fonctionnement (**droite**).

Si le réseau est confiant dans sa prédiction, alors la distribution sera concentrée vers un coin du simplexe, si la donnée d'entrée se trouve dans une zone de forte incertitude aléatoire alors la distribution sera concentrée au centre du simplexe, alors que pour une donnée d'entrée en dehors de la plage de fonctionnement (OOD), la distribution de sortie sera "diffuse" sur tout le simplexe. Cette dernière figure traduit la volonté d'avoir une entropie prédictive élevée lorsque la donnée d'entrée est située en dehors de la plage de fonctionnement définie par la base de données d'apprentissage. Ce type de méthode nécessite également l'établissement d'une fonction de coût adaptée, et qui fait l'objet de recherche de plusieurs articles scientifiques (Nandy et al., 2020).

À retenir

Ce travail met en partie en évidence la nécessité de pouvoir estimer l'incertitude au-delà de la dichotomie classiquement établie entre épistémique et aléatoire. Modéliser l'incertitude en extrayant le concept de plage de fonctionnement de l'incertitude épistémique peut permettre une meilleure compréhension de l'incertitude prédictive et la mise en place de solutions en accord avec cette décomposition. Par exemple, dans un cadre réglementaire et/ou métrologique, des structures de contrôle peuvent être mises en place afin de s'assurer que les données d'entrée à l'inférence font partie de la plage de fonctionnement spécifiée par $\mathcal{D}_N$.

1.2.5 Méthodes basées sur un unique réseau de neurones déterministe

Les méthodes de quantification d'incertitude peuvent être gourmandes en temps d'entraînement, en allocation mémoire mais également en temps de calcul à l'inférence. Pour pallier ces problématiques, plusieurs méthodes ont vu le jour afin d'estimer l'incertitude prédictive à l'aide d'un simple réseau de neurones déterministe, comme présenté au sein de la figure 1.5. Parmi ce groupe de méthodes, on souligne notamment les méthodes qui estiment l'incertitude "de modèle" en se basant sur l'évaluation du gradient autour du point estimé lors de l'apprentissage (Oberdiek et al., 2018; Lee and AlRegib, 2020) ou encore les méthodes de régression de quantiles (Tagasovska and Lopez-Paz, 2019; Akrami et al., 2021).

Ici, nous voulons mettre en avant le travail de (Van Amersfoort et al., 2020). Dans un cadre de classification, les auteurs proposent de se reposer sur les fonctions de base radiale (RBF). Ainsi, la couche de softmax est remplacé par une couche linéaire de sortie permettant "d'apprendre" une matrice $\mathbf{W}_c$ pour chacune des classes c. Les auteurs proposent alors d'utiliser la fonction RBF suivante :

$$K_c\left(f(\mathbf{x};\theta), \mathbf{e}_c\right) = \exp\left[-\frac{\frac{1}{n}\left\|\mathbf{W}_c f(\mathbf{x};\theta) - \mathbf{e}_c\right\|_2^2}{2\sigma^2}\right] \tag{1.26}$$

où $f(\mathbf{x};\theta)$ est la sortie du modèle privée du softmax, n est le nombre de classes, $\mathbf{e}_c$ le centroïde de la classe c, et σ est un hyperparamètres appelée parfois "échelle de longueur".

Ainsi, la classe prédite est définie comme étant celle présentant la plus petite distance entre la prédiction du modèle et son centroïde, traduit par la plus grande valeur de K_c : $\mathbf{y}^* = \arg\max_c K_c\left(f_\theta(\mathbf{x}), \mathbf{e}_c\right)$.

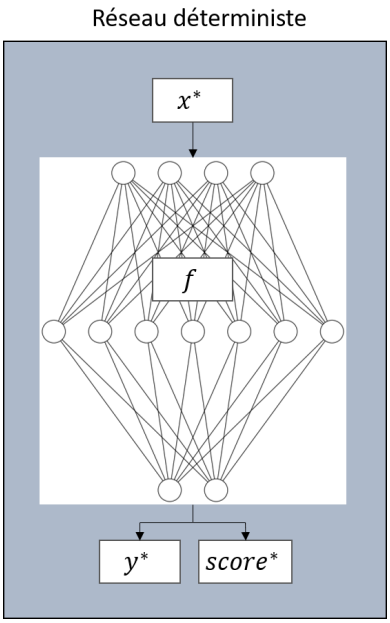


FIGURE 1.5 – Incertitude par un réseau déterministe.

La valeur de l'incertitude associée est, quant à elle, définie comme étant cette même distance entre la prédiction et le centroïde $\max_c K_c\left(f_\theta(\mathbf{x}), \mathbf{e}_c\right)$. Cette méthode innovante permet d'obtenir à faible coût une prédiction ainsi qu'une valeur d'incertitude associée tout en maintenant une bonne performance globale au regard des réseaux softmax. Elle reste cependant spécifique aux tâches de classification.

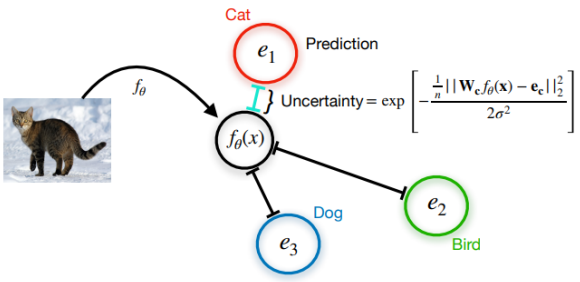


FIGURE 1.6 – Représentation schématique du réseau DUQ. (Figure reprise de (Van Amersfoort et al., 2020))

À retenir

Les méthodes reposant sur un simple réseau déterministe sont la plupart du temps rapides au temps d'inférence et ne nécessitent aucune augmentation de la mémoire requise. En revanche, l'interpretabilité de l'incertitude quantifiée reste difficile. Il n'est pas souvent clair quelle source d'incertitude quantifie la méthode considérée. Dans un cadre métrologique, il est préférable de s'appuyer sur des méthodes plus "classiques".

L'annexe B présente également quelques méthodes de réductions de la complexité calculatoire pouvant être utilisées lors de la quantification de l'incertitude des réseaux de neurones.

1.2.6 Une vue d'ensemble

La figure 1.7 présente une vue d'ensemble des méthodes précédemment présentées.

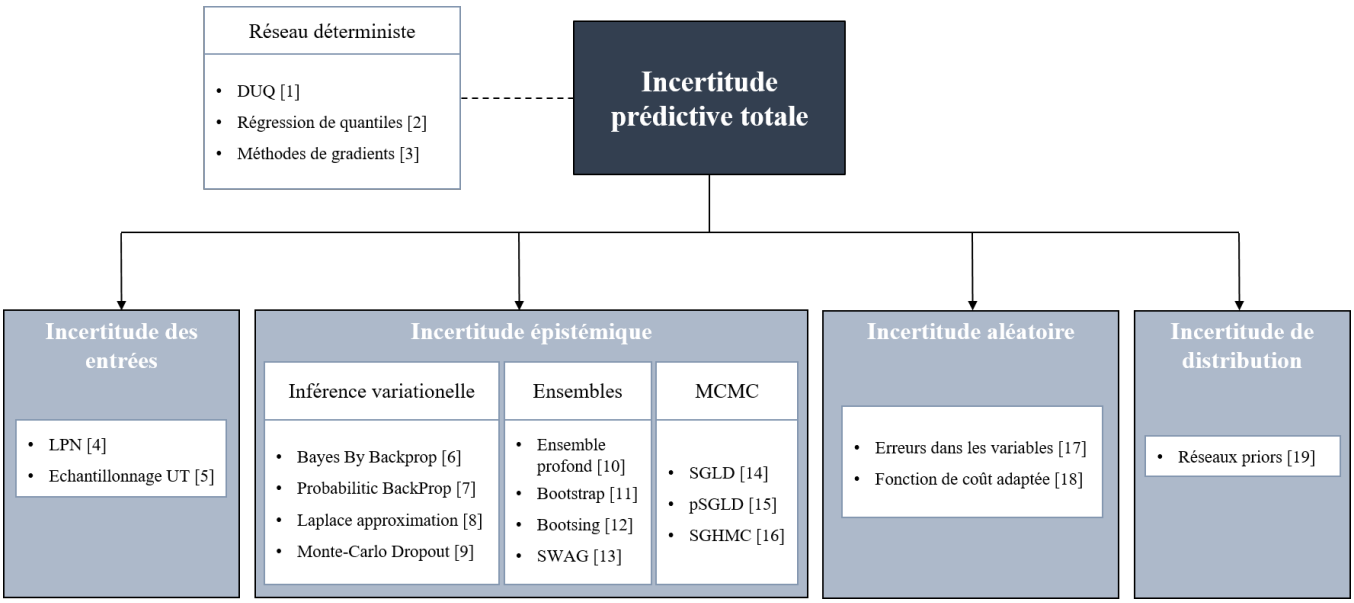


FIGURE 1.7 – Vue d'ensemble des méthodes de l'état de l'art pour la quantification d'incertitudes au sein de réseaux de neurones. [1] : Van Amersfoort et al. (2020) [2] : Tagasovska and Lopez-Paz (2019); Akrami et al. (2021) [3] : Oberdiek et al. (2018); Lee and AlRegib (2020) [4] : Gast and Roth (2018) [5] : Julier and Uhlmann (2004)Abdelaziz et al. (2015) [6] : Blundell et al. (2015) [7] : Hernández-Lobato and Adams (2015) [8] : Ritter et al. (2018) [9] : Gal and Ghahramani (2016) [10] : Lakshminarayanan et al. (2017) [11] : Breiman (1996) [12] : Friedman (2001) [13] : Wilson and Izmailov (2020) [14] : Welling and Teh (2011) [15] : Li et al. (2016) [16] : Chen et al. (2014) [17] : Martin and Elster (2022) [18] : Kendall and Gal (2017) [19] : Malinin and Gales (2018); Sensoy et al. (2018). Les pointillés indiquent que les méthodes présentées dans les articles adressent plus d'une source d'incertitude.

1.3 Le paradigme épistémique / aléatoire remis en cause

Nous venons de voir que la communauté de l'apprentissage profond s'était largement emparé de la problématique de la quantification de l'incertitude prédictive au sein d'algorithmes d'apprentissage statistique. Cette quantification est en majorité axée sur l'estimation de deux types d'incertitude : l'incertitude épistémique et l'incertitude aléatoire. Ici, nous défendons l'idée que cette taxonomie n'est pas satisfaisante au regard des exigences de modélisation de l'incertitude en métrologie.

Il s'agit d'une classification de sources d'incertitude, qui peut être effectuée a posteriori d'un bilan d'incertitude (Der Kiureghian and Ditlevsen, 2009). En effet, cette dichotomie, permet de répondre avant tout à la question suivante :

*Quelle est la **part** de l'incertitude prédictive pouvant être **réduite** en **augmentant** le nombre de **données** d'apprentissage ?*

Il ne s'agit donc pas de sources d'incertitude à proprement parler mais plutôt d'une vision particulière de cette dernière. Der Kiureghian and Ditlevsen (2009) mettent également en avant le fait que, selon le cas de figure envisagé, certaines sources d'incertitude pourraient être classées comme aléatoire ou épistémique, certaines sources pouvant même avoir certaines de ses composantes de nature aléatoire et certaines de nature épistémique.

Enfin, cette classification ne semble pas offrir un cadre sémantique rigide et bien établi, comme peut le demander un cadre métrologique. On note notamment, tout au long de la littérature, une grande disparité d'appellation des sources d'incertitude considérées. A titre d'exemple, la notion d'incertitude épistémique peut être également rencontrée sous les appellations d'incertitude des **paramètres**, du **modèle** ou encore du **réseau**. L'incertitude aléatoire quant à elle peut également être rencontrée sous les dénominations d'incertitude des **données**, des **entrées** ou encore des **observations**.

La dualité aléatoire/épistémique est alors un outil utile afin d'appréhender un pan de l'incertitude qui se focalise sur l'impact du manque de connaissance du modèle lors de l'apprentissage sur son doute à la prédiction. Elle ne constitue néanmoins pas une méthodologie bien établie afin de faire un bilan complet de l'incertitude prédictive d'un réseau de neurones. On note notamment, que d'autres travaux n'utilisent pas uniquement cette dichotomie épistémique/aléatoire. Parmi ceux-ci, nous pouvons citer le travail original de Der Kiureghian and Ditlevsen (2009) faisant la liste des sources d'incertitude au sein d'une modélisation probabiliste, celui de Gawlikowski et al. (2021) qui présente également les sources sous le terme de "facteur" et enfin le tout récent travail de Gruber et al. (2023). L'établissement d'une sémantique rigoureuse nous parait primordiale afin de mener à

bien un bilan d'incertitude compréhensible et de qualité.

Dans la suite de ce chapitre, nous présentons l'approche classique en métrologie de l'évaluation de l'incertitude (cf. section 1.4), avant de présenter notre proposition de classification des sources d'incertitudes au sein des réseaux de neurones en adoptant une vision métrologique (cf. section 1.5).

1.4 L'approche métrologique de l'évaluation de l'incertitude

La métrologie joue un rôle prépondérant dans l'analyse et l'étude des notions d'incertitude. Communément définie comme la science de la mesure (du grec ancien "mètre" signifiant "mesure" et "traité"), la métrologie se concentre sur l'estimation d'une valeur d'intérêt, appelée *mesurande*, dans des conditions spécifiques. Cette estimation est réalisée à l'aide d'un instrument de mesure, tel qu'une balance de Roberval pour la mesure de masse, et d'une référence, telle qu'un jeu de poids, avec lequel le *mesuré* est comparé. En métrologie, une mesure consiste donc à comparer une propriété d'un objet à une valeur de référence.

Ainsi, toute mesure effectuée, qu'il s'agisse d'une grandeur physique ou autre, est représentée par un résultat numérique accompagné d'une unité et d'une incertitude. Cette incertitude traduit la qualité de la mesure et reflète directement sa fiabilité. Selon la définition donnée dans (De Bièvre, 2012), l'incertitude est "un paramètre **non négatif** caractérisant la dispersion des valeurs attribuées à un **mesurande**, à partir des **informations utilisées**".

L'incertitude associée à la mesure permet donc de réaliser des comparaisons, que ce soit par rapport à une norme, à d'autres résultats expérimentaux ou à un étalon. Le besoin de comparaison a notamment conduit à l'établissement d'une science normative de la quantification de l'incertitude. Afin de pouvoir se comparer en termes d'incertitude, il est essentiel que tous les scientifiques et métrologues mesurent l'incertitude selon une méthodologie commune. Une autre conséquence importante de ce besoin est l'établissement d'une sémantique partagée permettant d'estimer les mêmes phénomènes de manière cohérente et analogue.

La métrologie fournit un cadre rigoureux pour l'estimation et la gestion de l'incertitude, garantissant ainsi la fiabilité des mesures et la comparabilité des résultats. En adoptant une approche métrologique dans nos travaux, nous pourrons assurer la précision, la cohérence et la crédibilité des résultats obtenus par algorithmes d'apprentissage profond.

La communauté mondiale des métrologues se réfère majoritairement à deux guides complémentaires pour l'estimation de l'incertitude de mesure :

— le Guide pour l'expression de l'incertitude de mesure (GUM) (ENV13005, 1999).
— Le Supplément 1 du GUM (GUMS1) (BIPM et al., 2008).

Ces guides établissent un cadre précis pour l'évaluation de l'incertitude. Ils définissent de manière rigoureuse les procédés de mesure, les règles générales pour l'évaluation et l'expression de l'incertitude, ainsi que les bonnes pratiques à travers des exemples de référence, à l'image de la mesure simultanée d'une résistance et d'une réactance. Il est essentiel de noter que ces guides de bonnes pratiques sont conçus pour être applicables de la "boutique du commerçant à la recherche fondamentale"(ENV13005, 1999). Ainsi, le processus de quantification de l'incertitude ne doit pas dépendre du type spécifique de mesure effectuée, mais doit au contraire être aussi générique que possible. Cette philosophie a été une de nos lignes directrices tout au long de nos travaux de recherche.

D'après le GUM, l'estimation et l'expression de l'incertitude d'une mesure doit être "universelle", "logique en elle-même" et "transférable". L'universalité de la méthode assure qu'elle puisse être appliquée à tous types de mesurages et de données d'entrée de ces derniers, alors que la transférabilité de la méthode assure que l'incertitude estimée d'un premier résultat de mesure puisse être directement utilisée comme donnée d'entrée d'un second mesurage utilisant les sorties du premier.

Le processus de quantification de l'incertitude suit généralement une démarche en 4 étapes :

1. L'**analyse** du processus de mesure : définition du mesurande, établissement du modèle mathématique, noté généralement f, et le calcul du résultat de mesure à l'aide des grandeurs d'entrée,
2. L'établissement de la liste des **sources d'incertitude** ainsi que l'estimation de leur incertitude type,
3. La **propagation** des sources d'incertitude au sein du modèle pour obtenir l'incertitude composée.
4. L'**expression finale** du résultat en déterminant notamment l'incertitude élargie.

Plus précisément, lors d'une mesure d'une valeur $\mathbf{y}$, on obtient un estimé, noté $\hat{\mathbf{y}}$, obtenu à travers un procédé de mesure noté f et qui peut dépendre de plusieurs (m) variables aléatoires d'entrées X_i. On a alors la relation suivante $\hat{\mathbf{y}} = f(\mathbf{x}_1, ..., \mathbf{x}_m)$; où les $\mathbf{x}_i$ sont des réalisations des variables aléatoires X_i. Le GUM et le GUM S1 proposent une méthodologie afin d'estimer les incertitudes types des différentes variables aléatoires misent en jeu lors de la mesure ainsi que leur propagation au sein du modèle f pour finalement obtenir l'incertitude finale.

Guide pour l'expression de l'incertitude de mesure : Ce guide propose de modéliser les incertitudes types $u(X_i)$, des différentes sources par le biais de leur écart-type σ_{X_i}, ainsi que de leurs

covariances $u(X_i, X_j)$. Par la suite, la propagation de ces incertitudes types à travers le modèle f est effectuée en utilisant la loi de propagation des variances :

$$\mathrm{Var}(\hat{\mathbf{y}}) = u^2(\hat{\mathbf{y}}) = \sum_i \left[\frac{\partial f}{\partial \mathbf{x}_i}\right]^2 u^2(\mathbf{x}_i) + \sum_i \sum_{i \neq j} \frac{\partial f}{\partial \mathbf{x}_i} \frac{\partial f}{\partial x_j} u(\mathbf{x}_i, \mathbf{x}_j) \tag{1.27}$$

La validité de cette propagation dépend de la validité de deux hypothèses sous-jacentes :

— la linéarité locale du modèle f, permettant d'effectuer un développement de Taylor à l'ordre 1 (et donc de négliger les dérivées d'ordre supérieur),

— toutes les dérivées partielles sont constantes au voisinage du point étudié.

L'incertitude finale est alors exprimée en fonction de la variance $\mathbb{V}(\hat{\mathbf{y}})$.

GUM Supplément 1 : Le Supplément 1 du GUM (Guide pour l'expression de l'incertitude de mesure) permet de caractériser l'incertitude des variables aléatoires d'entrée par le biais de leur densité de probabilité. Ces densités sont ensuite échantillonnées, ce qui permet d'évaluer la fonction f et d'obtenir des échantillons de la densité de probabilité de la variable $\hat{Y}$. Cette approche de propagation des distributions repose donc sur les techniques de Monte-Carlo (Mooney, 1997).

Estimer la densité de probabilité de sortie permet d'en extraire les éléments statistiques pertinents pour exprimer l'incertitude du résultat de mesure. Il s'agit entre autres des deux premiers moments, la moyenne et la variance, mais également de dériver un intervalle de prédiction de $\alpha\%$, PI_α, à l'intérieur duquel tombe une nouvelle prédiction avec une probabilité $\alpha\%$. On note que dans le cas de distributions multimodales ou asymétriques, l'expression de l'incertitude à travers la densité de probabilité et/ou un intervalle de couverture est à privilégier.

Remarque 5.

Dans le cas d'une distribution gaussienne de moyenne μ et d'écart type σ, l'intervalle de prédiction à 95% est directement obtenu a l'aide de la formule suivante : $PI_\alpha = [\mu - 1.96\sigma; \mu + 1.96\sigma]$.

1.5 Incertitude en apprentissage profond : un point de vue métrologique

Le succès des réseaux de neurones découle de leur dimension immense, pouvant contenir des millions, voire des milliards, de paramètres (par exemple, à sa sortie, GPT-3 (Brown et al., 2020) fut le plus grand réseau de neurones jamais entraîné avec environ 175 milliards de paramètres). Cette

taille est associée à celle des bases de données utilisées, ainsi qu'à leur capacité à effectuer des tâches complexes sur des entrées de très haute dimension, à l'instar des images pouvant contenir plusieurs millions de pixels.

La quantification de l'incertitude prédictive devient de plus en plus complexe à mesure que la dimension du problème augmente. Par conséquent, les approches classiques de quantification de l'incertitude basées sur le Guide pour l'expression de l'incertitude de mesure (GUM) et son Supplément 1 (GUMS1) ne peuvent être appliquées de manière exhaustive. En effet, en plus de la complexité en termes de coût en mémoire et de temps de calcul, rendant les approches par Monte-Carlo difficiles à mettre en œuvre, les réseaux de neurones se caractérisent par des fonctions fortement non linéaires, tant dans l'espace des entrées que dans l'espace des paramètres. Cela rend difficile l'utilisation des approches fondées sur l'estimation des variances et leur propagation. Enfin, le cadre standard établi dans le GUM ne traite pas des problématiques liées à l'entraînement à partir d'une base de données d'exemples. Pour toutes ces raisons, il a été nécessaire de développer des méthodologies adaptées aux algorithmes d'apprentissage profond afin de quantifier de manière fiable l'incertitude de prédiction. Ces méthodes permettent également de mieux comprendre les architectures des réseaux de neurones, de les améliorer et de développer des approches visant à renforcer la fiabilité et la sécurité de ces systèmes.

Nous souhaitons dresser ici une liste des diverses sources d'incertitude présentes dans le processus conduisant à une prédiction d'un réseau de neurones, tel qu'illustré dans la figure 1.1. Cette étude vise à suivre les normes établies en métrologie.

Afin de les expliciter, nous nous appuyons sur une expérience dont les dimensions permettent une étude simplifiée. Cette expérience consiste à entraîner un perceptron multi-couche simple, muni de la fonction d'activation Unité Linéaire Rectifiée (ReLU) (Nair and Hinton, 2010), et à accomplir la tâche de régression de la fonction $y = \sin x + x^{\frac{1}{3}}$.

Par la suite, nous ferons varier différents hyperparamètres de l'apprentissage afin de mettre en évidence une source spécifique. Ces hyperparamètres seront précisés pour chaque expérience considérée. Les valeurs par défaut à toutes les expériences sont présentées dans le tableau 1.2.

Hyperparamètres	Valeur
Taille de batch	128
Optimiseur	SGD
Pas d'apprentissage	0.01
Activation	ReLU
Répétition	10
Epochs	200
Valeur min/max	0.0 / 10.0

TABLE 1.2 – Valeurs par défaut des hyperparamètres de l'Exemple illustratif

1.5.1 Incertitude des entrées

Dans un processus de mesure, mais également en apprentissage statistique, il est courant que les données d'entrée d'un algorithme à l'inférence soient issues de capteurs physiques. Comme toutes données acquises par des capteurs, la donnée "vraie" est assujettie à divers bruits, et nous n'observons qu'une réalisation de la variable aléatoire associée. Ainsi, l'**incertitude des entrées** correspond à la distribution de l'entrée $X^* \sim P_{X^*}$. L'objectif ici sera donc de pouvoir propager la distribution de l'entrée courante au sein du réseau de neurones afin d'obtenir la distribution de sortie induite, i.e. estimer $Y^* \sim P_{Y^*}$ avec $\mathbf{y}^* = \hat{f}_{\mathcal{M}}(\mathbf{x}^*, \hat{\theta}_{\mathcal{D}_N,opt})$

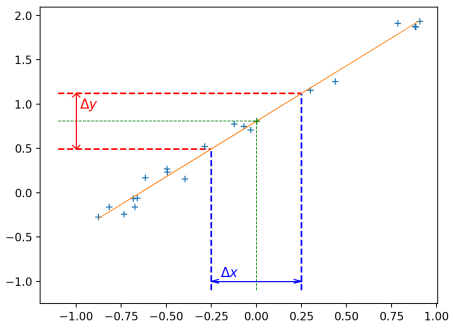


FIGURE 1.8 – Régression linéaire simple où une incertitude de $\Delta\mathbf{x}^*$ sur l'entrée verte induit une incertitude de $\Delta\mathbf{y}^*$ sur la prédiction.

Pour illustrer ce point, prenons le cas simple d'une régression linéaire, où le modèle est donné par l'équation suivante :

$$\mathbf{y} = \theta^T \mathbf{x} + \epsilon \tag{1.28}$$

avec $\mathbf{y} \in \mathbb{R}$, $\mathbf{x} \in \mathbb{R}^d$, $\theta \in \mathbb{R}^d$ et $\epsilon \sim \mathcal{N}(0, \sigma^2)$.

Pour une base de données d'exemples $\mathcal{D}_N = (\mathbf{x}_i, \mathbf{y}_i)$ indépendants et identiquement distribués, l'estimateur du maximum de vraisemblance $\hat{\theta}_{\text{MLE}}$ est donné par :

$$\hat{\theta}_{MLE} = \left(A^T A\right)^{-1} A^T \mathbf{b} \tag{1.29}$$

où A est la matrice contenant l'ensemble des données d'entrée $\mathbf{x}_i$ et $\mathbf{b}$ le vecteur contenant les sorties $\mathbf{y}_i$.

Pour un nouveau point d'entrée $\mathbf{x}^*$ et son incertitude associée, représentée par $\Delta\mathbf{x}^*$, on obtient :

$$\begin{aligned} \mathbf{y}^* + \Delta\mathbf{y}^* &= \hat{\theta}_{MLE}^T \left(\mathbf{x}^* + \Delta\mathbf{x}^*\right) + \epsilon & (1.30) \\ &= \left(\hat{\theta}_{MLE}^T \mathbf{x}^* + \epsilon\right) + \hat{\theta}_{MLE}^T \Delta\mathbf{x}^* & (1.31) \end{aligned}$$

La figure 1.8 présente ce phénomène, où une incertitude $\Delta\mathbf{x}^*$ sur le point de test (croix verte) induit une incertitude $\Delta\mathbf{y}^*$ sur la prédiction finale. On appelle ce type d'incertitude : l'**incertitude des entrées**. Dans cet exemple, il est pertinent de représenter l'incertitude en fonction d'un delta Δ ou d'un écart-type σ du fait de la linéarité du modèle considéré. Ce n'est pas le cas pour les réseaux de neurones. Formellement, on obtient :

$$u^2_{input}(\mathbf{y}^*) = \mathbb{V}_{\mathbf{x}^* \sim p(X^*)}(\hat{f}_{\mathcal{M}}(\mathbf{x}^*, \hat{\theta}_{\mathcal{D}_N,opt})) \tag{1.32}$$

Exemple illustratif : Incertitude des entrées

Voici les hyperparamètres spécifiques pour expliciter l'incertitude prédictive induite par l'incertitude de l'entrée courante :

Hyperparamètres	Valeur
Nombre de données	10000
Nombre de couches	5
Nombre de neurones par couche	64
Nombre d'échantillons propagés	10000

La figure 1.9 présente l'incertitude prédictive induite par l'incertitude d'entrée en présentant la densité de probabilité de sortie pour une entrée corrompue par un bruit gaussien $\mathbf{x}^* \sim \mathcal{N}(\mu, 0.1)$:

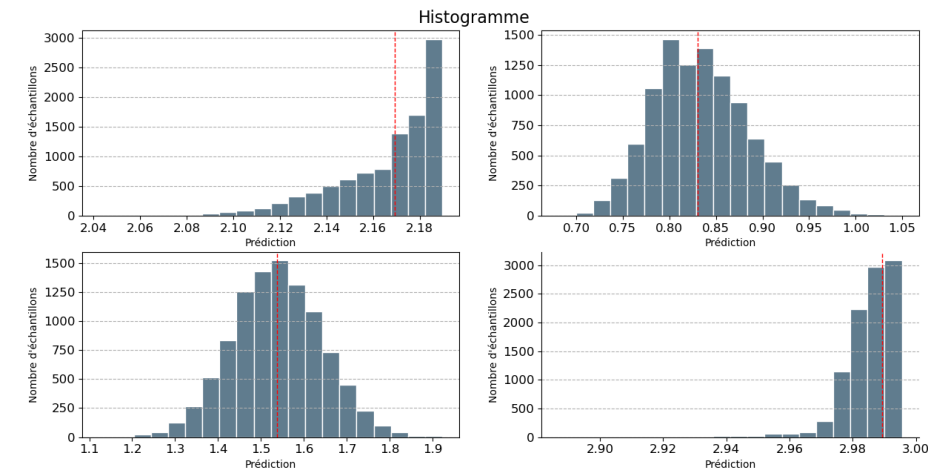


FIGURE 1.9 – Histogramme de sortie avec $\mathbf{x}^* \sim \mathcal{N}(\mu, 0.1)$ et $\mu = 2.0$ **(en haut à gauche)**, $\mu = 4.0$ **(en haut à droite)**, $\mu = 6.0$ **(en bas à gauche)** et $\mu = 8.0$ **(en bas à droite)** .

Pour $\mu = 2.0$ et $\mu = 8.0$, les distributions de sorties ne sont pas gaussiennes, traduisant le caractère non linéaire du réseau de neurones. Dans ce cas précis, une propagation simple des variances engendrerait une mauvaise estimation de l'incertitude de sortie.

1.5.2 Incertitude des paramètres

L'**incertitude des paramètres** est une catégorie de sources d'incertitude traduisant une variabilité dans les paramètres du réseau de neurones pour une tâche considérée. Nous pouvons identifier deux sources d'incertitude induisant cette variabilité.

Incertitude d'échantillonage

L'**incertitude d'échantillonnage** est l'incertitude induite par la composition de la base de données d'apprentissage. Cette source d'incertitude est induite par le manque de connaissance dans la base de données d'entraînement, provenant directement du nombre limité d'exemples à disposition.

Pour quantifier cette source d'incertitude, le cadre bayésien constitue une boite à outils adaptée à cette problématique en proposant une méthode systématique, rigoureuse, compréhensive et permettant d'incorporer de la connaissance a priori de l'observation des données. En effet, cette incertitude peut être modélisée par le postérieur des paramètres conditionné à la base d'entraînement $p(\theta \mid \mathcal{D}_N)$, incorporant une connaissance a priori, $p(\theta)$, ainsi que les observations, par la règle de Bayes :

$$p(\theta \mid \mathcal{D}_N) = \frac{p(\mathcal{D}_N \mid \theta)\, p(\theta)}{p(\mathcal{D}_N)} \tag{1.33}$$

En augmentant le nombre de données N de la base, la distribution des paramètres conditionnellement à la base d'entraînement deviendrait plus concentrée autour de la valeur "optimale" des paramètres θ.

Reprenons l'exemple de la simple régression linéaire. On rappelle que le modèle est défini par :

$$\mathbf{y} = \theta^T \mathbf{x} + \epsilon \tag{1.34}$$

avec $\mathbf{y} \in \mathbb{R}$, $\mathbf{x} \in \mathbb{R}^d$, $\theta \in \mathbb{R}^d$ et $\epsilon \sim \mathcal{N}(0, \sigma^2)$.

Donc $Y \mid X, \theta \sim \mathcal{N}(\theta^T \mathbf{x}, \sigma^2)$. En plaçant un prior gaussien sur le paramètre $\theta \sim \mathcal{N}(\mu_{\theta_0}, \Sigma_{\theta_0})$, on obtient le postérieur suivant :

$$\theta \mid X, Y \sim \mathcal{N}(\mu_N, \Sigma_N) \tag{1.35}$$

où $\Sigma_N = \left(\Sigma_{\theta_0}^{-1} + (\sigma^2)^{-1} A^T A\right)^{-1}$ et $\mu_N = \Sigma_N \left(\Sigma_{\theta_0}^{-1} \mu_{\theta_0} + (\sigma^2)^{-1} A^T A\right)$

Dans ce cas, il est aisé de montrer que le postérieur se concentre autour de la solution du maximum de vraisemblance lorsque le nombre de données observées N augmente. La conséquence

direct étant la réduction de la variance de cette densité. La matrice $A \in \mathbb{R}^{d+1 \times d}$ contenant l'ensemble des observations $(\mathbf{x}_i)_{1 \leq i \leq N}$, le terme $A^T A$ augmente à mesure que N augmente en tant que somme de carrés. σ et Σ_{θ_0} étant constant, on a $\left(\Sigma_{\theta_0}^{-1} + (\sigma^2)^{-1} A^T A\right)$ qui "augmente" et donc $\Sigma_N = \left(\Sigma_{\theta_0}^{-1} + (\sigma^2)^{-1} A^T A\right)^{-1}$ qui "diminue". Cette diminution est traduit par de plus faibles valeurs propres de la matrice de variance covariance du postérieur des paramètres conditionnellement aux données d'entraînement.

Ce phénomène de concentration du postérieur est largement étudié dans la communauté statistique. On peut citer notamment l'ouvrage (Bishop and Nasrabadi, 2006, Sect. 3.4) qui discute l'influence de la complexité du modèle considéré sur la concentration du postérieur. On note également un pan de la littérature proposant des vitesses de concentration du postérieur dans certains cas simples, tel que les modèles linéaires en haute dimension (Martin et al., 2017), ou encore dans un processus de Dirichlet pour un prior consistant en un mélange fini. Enfin, il existe certaines conditions afin d'observer une concentration du postérieur dans un cadre d'inférence bayésienne. Dans l'approche classique, le modèle considéré doit être identifiable, c'est-à-dire qu'il existe une unique valeur des paramètres correspondant au vrai processus de génération des données. Le processus d'estimation doit également être consistant, c'est-à-dire que les paramètres estimés convergent vers la "vraie" valeur des paramètres quand le nombre des données tend vers l'infini et enfin que, a minima, le support de la distribution a priori contienne cette "vraie" valeur. Ces conditions ne sont, en général, pas vérifiées pour les réseaux de neurones. En effet, les réseaux de neurones présentent de nombreuses permutations possibles des paramètres conduisant au même prédicteur. Enfin, les distributions a priori classiquement utilisées dans les réseaux de neurones bayésiens sont en général inadéquats. De part la haute dimension du problème et le faible nombre relatif de données observées, les priors établis sont "involontairement informatifs" (Wenzel et al., 2020), pouvant engendrer des difficultés quand à la fiabilité de l'estimation du postérieur. Ce phénomène devient de plus en plus marqué à mesure que le nombre de paramètres du réseau augmente.

Remarque 6.

Wasserman (2004, Sec. 11.9) discute des forces et faiblesses du cadre bayésien. Les auteurs soulignent que ce cadre est adapté lorsque de la connaissance a priori est disponible. Cette méthodologie est en revanche compliquée à utiliser pour des problèmes en haute dimension.

Formellement, on obtient :

$$u^2_{sampling}(\mathbf{y}^*) = \mathbb{V}_{\hat{\theta}_{\mathcal{D}_N,opt} \sim p(\theta|\mathcal{D}_N)}(\hat{f}_{\mathcal{M}}(\mathbf{x}^*, \hat{\theta}_{\mathcal{D}_N,opt})) \tag{1.36}$$

On illustre une nouvelle fois au sein de notre exemple illustratif, la notion d'incertitude d'échantillonnage. Cette incertitude peut s'illustrer selon deux visions complémentaires :

— le domaine d'entrée est sous-échantillonné, à l'image d'une règle graduée fournissant des références à intervalles réguliers,
— le domaine d'entrée présente un *trou* sur un sous-domaine (dans l'exemple de la règle, plusieurs graduations consécutives sont effacées).

Exemple illustratif : Incertitude d'échantillonnage

Voici les hyperparamètres spécifiques pour expliciter l'incertitude d'échantillonnage :

Hyperparamètres	Valeur
Nombre de données	Variable
Nombre de couches	5
Nombre de neurones	64

La figure 1.10 présente l'incertitude prédictive induite par l'incertitude d'échantillonnage lorsque le domaine d'entrée est sous-échantillonné. L'expérience consiste à visualiser l'évolution des performances d'un réseau de neurones (dont la complexité permet une calibration presque parfaite pour un nombre de données suffisant) en fonction du nombre de données dans la base d'apprentissage. Pour ces différents entraînements, on maintient un nombre constant de données présentées au réseau (par exemple, l'entraînement avec 10 000 données est effectué sur 200 epochs alors que l'entraînement avec 100 000 données fut effectué avec 20 epochs. La stochasticité de la descente de gradient est fixée grâce à l'établissement d'une graine.

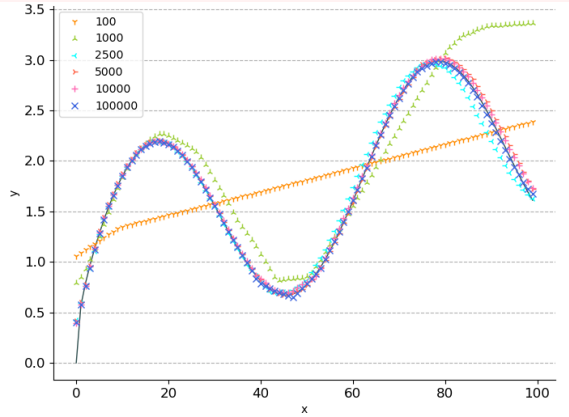


FIGURE 1.10 – Performance sur 100 points de tests du réseau de neurones entraînés sur 100 (**orange**), 1000 (**vert**), 2500 (**cyan**), 5000 (**rouge**), 10 000 (**rose**) et 100 000 données (**violet**).

Pour un nombre suffisant de données (à partir de 2500 points d'apprentissage), les performances des différents entraînements présentent une très faible variation. Il est important de noter que 2500 données pour une tâche en une dimension reste très élevé. On peut imaginer que pour des taches de segmentation d'images, par exemple, le nombre de données équivalent est très faible.

Exemple illustratif : Incertitude d'échantillonnage (suite)

Nous fournissons également au sein de la figure 1.11, la performance moyenne sur 100 points de test de 10 entraînements indépendants munis d'un jeu de données d'entraînement constitué de 2500 points entre 0 et 4.0 et 2500 entre 6.0 et 10.0. L'écart-type des prédictions est présenté par une plage rouge.

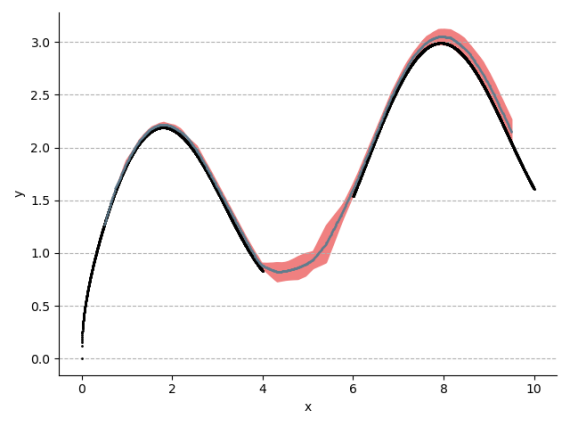


FIGURE 1.11 – Performance moyenne (en bleu) sur 100 points de test de 10 entraînements indépendants. Les points d'entraînement sont tracés en noirs, la zone rouge représente une marge équivalente à 2 fois l'écart-type.

Nous observons ici la performance du réseau considéré lorsque la base d'entraînement est constituée d'un "trou" au milieu de sa plage de fonctionnement. On note que la calibration des réseaux se détériore légèrement à partir de la zone manquante à l'entraînement. L'incertitude traduite par l'écart-type est également plus important à partir de ce moment.

Incertitude de répétabilité

L'**incertitude de répétabilité** est une source d'incertitude classique en métrologie et consiste en l'incertitude induite sur le mesurande lors de répétitions de la même expérience. Dans le cadre d'un réseau de neurones entraîné à l'aide d'une base d'entraînement parfaite (infinie, complète[1] et non bruitée), la complexité de la tâche à apprendre couplée à la limite mémoire du matériel utilisée ne permet pas une représentation parfaite de cette dernière. Ainsi, la stochasticité de l'algorithme d'optimisation conduit le ré-entraînement du même réseau à trouver un autre jeu de poids différent du premier, tout en maintenant une performance quasi-constante sur la tâche considérée, i.e. à une base d'entraînement $\mathcal{D}_N$ fixée. Cette variabilité dans les paramètres induite par la stochasticité de l'optimisation est appelée incertitude de répétabilité.

1. ici, on entend par complète que la valeur de sortie peut être entièrement déterminée par sa donnée d'entrée

Soit $\hat{\theta}^i_{\mathcal{D}_N,opt}$ les paramètres estimés lors de l'entraînement i, on obtient alors :

$$u^2_{repeat}(\mathbf{y}^*) = \mathbb{V}(\hat{f}_{\mathcal{M}}(\mathbf{x}^*, \hat{\theta}^i_{\mathcal{D}_N,opt})) \tag{1.37}$$

Nous avons vu précédemment que cette source d'incertitude n'est pas considérée en tant que telle dans la communauté de l'apprentissage profond. L'incertitude de répétabilité semble difficile à classer dans un paradigme aléatoire/épistémique.

Exemple illustratif : Incertitude de répétabilité

Voici les hyperparamètres spécifiques pour expliciter l'incertitude prédictive induite par l'incertitude des données constituant la base de données d'apprentissage :

Hyperparamètres	Valeur
Nombre de données	5000
Nombre d'époch s	200
Nombre de données de test	100

Cette expérience consiste à visualiser la variabilité des prédictions lors du ré-entraînement du réseau de neurones. La tâche considérée étant très simple pour notre réseau de neurones, on simule une tâche plus complexe en faisant varier le nombre d'époch s.

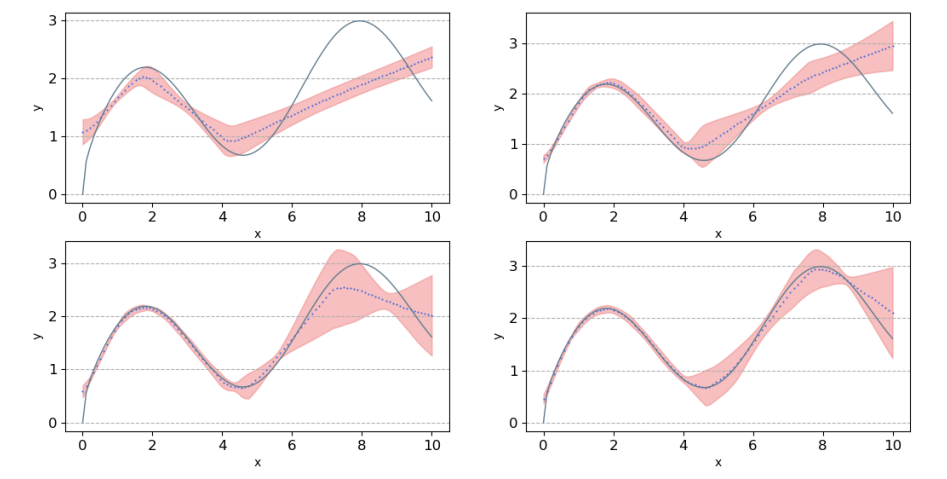


FIGURE 1.12 – Variabilité des prédictions sur 100 points de tests après 10 ré-entraînements indépendants de 50 epochs (**en haut à gauche**), 100 epochs (**en haut à droite**), 200 epochs (**en bas à gauche**), et 400 epochs (**en bas à droite**). La prédiction moyenne correspond aux pointillés bleus, alors que l'incertitude est représentée ici comme une plage rouge recouvrant deux fois l'écart-type. La référence est tracée en bleu foncé.

Ces deux dernières sources d'incertitude sont intriquées car elles induisent toutes deux une

variabilité dans les paramètres "possibles" du réseau de neurones. On parlera notamment par la suite de *paysage des paramètres*, dans lequel ces derniers évoluent lors de l'entraînement.

1.5.3 Incertitude de modèle

L'incertitude de modèle est une source courante dans la modélisation paramétrique probabiliste. Afin d'acquérir la capacité d'effectuer une tâche spécifique, définie par la constitution de la base de données et la fonction de coût, il est nécessaire de mettre en place une architecture de réseau composée de nombreux hyperparamètres tels que le nombre de couches, le nombre de paramètres, l'utilisation de couches de régularisation, et l'initialisation des paramètres.

Cette architecture définit une famille paramétrique, notée $f_{\mathcal{M}}$, dans laquelle nous cherchons à trouver l'ensemble optimal de paramètres pour accomplir la tâche souhaitée. Cependant, il existe une infinité d'architectures $\mathcal{M}$ qui pourraient fournir des résultats satisfaisants, c'est-à-dire atteindre une certaine performance sur un jeu de test. Néanmoins, une performance globale (moyenne) satisfaisante ne garantit pas une prise de décision locale identique.

Dans leur étude, Hüllermeier and Waegeman (2021) proposent de décomposer cette incertitude en deux sources distinctes. La première, appelée ici incertitude d'architecture, se réfère à l'écart minimal entre la véritable fonction f_{true} et le meilleur prédicteur possible pour $f_{\mathcal{M}}$, noté $\tilde{f}_{\mathcal{M}}$. La seconde, nommée ici incertitude d'approximation (en accord avec (Hüllermeier and Waegeman, 2021)), représente l'erreur ou la distance entre le meilleur prédicteur théorique $\tilde{f}_{\mathcal{M}}$ et le prédicteur obtenu après entraînement, noté $\hat{f}_{\mathcal{M}}$.

Cette variabilité des familles de prédicteurs est appelée incertitude de modèle. En notant $\mathcal{M}^i$, la i^e architecture considérée, on obtient plus formellement :

$$u^2_{model}(\mathbf{y}^*) = \mathbb{V}(\hat{f}_{\mathcal{M}^i}(\mathbf{x}^*, \hat{\theta}_{\mathcal{D}_N,opt})) \tag{1.38}$$

Remarque 7.

Il est "facile" de réduire l'incertitude d'architecture : il suffirait d'augmenter la complexité du réseau de neurones en augmentant son nombre de couches et son nombre de neurones. Cependant, une telle augmentation peut augmenter l'erreur d'approximation, dans le sens où il sera nécessaire de présenter plus de données afin de bien calibrer un modèle plus complexe.

Exemple illustratif : Incertitude de modèle

Voici les hyperparamètres spécifiques pour expliciter l'incertitude prédictive induite par l'incertitude de modèle :

Hyperparamètres	Valeur
Nombre de données	10000
Nombre de couches	Variable
Nombre de neurones	Variable
Fonction d'activation	Variable
Nombre d'époch s	200
Nombre de données de test	100

La figure 1.13 présente l'incertitude prédictive induite par l'incertitude de modèle. Ainsi, on fait varier le nombre de couches L et de neurones d au sein de notre architecture ($L \in [2, 3, 4, 5]$ et $d \in [16, 32, 64]$), nous offrant en tout 12 estimateurs desquels on extrait une moyenne et un écart-type sur 100 points de test.

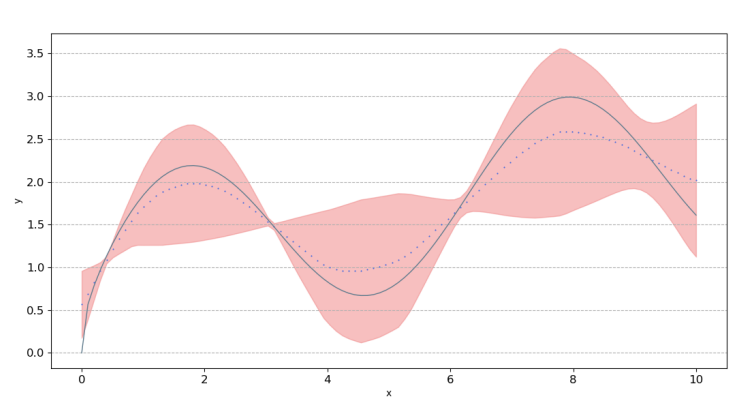


FIGURE 1.13 – Variabilité des prédictions sur 100 points de tests et 12 configurations de réseau. La prédiction moyenne correspond aux pointillés bleus, alors que l'incertitude est représentée ici comme une plage rouge recouvrant deux fois l'écart-type. La référence est tracée en bleu foncé.

Une nouvelle fois, l'incertitude de modèle n'est pas inclue lors de la quantification de l'incertitude sous l'angle aléatoire/épistémique.

1.5.4 Incertitude de la base de données

La dernière catégorie d'incertitudes est composée des incertitudes des éléments constitutifs de la base de données d'entraînement. En apprentissage statistique, la constitution de la base de données

d'apprentissage reste primordiale. Plusieurs problématiques peuvent émerger :

— Y a-t-il assez de données dans la base ? Cette question a une réponse à travers la quantification de l'incertitude d'échantillonnage (cf. section 1.5.2).

— Quelles sont les limites de fonctionnement établies par les données ? Cette question a une réponse à travers la quantification de l'incertitude de distribution (cf. section 1.5.5).

— Les données brutes d'entrée sont-elles bruitées ? La base de données d'apprentissage est constituée d'éléments $\mathbf{x}_i$, estimations (ou versions bruitées) de la "vraie" valeur $\mathbf{x}_{true}$.

— Les données brutes de sortie sont-elles bruitées ? Dans un cas de classification, par exemple, les classes sont statuées par des opérateurs humains et donc assujetties à des erreurs d'annotations.

Nous désirons traiter ici les concepts mis en lumière par ces deux derniers points. L'incertitude des données brutes provient la plupart du temps de l'incertitude issue de l'acquisition des données, qui sont souvent issues de capteurs physiques. Avec les notations précédentes, la base de données est constituée de données $\mathbf{x}_i$, réalisation de la variable aléatoire $\mathbf{X}_i$, approximation de la vraie valeur $\mathbf{x}_{true}$. Enfin, l'incertitude d'annotation consiste également à un bruit présent dans l'espace des sorties. Ce bruit peut également provenir d'une acquisition de données (par exemple, un temps chronométré) dans un cas de régression, ou de mauvaise annotation pour un cas de classification (une image de chat annotée comme chien par exemple). Avant d'illustrer ces sources d'incertitude au sein de notre exemple de régression, nous désirons fournir ici un cas de classification. Pour cela, nous générons un cas de classification binaire consistant en deux cercles de centre $(0, 0)$ et $(0, 2)$. Les entrées $\mathbf{x}$ sont corrompues par un bruit gaussien de moyenne $\mu = 0.0$ et d'écart-type σ variable. Nous mettons en place une classification par SVM (Cortes and Vapnik, 1995). La figure 1.14 présente cette classification pour un bruit $\sigma = 0.1$ ainsi que l'évolution de la largeur des marges en fonction de l'intensité du bruit d'entrée.

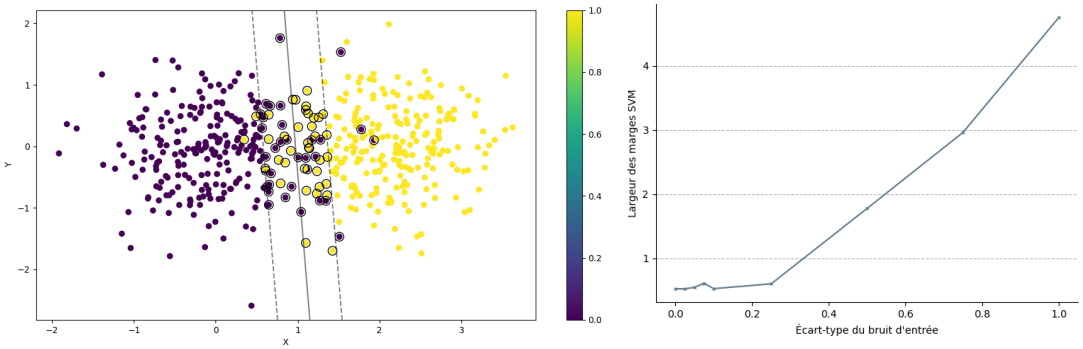


FIGURE 1.14 – **Gauche :** Classification SVM de 2 cercles dont les entrées sont corrompues par un bruit gaussien d'écart-type $\sigma = 0.1$. La frontière de décision est tracée en ligne pleine, les marges sont tracées en pointillés. **Droite :** Évolution de la largeur des marges du classifieur en fonction de σ.

La largeur des marges d'un classifieur SVM traduit la largeur de la zone où la "confiance" en les prédictions est faible. On observe aisément que la largeur de la zone de doute à la prédiction

augmente lorsque le bruit sur les entrées d'entraînements augmente. L'expérience consistant, cette fois ci, à corrompre les valeurs des classes des échantillons à la frontière des deux cercles conduit à des résultats similaires.

On présente ci-après l'application de ces concepts au sein de notre exemple illustratif.

Exemple illustratif : Incertitude des données

Voici les hyperparamètres spécifiques pour expliciter l'incertitude prédictive induite par l'incertitude des données constituant la base de données d'apprentissage :

Hyperparamètres	Valeur
Nombre de données	5000
Nombre de couches	5
Nombre de neurones	64
Nombre d'épochs	200
Nombre de données de test	100

La figure 1.15 présente l'incertitude prédictive induite par l'incertitude des données d'entraînement. Cette incertitude peut se manifester dans l'espace des entrées (par exemple, des images de faible résolution) ou dans l'espace des sorties (lors de mauvaises mesures ou annotations). La figure de gauche montre la variabilité des prédicteurs lors de 10 ré-entraînements lorsque les données d'entrées sont corrompues avec un bruit gaussien d'écart-type $\sigma = 0.2$. La figure de droite présente la variabilité des prédicteurs lors de 10 ré-entraînements lorsque les données de sorties sont corrompues par un bruit gaussien d'écart-type $\sigma = 1.0$.

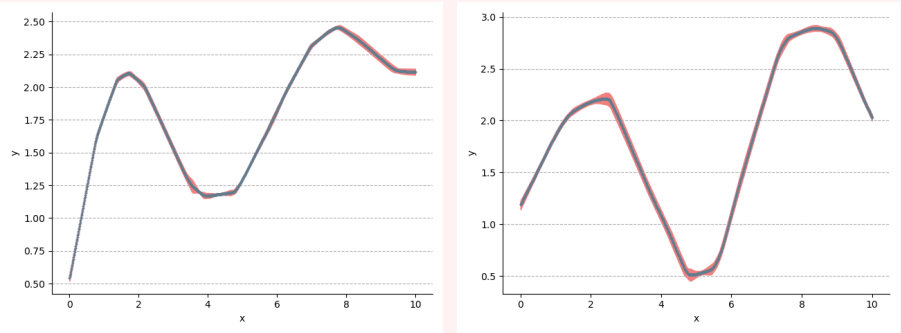


FIGURE 1.15 – Variabilité des prédictions sur 100 points de tests après 10 ré-entraînements indépendants avec corruption gaussienne des entrées $\mathbf{x}_i$ (**à gauche**), et des sorties $\mathbf{y}_i$ (**à droite**). La prédiction moyenne correspond aux pointillés bleus, alors que l'incertitude est représentée ici comme une plage rouge recouvrant deux fois l'écart-type.

Plusieurs observations sont possibles :

— la variabilité des prédictions est faible sur l'ensemble du domaine d'entrée, cela est en partie expliquée par la fonction de coût consistant à minimiser l'erreur quadratique moyenne,
— la calibration est fortement affectée,
— la corruption des entrées et des sorties rendent des résultats similaires.

Nous tenons à mettre en avant le très récent travail effectué au sein de Gruber et al. (2023) qui propose une étude poussée des concepts autour de l'incertitude issue de la base d'entraînement. Nous recommandons aux lecteurs de consulter cette référence pour plus d'informations. En effet, ces notions n'ont pas fait l'objet d'une recherche avancée lors de ces travaux de thèse.

1.5.5 Incertitude de distribution

La quantification de l'incertitude peut être axée sur la détection des échantillons hors distributions. En règle générale, la cohérence d'une méthode est établie lorsque l'incertitude globale est plus prononcée pour les échantillons hors distributions que pour ceux se situant dans la distribution spécifiée par la base d'entraînement.

Exemple illustratif : Incertitude de distribution

Voici les hyperparamètres spécifiques pour expliciter l'incertitude prédictive induite par l'incertitude de distribution :

Hyperparamètres	Valeur
Nombre de données	10000
Nombre de données de test	150

La figure 1.16 présente l'incertitude prédictive induite par l'incertitude de distribution. Le réseau de neurone est entraîné sur une gamme comprise entre 0.0 et 10.0. L'inférence se fait sur 150 points entre -2.0 et 12.0.

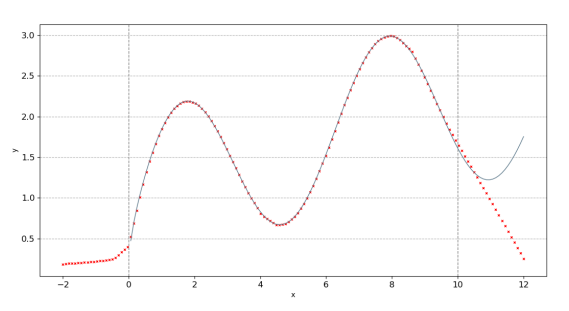


FIGURE 1.16 – Prédictions (**croix rouge**) sur 150 points de test. Les lignes verticales indiquent les limites de la plage de fonctionnement. La référence est tracée en bleu foncé.

On observe que la calibration du modèle est parfaite sur le domaine spécifié par la base de données d'entraînement (pour des entrées situées entre 0.0 et 10.0). Les performances chutent drastiquement en dehors de cette plage de fonctionnement.

En revanche, d'un point de vue métrologique, la notion de *hors distribution* peut ne pas être pertinente. Le métrologue préférera développer une structure de contrôle pour déterminer quel échantillon peut être mesuré avec l'algorithme et lequel ne peut pas l'être.

Néanmoins, les tests hors distribution sont de bons outils afin d'obtenir une intuition de la performance de la méthode de quantification de l'incertitude, mais ne doivent pas être considérés comme la preuve d'une bonne quantification.

1.5.6 Une vue d'ensemble

La figure 1.17 présente une vue d'ensemble des sources d'incertitude lors de la prédiction d'un réseau de neurones où l'apprentissage fut supervisé.

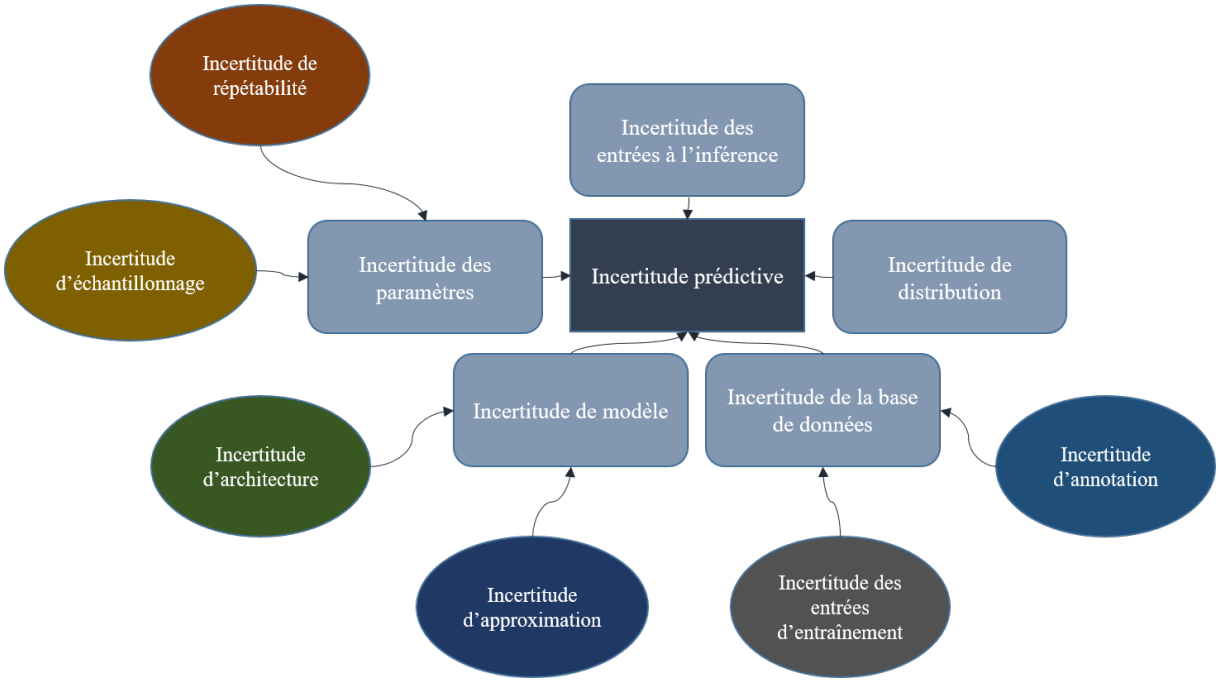


FIGURE 1.17 – Vue d'ensemble des sources d'incertitude dans un réseau de neurones d'apprentissage supervisé

Nous désirons, maintenant, classer l'ensemble des méthodes de l'état de l'art précédemment présentées au sein de notre cadre sémantique proposée en section 1.4. La table 1.3 présente cette classification.

TABLE 1.3 – Classification des méthodes de l'état de l'art en accord avec les sources d'incertitude quantifiées. Une croix valide verte indique que la méthode est alignée sur notre définition spécifiée précédemment, une croix rouge pour indiquer que la méthode ne correspond pas à notre définition, un point d'interrogation pour indiquer qu'il n'est pas clair si la méthode permet de capturer l'incertitude de la source correspondante.

Méthodes / Sources	Entrée	Données	Répétabilité	Échantillonnage	Modèle	Distribution
LPN (Gast and Roth, 2018)	✓	✗	✗	✗	✗	✗
Unscented transform (Julier and Uhlmann, 2004)	✓	✗	✗	✗	✗	✗
Zhang and Shin (2021)	✓	✗	✗	✗	✗	✗
WGMprop (Monchot et al., 2023)	✓	✗	✗	✗	✗	✗
BBB (Blundell et al., 2015)	✗	✗	✗	✓	✗	✗
PBP (Hernández-Lobato and Adams, 2015)	✗	✗	✗	✓	✗	✗
Laplace (Hernández-Lobato and Adams, 2015)	✗	✗	✗	✓	✗	✗
MC Dropout (Gal and Ghahramani, 2016)	✗	✗	✗	✓	?	✗
Deep ensembles (Lakshminarayanan et al., 2017)	✗	✗	✓	?	✗	✗
Bootstrap (Breiman, 1996)	✗	✗	✓	?	✗	✗
Multi-SWAG (Maddox et al., 2019)	✗	✗	✓	?	✗	✗
SGLD (Welling and Teh, 2011)	✗	✗	✗	✓	✗	✗
pSGLD (Li et al., 2016)	✗	✗	✗	✓	✗	✗
SGHMC (Chen et al., 2014)	✗	✗	✗	✓	✗	✗
EiV (Martin and Elster, 2022)	✗	✓	✗	✗	✗	✗
Y. Gal (Kendall and Gal, 2017)	✗	✓	✗	✗	✗	✗
Prior network (Malinin and Gales, 2018)	✗	✓	✗	✗	✗	✓
DUQ (Van Amersfoort et al., 2020)	✗	?	✗	?	✗	✓
OCs (Tagasovska and Lopez-Paz, 2019)	✗	✓	✗	✓	✗	✗

Cette classification ne juge pas l'efficacité de ces méthodes, mais seulement leur "philosophie" et ce qu'elles tentent de capturer. On note notamment qu'il existe un doute quant à la pertinence des méthodes des ensembles profonds et (Multi-)SWAG quant à la quantification de l'incertitude d'échantillonnage. Cette problématique est encore une question débattue dans la littérature. L'annexe A présente une réflexion sur ces notions.

Nous venons d'établir les concepts de base de la quantification de l'incertitude des réseaux de neurones. Nous avons notamment vu une approche "métrologique" de cette problématique et proposé une sémantique plus proche des exigences des problématiques de quantification d'incertitudes. Nous avons également mis en lumière les deux catégories d'incertitudes considérées dans la communauté de l'apprentissage profond ainsi que les grandes méthodes à l'état de l'art. Dans la suite de ce rapport, nous présentons nos travaux effectués sur ces problématiques. Il est important de noter que dans un cadre métrologique, notre recherche de quantification d'incertitudes ne peut reposer sur des solutions elles-mêmes basées sur des réseaux de neurones, à l'image des réseaux enseignants (Gou et al., 2021). De plus, nos solutions doivent être aussi génériques que possible afin de pouvoir les utiliser sur une majorité des cas rencontrés dans un contexte industriel.

Tout d'abord, nous aborderons la problématique de la propagation des incertitudes des entrées d'un réseau de neurones au temps d'inférence. Ce champ de l'état de l'art manque de méthodes performantes et génériques ne reposant pas sur des hypothèses fortes de forme. Nous proposerons donc à cette occasion, une nouvelle méthodologie, appelée WGMprop (Monchot et al., 2023), permettant de propager une distribution gaussienne au sein d'un réseau de neurones sous garanties théoriques.

Par la suite, nous présenterons nos recherches concernant l'incertitude des paramètres d'un réseau de neurones, mêlant incertitude d'échantillonnage et incertitude de répétabilité. Nous présenterons alors une étude comparant les principales méthodes de l'état de l'art avant de présenter nos développements s'intéressant aux notions de diversité d'un ensemble de prédicteurs, et de bassins d'attraction lors de l'apprentissage. Nous détaillerons, à cette occasion, une première version de notre procédure d'exploration du paysage des paramètres d'un réseau de neurones, nommée WEUQ, tentant d'approcher localement les ensembles profonds.

Enfin, nous présenterons notre cas d'application industriel, se focalisant sur l'estimation automatisée de la distribution de tailles de particules de TiO_2 au sein d'images acquises par microscopie électronique à balayage. Nous détaillerons les solutions technologiques mises en place, que ce soit pour le pipeline de traitement, que pour la quantification de l'incertitude, ainsi que les résultats numériques associés. Nous présenterons également rapidement la plateforme web créée à cette occasion.

Chapitre 2

Propagation de l'incertitude des entrées

Lorsque des données proviennent de capteurs physiques, tels que des capteurs d'images, la propagation de l'incertitude des données d'entrée est souvent une composante majeure de l'incertitude de sortie des modèles d'apprentissage profond. Dans ce chapitre, nous abordons le problème de la propagation de l'incertitude des données d'entrée au sein de réseaux de neurones pré-entraînés. Pour ce faire, nous ne désirons pas nous appuyer sur l'hypothèse gaussienne des distributions de chaque entrée / sortie des couches intermédiaires du réseau considéré. Nous désirons propager à la place un modèle de mélange gaussien (GMM) qui offre beaucoup plus de flexibilité. La principale contribution de ce chapitre à l'état de l'art est l'introduction d'un critère de Wasserstein pour contrôler la procédure de division gaussienne pour laquelle des garanties théoriques de convergence sont dérivées. La méthodologie est testée sur un éventail de bases de données et de réseaux sur lesquels elle fait preuve de robustesse et de généricité et offre une estimation précise de la fonction de densité de probabilité de sortie tout en maintenant un coût de calcul raisonnable par rapport à l'approche Monte Carlo (MC) standard.

Pour cela, nous proposons de nous appuyer sur la longue histoire du paradigme Split&Merge (Diviser et Fusionner) pour propager une distribution à travers des modèles non linéaires. Ce paradigme a été proposé à l'origine par Sorenson and Alspach (1971), qui utilisent des filtres de Kalman pour des problèmes de filtrage linéaire, avec une extension aux problèmes non linéaires dans (Alspach and Sorenson, 1972). Cette méthode a, par la suite, été reprise et améliorée au fil du temps. On peut noter en particulier l'intégration de l'échantillonnage par transformation non centrée (UT) (Julier and Uhlmann, 2004) par Faubel et al. (2009), une mise à jour adaptative des poids dans (Terefjanu, 2011), et enfin l'intégration d'un critère adapté aux systèmes dynamiques non linéaires dans (DeMars et al., 2013). Ce paradigme sera détaillé dans la section 2.1.2. La méthodologie proposée utilise alors un mélange de distributions gaussiennes propagé dans le réseau de neurones via une procédure de division contrôlée par un critère s'appuyant sur la distance de Wasserstein (voir, par exemple, Villani, 2009, Sect. 7.1). L'introduction de la distance de Wasserstein comme critère de

division offre ainsi un critère robuste et adapté aux architectures de réseaux de neurones. Il permet de détecter l'impact de la non-linéarité des réseaux de neurones introduit par les différentes fonctions d'activation sur les différentes composantes gaussiennes de notre mixture. Ainsi, cela permet de contrôler le nombre de composantes gaussiennes dans le mélange, établissant un compromis entre précision des estimations, complexité mémoire et temps de calcul. Des garanties théoriques de convergence sur les estimations de la distribution de sortie sont proposées dans la section 2.1. Enfin, des résultats expérimentaux détaillés démontrent que la méthode proposée se compare favorablement aux approches de l'état de l'art sur les ensembles de données MNIST (LeCun, 1998), CIFAR10 (Krizhevsky et al., 2009), CIFAR100 (Krizhevsky et al., 2009), et Camelyon (Litjens et al., 2018).

2.1 Propagation de l'incertitude des entrées utilisant des modèles de mélange gaussien (WGMprop)

La première tentative de propagation d'une distribution de mélange gaussien (MG) dans un réseau de neurones $f\colon \mathbb{R}^d \to \mathbb{R}^{d'}$ à l'aide du paradigme Split&Merge (voir la section 2.1.2 pour plus de détails) a été réalisée par Zhang and Shin (2021). Pour une variable aléatoire gaussienne $X \in \mathbb{R}^d$, les auteurs mesurent la similarité entre $f(X)$ et une approximation de Taylor du premier ordre $\overline{f}(X)$ en utilisant la divergence de Kullback-Leibler de $f(X)$ par rapport à $\overline{f}(X)$. La section 2.1.5 détaille comment la distribution de X est *divisée* en un mélange de *petites* gaussiennes lorsque la valeur de cette divergence dépasse un certain seuil. La divergence de Kullback-Leibler est exprimée sous la forme d'une intégrale impliquant les densités des distributions de $f(X)$ et $\overline{f}(X)$ par rapport à une certaine mesure dominante. Malheureusement, l'expression de la distribution de $f(X)$ avec une densité est en général irréalisable pour un réseau de neurones f. Pour contourner ce problème, Zhang and Shin (2021) utilisent récursivement le paradigme Split&Merge pour propager la distribution à travers les couches linéaires et les fonctions d'activation. La propagation de (mélanges de) distributions gaussiennes à travers les couches linéaires est simple et se réduit à la mise à jour des deux premiers moments de la distribution gaussienne. En revanche, pour une fonction d'activation lisse et inversible $\sigma\colon \mathbb{R}^q \to \mathbb{R}^q$ (agissant typiquement composante par composante), Zhang and Shin (2021) proposent de dériver une borne supérieure de la divergence de Kullback-Leibler à l'aide la formule suivante $p(\sigma(A)) = |J_\sigma(A)^{-1} p(A)|$, où $J_\sigma(A)$ désigne la jacobienne de σ. Cependant, l'unité Linéaire Rectifiée (ReLU, Maas et al., 2013) n'est pas inversible et, plus fondamentalement, la divergence de Kullback-Leibler d'une distribution gaussienne rectifiée à partir d'une variable aléatoire continue est toujours infinie. Les auteurs proposent alors d'approcher la fonction d'activation ReLU par d'autres fonctions d'activation qui leur sont très proches : les Leaky ReLU avec une pente $\delta \ll 1$. Cependant, la section 2.2 montrera expérimentalement que cette approche est impraticable en présence d'un bruit significativement affecté par la non-linéarité du réseau. Ceci est dû au fait qu'une fonction d'activation telle que la Leaky ReLU présentant une pente $\delta \ll 1$ concentre encore une fraction

potentiellement importante de la masse de probabilité dans un intervalle de la forme $[-O(\delta), 0]$. Dans ce cas, la divergence de Kullback-Leibler entre une gaussienne dont la variance est supérieure à $O(\delta)$ et son image par Leaky ReLU devient très importante. Par conséquent, le paradigme récursif de Split&Merge conduit à un nombre de divisions très élevé dès que la première fonction d'activation (Leaky) ReLU est rencontrée.

Le critère de Wasserstein proposé dans la section 2.1.4 permet de surmonter ces limites en : 1) ne nécessitant pas l'utilisation de densités ; 2) étant moins sensible aux événements de faible probabilité ; 3) en permettant d'utiliser le paradigme Split&Merge sur l'ensemble du réseau (suivant une approche boite noire) ; et 4) assurer des garanties théoriques de convergence présentées dans la section 2.1.4.

2.1.1 Notations.

Soit $\boldsymbol{U} \in \mathbb{R}^d$ la variable aléatoire multidimensionnelle de distribution de probabilité $P_{\boldsymbol{U}}$ et de moyenne $\mu_{\boldsymbol{U}}$ et de matrice de covariance $\Sigma_{\boldsymbol{U}}$.

Soit $\mathbb{M}$ l'espace des mélanges de distributions gaussiennes et, pour $\boldsymbol{U}$ tel que $P_{\boldsymbol{U}} \in \mathbb{M}$, on écrit $P_{\boldsymbol{U}} = \sum_{i=1}^{M} w_i P_{\boldsymbol{U},i}$, avec w_i non négatifs dont la somme est égale à un et $P_{\boldsymbol{U},i} = \mathcal{N}(\mu_{\boldsymbol{U},i}, \Sigma_{\boldsymbol{U},i})$. Les symboles M et w_i resteront génériques pour tout mélange, sauf si indiqué explicitement.

En outre, supposons que $S^{[1]}(\cdot; n)$ soit un opérateur de division défini sur l'espace des distributions gaussiennes et étendu aux mélanges gaussiens par linéarité. $S^{[1]}(\cdot; n)$ approche $P_{\boldsymbol{U}}$ par une distribution de mélange gaussien de n composantes :

$$P_{\hat{\boldsymbol{U}}^{[1]}} = S^{[1]}(P_{\boldsymbol{U}}; n) = \sum_{i=1}^{n} w_i P_{\hat{\boldsymbol{U}}^{[1]},i}$$

Étant donné un certain $\boldsymbol{X} \in \mathbb{R}^d$, l'objectif est de propager $P_{\boldsymbol{X}}$ à travers un réseau de neurones $L_{2,2}$-Lipschitzien : $\boldsymbol{f} \colon \mathbb{R}^d \to \mathbb{R}^{d'}$. (Plus précisément, toutes les fonctions d'activation de $\boldsymbol{f}$ sont supposées être lipschitzienne). En tirant parti du pouvoir d'approximation des mélanges de distributions gaussiennes (voir, par exemple, Scott, 2015), nous supposons désormais que $\boldsymbol{X} \in \mathbb{R}^d$ est une variable aléatoire normale multivariée.

Remarque 9.

Ce choix est fréquemment adopté en pratique, surtout lorsqu'il s'agit de distributions d'entrée dont la forme demeure généralement difficilement accessible. Cela s'avère particulièrement prévalent dans le cas d'entrées de haute dimension, telles que les images.

La distribution de probabilité $P_{\boldsymbol{X}}$ sera souvent un mélange de distributions gaussiennes $\sum_{i=1}^{M} w_i P_{\boldsymbol{X},i} \in$

$\mathbb{M}$, avec w_i des poids positifs dont la somme est égale à un et $P_{\boldsymbol{X},i} = \mathcal{N}(\mu_{\boldsymbol{X},i}, \Sigma_{\boldsymbol{X},i})$. En utilisant les notations précédentes, sans tenir compte de la dépendance de M par rapport à n et à s—, itérer s fois l'opérateur de split S sur $P_{\boldsymbol{X}}$ nous conduit à :

$$P_{\hat{\boldsymbol{X}}^{[s]}} = S^{[s]}(P_{\boldsymbol{X}}; n) = \sum_{i=1}^{M} w_i P_{\hat{\boldsymbol{X}}^{[s]},i}$$

Ceci jouera un rôle majeur dans ce travail.

Enfin, notons $\boldsymbol{Y}$ et $\hat{\boldsymbol{Y}}^{[s]}$ les variables aléatoires telles que $\boldsymbol{Y} = \boldsymbol{f}(\boldsymbol{X})$, $\hat{\boldsymbol{Y}}^{[s]} = \boldsymbol{f}(\hat{\boldsymbol{X}}^{[s]})$ et notons $\tilde{\boldsymbol{Y}}^{[s]}$ la variable aléatoire obtenue en propageant chaque composante de $P_{\hat{\boldsymbol{X}}^{[s]}}$ à travers $\boldsymbol{f}$ par appariement des moments. L'appariement des moments consiste à estimer les deux premiers moments d'une distribution sous hypothèse de forme (ici hypothèse de gaussianité).

2.1.2 Paradigme Split&Merge pour les modèles de mélange gaussien.

L'algorithme Split&Merge est un processus itératif construit en 4 étapes. À l'itération $s \in \mathbb{N}$, avec $P_{\hat{\boldsymbol{X}}^{[0]}} = P_{\boldsymbol{X}}$, nous appliquons les étapes suivantes pour chaque composante $P_{\hat{\boldsymbol{X}}^{[s]},i}$ de $P_{\hat{\boldsymbol{X}}^{[s]}}$:

— step1/**propagation** : propage, sous hypothèse linéaire, la composante courante à travers $\boldsymbol{f}$. Étant donné que les distributions normales sont stables par opération linéaire, l'étape de propagation se réduit à l'estimation des deux premiers moments de sortie d'une telle gaussienne, à partir de laquelle nous obtenons l'estimation $P_{\tilde{\boldsymbol{Y}}^{[s]},i}$ obtenue par appariement des moments de la véritable distribution de sortie $P_{\hat{\boldsymbol{Y}}^{[s]},i}$. Les moments peuvent être estimés, par des techniques d'échantillonnage, ou par des formules analytiques lorsqu'elles sont disponibles.

— step2/**détection de non-linéarité** : quantifie l'*intensité* avec laquelle la distribution gaussienne $P_{\hat{\boldsymbol{X}}^{[s]},i}$ a été affectée par la non-linéarité au cours de la propagation en utilisant une mesure de divergence qui sera comparée à un seuil noté T_{split}.

— step3/**division** : divise la distribution gaussienne d'entrée $P_{\hat{\boldsymbol{X}}^{[s]},i}$ en utilisant un opérateur de division $S(\cdot, n)$ dans le cas où l'hypothèse de linéarité locale est invalidée à l'étape 2 (la mesure de divergence a été supérieure au seuil T_{split}), puis recommence à l'étape 1 sur les "sous"-composantes divisées. Dans le cas contraire, $P_{\hat{\boldsymbol{X}}^{[s]},i}$ reste inchangé. La division d'une distribution gaussienne multivariée $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ consiste à diviser la distribution dans une direction unique spécifique, selon la formule $S^{[1]}(\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma}); n) = \sum_{i=1}^{n} w_i \mathcal{N}(\boldsymbol{\mu_i}, \boldsymbol{\Sigma_i})$; où w_i, $\boldsymbol{\mu_i}$ et $\boldsymbol{\Sigma_i}$ sont calculés à l'aide des formules présentées en section 2.1.5. L'opérateur de division introduit une approximation à chaque itération, liée à l'erreur type $\epsilon_{0,1} = W_2(\mathcal{N}(0,1), S^{[1]}(\mathcal{N}(0,1); n)) > 0$. Cependant, nous pouvons trouver un mélange gaussien rendant $\epsilon_{0,1}$ arbitrairement petit, tant qu'un nombre suffisant de composantes, de variances réduites, est utilisé dans le mélange. La procédure de division s'arrête lorsque le critère calculé à l'étape 2 devient inférieur au seuil

T_{split}.

— step4/**fusion** : fusionne les composantes similaires à l'aide d'une mesure de divergence (une nouvelle fois comparée à un second seuil T_{merge}) afin de réduire le nombre de composantes dans le mélange de sortie $P_{\tilde{\boldsymbol{Y}}^{[s]}}$. Cette étape n'est pas appliquée dans notre travail. Nous proposons néanmoins un critère de fusion (en annexe D), une nouvelle fois s'appuyant sur la distance de 2-Wasserstein.

En itérant suffisamment (s fois) sur l'étape de division, on obtient l'approximation $P_{\hat{\boldsymbol{X}}^{[s]}} = S^{[s]}(P_{\boldsymbol{X}}; n) = \sum_{i=1}^{M} w_i P_{\hat{\boldsymbol{X}}^{[s]},i} = \sum_{i=1}^{M} w_i \mathcal{N}(\mu_{\hat{\boldsymbol{X}}^{[s]},i}, \Sigma_{\hat{\boldsymbol{X}}^{[s]},i})$ de $P_{\boldsymbol{X}}$ où chaque composante vérifie l'hypothèse de linéarité locale pendant l'étape de propagation, et permet donc de calculer facilement une estimation de la distribution $P_{\tilde{\boldsymbol{Y}}^{[s]}} = \sum_{i=1}^{M} w_i P_{\tilde{\boldsymbol{Y}}^{[s]},i}$ obtenue en propageant chaque composante de $P_{\hat{\boldsymbol{X}}^{[s]}}$ par appariement des moments. La figure 2.1 présente un schéma blocs d'un processus de Split&Merge.

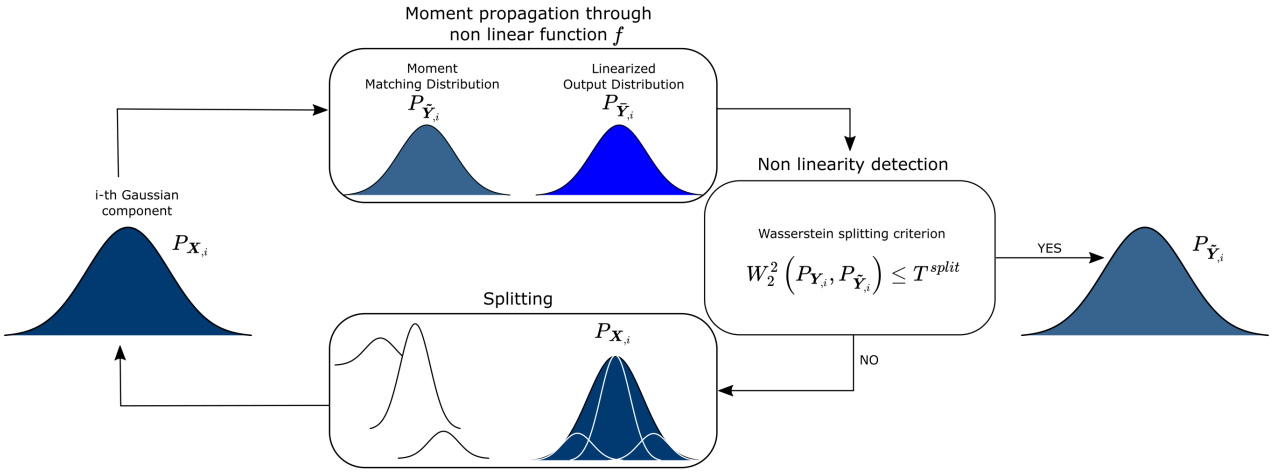


FIGURE 2.1 – Schéma blocs d'un processus itératif de Split&Merge.

La question essentielle lors de l'utilisation de cet algorithme itératif est le choix du critère agissant comme détecteur de non-linéarité.

2.1.3 Illustration 1D

Nous proposons, ici, une illustration unidimensionnelle de la propagation d'une distribution gaussienne $\mathcal{N}(\mu, \sigma = 1.0)$ à travers une fonction d'activation ReLU.

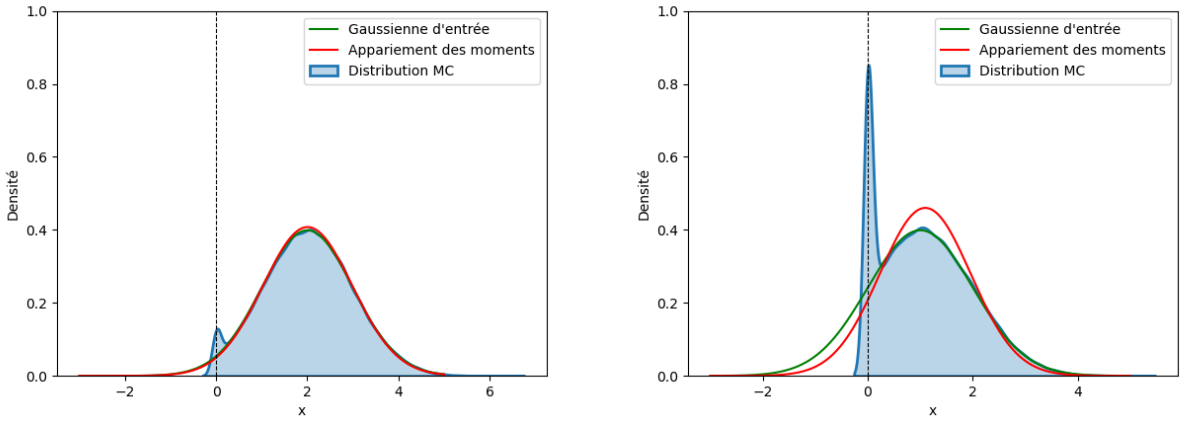


FIGURE 2.2 – Propagation d'une distribution gaussienne unidimensionnelle (en vert) $\mathcal{N}(\mu, \sigma = 1.0)$ de moyenne $\mu = 2.0$ (**Gauche**), $\mu = 1.0$ (**Froite**) à travers une fonction d'activation ReLU. La distribution de sortie de Monte-Carlo est affichée en bleu, alors que la distribution gaussienne resultante par appariement des moments est présentée en rouge.

La figure 2.2 présente la propagation d'une distribution gaussienne unidimensionelle $\mathcal{N}(\mu, \sigma = 1.0)$ de moyenne variable à travers une fonction d'activation ReLU. Cette propagation se fait par appariement des moments, que l'on compare à une référence de Monte-Carlo, présentée en bleu au sein de la figure. On observe que plus la moyenne de la distribution gaussienne d'entrée est proche de la non linéarité située en $\mathbf{x} = 0.0$ et plus la distribution gaussienne obtenue par appariement des moments est "distante" de la référence. Lors de l'utilisation d'un algorithme Split&Merge, le critère de division permet de venir quantifier de quelle façon la distribution d'entrée est impactée par la non-linéarité de la fonction $\boldsymbol{f}$.

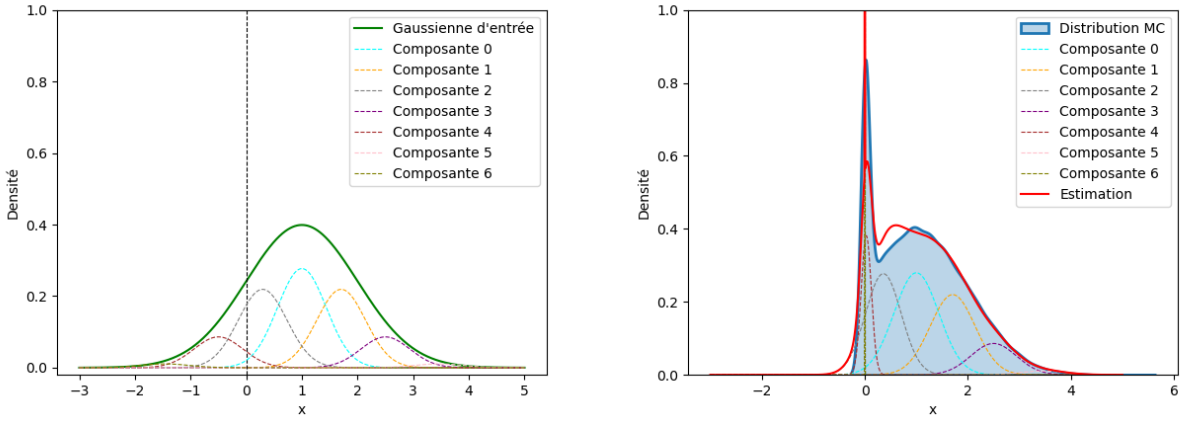


FIGURE 2.3 – Propagation d'une distribution gaussienne unidimensionnelle (en vert) $\mathcal{N}(\mu = 1.0, \sigma = 1.0)$ divisée en 7 composantes (**Gauche**) et estimation de la distribution résultante par propagation par appariement des moments de chacune des composantes d'entrée (**Droite**).

La figure 2.3 présente la propagation d'une distribution gaussienne unidimensionelle $\mathcal{N}(\mu = 1.0, \sigma = 1.0)$ à travers une fonction d'activation ReLU lorsque cette dernière est divisée en 7 "petites" composantes. Chacune de ces composantes sont propagées par appariement des moments

et permettent de reconstituer l'estimation après propagation. L'estimation résultante (en rouge) est beaucoup plus proche de la référence de Monte-Carlo que de l'estimation proposée au sein de la figure 2.3 qui n'utilise pas de phase de division. Ainsi, diviser la gaussienne d'entrée en plus "petits morceaux" permet une décomposition de cette dernière en une mixture de composantes qui individuellement sont faiblement impactées par la non linéarité. Comme mentionné précédemment, le choix du critère permettant de quantifier l'impacte de la non-linéarité sur la distribution d'entrée est primordiale lors de l'utilisation de l'algorithme Split&Merge. Dans la section suivante, nous présentons un critère basé sur la distance de Wasserstein.

2.1.4 La métrique de Wasserstein comme critère de division.

La métrique de Wasserstein

La métrique de Wasserstein (voir, par exemple, Villani, 2009, p.118 et ses références) est un outil populaire utilisé notamment dans la théorie du transport. Étant donné un entier strictement positif p, la distance de Wasserstein entre les mesures P et Q est définie comme suit

$$W_p\left(P,\, Q\right) = \left(\inf_{\gamma \in \Gamma(P,\, Q)} \mathbb{E}_{(X,Y)\sim\gamma}\left(\|X - Y\|^p\right)\right)^{1/p}, \tag{2.1}$$

où Γ est l'ensemble des mesures de probabilité sur $\mathbb{R}^d \times \mathbb{R}^d$ ayant pour marginales P et Q.

Les distances de Wasserstein présentent plusieurs propriétés intéressantes. Tout d'abord, ce sont des distances et vérifient, en particulier, l'inégalité triangulaire. Cela sera utile pour calculer les distances entre les mélanges de distributions gaussiennes. Deuxièmement, elles ont également des propriétés théoriques intéressantes. En effet, il est connu (voir, par exemple, Villani, 2009, Proposition 7.29) que la distance de p-Wasserstein contrôle les moments jusqu'au p-ième ordre. Par conséquent, lorsque $p \geq 2$, $W_p(\cdot, \cdot)$ peut être utilisé pour contrôler l'erreur sur les deux premiers moments, qui sont parfois d'un intérêt primordial.

La proposition suivante est une reformulation mineure de ces résultats (connus).

Proposition 2.1.1. *Soit $W_p(\cdot, \cdot)$ la distance de p-Wasserstein par rapport à la norme euclidienne habituelle sur $\mathbb{R}^d$, alors : En utilisant les notations précédentes, on a :*

$$\|\mu_{\boldsymbol{Y}} - \mu_{\tilde{\boldsymbol{Y}}^{[s]}}\| \leq W_p(P_{\boldsymbol{Y}}, P_{\tilde{\boldsymbol{Y}}^{[s]}}).$$

Par ailleurs, notons $\sigma_{\boldsymbol{Y},i}$ et $\sigma_{\tilde{\boldsymbol{Y}}^{[s]},i}$ les écarts-types des i-èmes marginales de $P_{\boldsymbol{Y}}$ et $P_{\tilde{\boldsymbol{Y}}^{[s]}}$. Si $p \geq 2$,

alors

$$\left|\sigma_{\boldsymbol{Y},i} - \sigma_{\tilde{\boldsymbol{Y}}^{[s]},i}\right| \leq W_p(P_{\boldsymbol{Y}}, P_{\tilde{\boldsymbol{Y}}^{[s]}})$$

Preuve de la proposition 2.1.1. Soit P_0 et $\tilde{P}_0$ les versions centrées de P et $\tilde{P}$. Utilisant l'identité $\mathbb{E}\left[\|A\|^2\right] = \|\mu_A\|^2 + \mathrm{Tr}\left(\mathrm{Cov}\left(A\right)\right)$ pour un vecteur aléatoire A, on obtient

$$\begin{aligned} W_2^2(P,\, \tilde{P}) &= \inf_{\gamma\in\Gamma} \mathbb{E}_{X,Y\sim\gamma}\left[\|X-Y\|^2\right] \\ &= \|\mu_P - \mu_{\tilde{P}}\|^2 + \inf_{\gamma\in\Gamma\left(P,\tilde{P}\right)} \mathrm{Tr}\left(\mathrm{Cov}\left(X-Y\right)\right) \\ &= \|\mu_P - \mu_{\tilde{P}}\|^2 + W_2^2(P_0,\, \tilde{P}_0). \end{aligned}$$

La dernière équation nous fournit directement la première propriété à démontrer. De plus, soit $X_0 \sim P_0$ et $Y_0 \sim \tilde{P}_0$. Avec $\phi_i\colon\ (x_1,\ldots,x_n) \in \mathbb{R}^d \to x_i$ et (Villani, 2009, Proposition 7.29), on a :

$$W_2(P,\, \tilde{P}) \geq W_2(P_0,\, \tilde{P}_0) \geq \left|\sqrt{\int x_i^2 dP_0} - \sqrt{\int x_i^2 d\tilde{P}_0}\right| = \left|\sigma_{P,i} - \sigma_{\tilde{P},i}\right| \tag{2.2}$$

□

De plus, les distances de Wasserstein métrisent la convergence faible (voir, par exemple, Villani, 2009, Théorème 6.9). Par conséquent, la convergence au sens de Wasserstein implique la convergence de nombreuses quantités statistiques intéressantes. La proposition suivante 2.1.2 montre notamment que les distances de Wasserstein limitent les erreurs sur les fonctions quantiles des marginales.

Proposition 2.1.2. *Soit $Q_{\boldsymbol{Y}}$ (respectivement $Q_{\tilde{\boldsymbol{Y}}^{[s]}}$) la fonction quantile de Y (respectivement $\tilde{Y}^{[s]}$), alors :*

$$\frac{1}{\sqrt{d}} \sum_{i=1}^{d} \int_0^1 |Q_{\boldsymbol{Y}}(q) - Q_{\tilde{\boldsymbol{Y}}^{[s]}}(q)| dq \leq W_p(P_{\boldsymbol{Y}}, P_{\tilde{\boldsymbol{Y}}^{[s]}})$$

Preuve de la proposition 2.1.2. Soit Y et $\tilde{Y}$ deux variables aléatoires de $\mathbb{R}^d$ avec pour distributions respectives P et $\tilde{P}$.

$$W_2(P, \tilde{P}) \geq W_1(P, \tilde{P}) = \inf_{\gamma\in\Gamma(P,\tilde{P})} \mathbb{E}\left[\|Y - \tilde{Y}\|_2^1\right] \tag{2.3}$$

Or, $\forall x \in \mathbb{R}^d : \|x\|_2 \geq \frac{1}{\sqrt{d}}\|x\|_1$, donc :

$$W_2(P, \tilde{P}) \geq \frac{1}{\sqrt{d}} \inf_{\gamma\in\Gamma(P,\tilde{P})} \mathbb{E}\left[\|Y - \tilde{Y}\|_1^1\right] \tag{2.4}$$

Et :

$$\begin{aligned} W_1(P,\tilde{P}) &= \inf_{\gamma\in\Gamma(P,\tilde{P})} \mathbb{E}\left[\|Y-\tilde{Y}\|_1\right] = \inf_{\gamma\in\Gamma(P,\tilde{P})} \mathbb{E}\left[\sum_{i=1}^{d}|Y_i-\tilde{Y}_i|\right] \\ &\geq \sum_{i=1}^{d}\inf_{\gamma_i\in\Gamma(P_i,\tilde{P}_i)} \mathbb{E}\left[|Y_i-\tilde{Y}_i|\right] \geq \sum_{i=1}^{d} W_1(P_i,\tilde{P}_i) \end{aligned}$$

La première inégalité précédente est obtenue par inversion de la somme et de l'infimum et en notant que si $\gamma \in \Gamma(P,\tilde{P})$ alors par définition, on a $\gamma_i \in \Gamma(P_i,\tilde{P}_i)$. Puis, lorsque $d=1$:

$$W_p(X,Y) = \left\|F_X^{-1}-F_Y^{-1}\right\|_p = \left(\int_0^1 \left|F_X^{-1}(\alpha)-F_Y^{-1}(\alpha)\right|^p \, \mathrm{d}\alpha\right)^{1/p} \tag{2.5}$$

avec F_X et $F_X^{-1}(q) = \inf\{x : F_X(x) \geq q\}, q \in (0,1)$, la densité et la fonction quantile de X.
Donc, en notant Q (respectivement $\tilde{Q}$) la fonction quantile de P (respectivement $\tilde{P}$) :

$$\frac{1}{\sqrt{d}}\sum_{i=1}^{d}\int_0^1 |Q(q)-\tilde{Q}(q)|dq \leq W_2(P,\tilde{P}) \tag{2.6}$$

□

Cela peut être interprété comme une limite de l'erreur moyenne sur les quantiles des distributions marginales. L'utilisation de la distance de Wasserstein ne permet pas de limiter un quantile arbitraire en général. Contrairement à la métrique KL, les distances de Wasserstein sont utiles pour comparer les mesures de probabilité lorsque la distribution de référence n'est pas dominée par l'autre. Cette situation se présente, par exemple, lorsque l'on compare $P_{\boldsymbol{Y}}$ et $P_{\tilde{\boldsymbol{Y}}^{[s]}}$ pour un réseau de neurones utilisant des fonctions d'activation ReLU. Comme indiqué dans l'annexe C, dans ce contexte, la distance KL entre les distributions est toujours $+\infty$, ce qui rend impossible l'obtention de limites. Même dans le cadre de la Leaky-ReLU, les bornes qui pourraient être obtenues n'auraient aucun sens lorsque la Leaky-ReLU devient proche de la ReLU classique.

Limite supérieur et critère de division

Afin de détecter la non-linéarité du réseau de neurones, nous proposons de s'appuyer sur un critère de Wasserstein entre $P_{\boldsymbol{Y}}$ (la vraie distribution de sortie) et $P_{\tilde{\boldsymbol{Y}}^{[s]}}$ (la distribution de sortie estimée par appariement des moments). Le calcul direct de la distance étant difficile, nous nous appuierons sur l'établissement d'une borne supérieure de ce critère, obtenue en introduisant une distribution intermédiaire $P_{\bar{\boldsymbol{Y}}^{[s]}}$ obtenue en linéarisant localement (développement de Taylor à l'ordre 1) la fonction $\boldsymbol{f}$. La proposition 2.1.3 établit cette borne supérieure.

Proposition 2.1.3. *Soit $W_2(\cdot,\cdot)$ la distance 2-Wasserstein munie de la norme euclidienne usuelle sur $\mathbb{R}^d$. Alors, pour $s \in \mathbb{N}$ et après application de $S^{[s]}(\cdot, n)$ à $P_{\boldsymbol{X}}$, on obtient :*

$$W_2(P_{\boldsymbol{Y}}, P_{\tilde{\boldsymbol{Y}}^{[s]}}) \quad \leq \quad B_s, \tag{2.7}$$

avec

$$\begin{aligned} B_s = L_{2,2}\epsilon_{0,1} \sum_{k=0}^{s} \sqrt{\lambda_\infty^{[k]}} + \\ + \sum_i w_i E_{\boldsymbol{X}\sim P_{\hat{\boldsymbol{X}}^{[s]},i}} \left[\|\boldsymbol{f}(\boldsymbol{X}) - \bar{\boldsymbol{f}}_{\boldsymbol{i}}(\boldsymbol{X})\|_2^2 \right]^{\frac{1}{2}} + \left(\|\mu_{\tilde{\boldsymbol{Y}}^{[s]},i} - \mu_{\bar{\boldsymbol{Y}}^{[s]},i}\|_2^2 \right. \\ \left. + \operatorname{Tr}\left(\Sigma_{\bar{\boldsymbol{Y}}^{[s]},i} + \Sigma_{\tilde{\boldsymbol{Y}}^{[s]},i} - 2\left(\Sigma_{\tilde{\boldsymbol{Y}}^{[s]},i}^{\frac{1}{2}} \Sigma_{\bar{\boldsymbol{Y}}^{[s]},i} \Sigma_{\tilde{\boldsymbol{Y}}^{[s]},i}^{\frac{1}{2}} \right)^{\frac{1}{2}} \right) \right)^{\frac{1}{2}}, \end{aligned} \tag{2.8}$$

où $\bar{\boldsymbol{f}}_{\boldsymbol{i}}$ est la linéarisation locale de Taylor d'ordre 1 de $\boldsymbol{f}$ en $\mu_{\hat{\boldsymbol{X}}^{[s]},i}$ et $\lambda_\infty^{[k]}$ est la plus grande valeur propre de $\Sigma_{\hat{\boldsymbol{X}}^{[k]},\boldsymbol{i}}$ après application de $S^{[k]}(\cdot, n)$ à $P_{\boldsymbol{X}}$. $\mu_{\tilde{\boldsymbol{Y}}^{[s]},i}$ et $\Sigma_{\tilde{\boldsymbol{Y}}^{[s]},i}$ (respectivement $\mu_{\bar{\boldsymbol{Y}}^{[s]},i}$ et $\Sigma_{\bar{\boldsymbol{Y}}^{[s]},i}$) sont les deux premiers moments de $P_{\tilde{\boldsymbol{Y}}^{[s]},i}$ (respectivement $P_{\bar{\boldsymbol{Y}}^{[s]},i}$). $L_{2,2}$ est la constante de Lipschitz de $\boldsymbol{f}$ (par rapport à la norme L_2), et $\epsilon_{0,1}$ l'erreur type de l'opérateur de split S.

L'hypothèse de linéarité locale est utilisée ici de deux manières distinctes. La première est la linéarisation de Taylor du premier ordre, et la seconde utilise la stabilité des distributions gaussiennes par transformations linéaires. De plus, cette borne supérieure est composée de trois parties distinctes. La première partie traduit l'erreur de sortie due à l'erreur commise en divisant la distribution d'entrée ($\epsilon_{0,1} > 0$). La deuxième partie évalue l'erreur due à l'approximation après la linéarisation de Taylor. La dernière partie quantifie la dérive du moment entre $P_{\bar{\boldsymbol{Y}}^{[s]}}$ et $P_{\tilde{\boldsymbol{Y}}^{[s]}}$ induite par l'hypothèse de linéarisation. On peut noter qu'aucune formule analytique ne permet de calculer cette borne supérieure. Cependant, des techniques pour estimer cette quantité sont présentées dans la section 2.1.5.

Afin de démontrer la proposition précédente, nous nous appuyons sur deux petits lemmes :

Lemme 2.1.4. *Soit Q une distribution normale mutlivariée $\mathcal{N}(\mu, \Sigma)$ de moyenne $\mu \in \mathbb{R}^d$ et de matrice de variance/covariance $\Sigma \in \mathcal{S}_d^+(\mathbb{R})$. Soit $\hat{Q}$ une mixture de distribution gaussiennes multivariées telle que $\hat{Q} = S(Q; n) = \sum_{i=1}^n w_i \mathcal{N}(\mu_i, \Sigma_i)$. Soit λ_∞ la plus grande valeur propre de Σ. Alors :*

$$W_2(Q, \hat{Q}) \leq \epsilon_{0,1}\sqrt{\lambda_\infty} \tag{2.9}$$

Preuve du lemme 2.1.4. Soit Q une distribution normale mutlivariée $\mathcal{N}(\mu, \Sigma)$ de moyenne $\mu \in \mathbb{R}^d$ et

de matrice de variance/covariance $\Sigma \in \mathcal{S}_d^+(\mathbb{R})$. Donc :

$$\exists U \in \mathcal{O}_d^+(\mathbb{R}) \quad \exists \Lambda \in \mathcal{D}_d(\mathbb{R}) : \quad \Sigma = U\Lambda U^T$$

.

où $\Lambda = diag(\lambda_1, .., \lambda_d)$. Par conséquent, par la définition de l'opérateur de division S, on a $\Sigma_i = U\Lambda_i U^T$, avec $\Lambda_i = diag(\lambda_1, .., \tilde{\sigma}^2\lambda_j, .., \lambda_d)$, où $j = \text{argmax}_k\, \lambda_k$ (ici la direction de division correspond à la direction de plus forte incertitude (cf. 2.1.5)).

Puis, par orthogonalité de la matrice U, on a :

$$\begin{aligned} W_2(Q, \hat{Q}) &= \inf_{\gamma \in \Gamma(Q,\hat{Q})} \mathbb{E}\left[\|X - \hat{X}\|_2^2\right]^{\frac{1}{2}} = \inf_{\gamma \in \Gamma(Q,\hat{Q})} \mathbb{E}\left[\|UX - U\hat{X}\|_2^2\right]^{\frac{1}{2}} \\ &= \inf_{\gamma \in \Gamma(P,\hat{P})} \mathbb{E}\left[\|X - \hat{X}\|_2^2\right]^{\frac{1}{2}}. \end{aligned} \tag{2.10}$$

avec P la distribution gaussienne multivariée $\mathcal{N}(U\mu, \Lambda)$ et $\hat{P} = \sum_{i=1}^n w_i \mathcal{N}(U\mu_i, \Lambda_i)$. Enfin, l'opérateur de split S modifie uniquement la marginale de P ayant la plus grande valeur propre (λ_∞), notée P_∞, ayant pour moyenne μ_∞ et pour variance σ_∞^2.

$$\begin{aligned} W_2^2(P, \hat{P}) &= \inf_{\gamma \in \Gamma(P,\hat{P})} \mathbb{E}\left[\|X - \hat{X}\|_2^2\right] = \inf_{\gamma \in \Gamma(P,\hat{P})} \mathbb{E}\left[\sum_{i=1}^d (X_i - \hat{X}_i)^2\right] \\ &= \inf_{\gamma \in \Gamma(P,\hat{P})} \mathbb{E}\left[(X_\infty - \hat{X}_\infty)^2\right] + \mathbb{E}\left[\sum_{i \neq \infty} (X_i - \hat{X}_i)^2\right] \\ &\leq \inf_{\gamma \in \Gamma(P_\infty,\hat{P}_\infty)} \mathbb{E}\left[\|X_\infty - \hat{X}_\infty\|_2^2\right] = W_2^2(P_\infty, \hat{P}_\infty) \end{aligned} \tag{2.11}$$

car $\Gamma(P_\infty, \hat{P}_\infty) \subset \Gamma(P, \hat{P})$. $\Gamma(P_\infty, \hat{P}_\infty)$ est choisi comme la sous partie de $\Gamma(P, \hat{P})$ où l'on restreint à l'identité pour toutes les dimensions à l'exception de celle présentant la plus forte valeur propre.

Donc :

$$\begin{aligned} W_2(P, \hat{P}) &\leq W_2(P_\infty, \hat{P}_\infty) = \inf_{\gamma \in \Gamma(P_\infty,\hat{P}_\infty)} \mathbb{E}\left[\|X_\infty - \hat{X}_\infty\|_2^2\right]^{\frac{1}{2}} \\ &= \sigma_\infty \inf_{\gamma \in \Gamma(P_\infty,\hat{P}_\infty)} \mathbb{E}\left[\|\frac{X_\infty - \mu_\infty}{\sigma_\infty} - \frac{\hat{X}_\infty - \mu_\infty}{\sigma_\infty}\|_2^2\right]^{\frac{1}{2}} \\ &= \sigma_\infty \epsilon_{0,1} = \sqrt{\lambda_\infty}\epsilon_{0,1} \end{aligned} \tag{2.12}$$

□

Le lemme précédent établit que la distance de 2-Wasserstein entre une distribution gaussienne et son image par un opérateur de split S est inférieur à l'erreur standard de l'opérateur de split fois la racine carrée de la valeur propre maximale (identifiée ici comme un écart-type σ_∞) de la matrice de covariance de la distribution d'entrée. Le lemme suivant présente un résultat similaire lors de la composition de s opérateur de split sur la distribution d'entrée.

Lemme 2.1.5. *Soit Q une distribution normale multivariée $\mathcal{N}(\mu, \Sigma)$ de moyenne $\mu \in \mathbb{R}^d$ et de matrice de variance covariance $\Sigma \in \mathcal{S}_d^+(\mathbb{R})$. Soit $\hat{Q}^{[s]}$ la mixture de distribution gaussiennes multivariées telle que $\hat{Q}^{[s]} = S^{[s]}(Q; n) = \sum_{i=1}^{M} w_i \mathcal{N}(\mu_i^{[s]}, \Sigma_i^{[s]})$, $s \in \mathbb{N}$. Soit $\lambda_\infty^{[k]}$ la plus grande valeur propre de $\Sigma^{[k]}$ où $\Sigma^{[k]}$ est la matrice de variance covariance de chaque composante de $\hat{Q}^{[k]}$ obtenue après application de k étapes de division. Alors :*

$$W_2(Q, \hat{Q}^{[s]}) \leq \epsilon_{0,1} \sum_{k=0}^{s-1} \sqrt{\lambda_\infty^{[k]}} \tag{2.13}$$

Preuve du lemme 2.1.5. Soit Q une distribution normale multivariée $\mathcal{N}(\mu, \Sigma)$ de moyenne $\mu \in \mathbb{R}^d$ et de matrice de variance covariance $\Sigma \in \mathcal{S}_d^+(\mathbb{R})$. Soit $\hat{Q}^{[s]}$ la mixture de distribution gaussiennes multivariées telle que $\hat{Q}^{[s]} = S^{[s]}(Q; n) = \sum_{i=1}^{M} w_i \mathcal{N}(\mu_i^{[s]}, \Sigma_i^{[s]})$, $s \in \mathbb{N}$. Soit $\lambda_\infty^{[k]}$ la plus grande valeur propre de $\Sigma^{[k]}$ où $\Sigma^{[k]}$ est la matrice de variance covariance de chaque composante de $\hat{Q}^{[k]}$ obtenue après application de k étapes de division.

$$W_2(Q, \hat{Q}^{[s]}) = W_2\left(Q, S^{[s]}(Q; n)\right) \leq \sum_{k=0}^{s-1} W_2\left(S^{[k]}(Q; n), S^{[k+1]}(Q; n)\right) \tag{2.14}$$

Avec $S^{[0]}(Q; n) = Q$

Et :

$$\begin{aligned} W_2\left(S^{[k]}(Q; n), S^{[k+1]}(Q; n)\right) &\leq \sum_{i=1}^{M} w_i W_2\left(\hat{Q}_i^{[k]}, S(\hat{Q}_i^{[k]}; n)\right) \\ &\leq \sum_{i=1}^{M} w_i W_2\left(\mathcal{N}(\mu_i^{[k]}, \Sigma_i^{[k]}), S(\mathcal{N}(\mu_i^{[k]}, \Sigma_i^{[k]}); n)\right) \end{aligned} \tag{2.15}$$

Or, pour un même niveau de division k, toutes les composantes de la mixture ont la même matrice de variance covariance, i.e. $\forall i \in \{1...M\} : \quad \Sigma_i^{[k]} = \Sigma_0^{[k]}$. Par conséquent, avec $\sum_i w_i = 1$, et en notant $\lambda_\infty^{[k]}$ la plus grande valeur propre de $\Sigma_0^{[k]}$, et en utilisant le Lemme 2.1.4, on obtient :

$$W_2(Q, \hat{Q}^{[s]}) \leq \epsilon_{0,1} \sum_{k=0}^{s-1} \sqrt{\lambda_\infty^{[k]}} \tag{2.16}$$

□

Preuve de la proposition 2.1.3. Soit $W_2(\cdot,\cdot)$ la distance 2-Wasserstein munie de la norme euclidienne usuelle sur $\mathbb{R}^d$. Avec les notations générales, on a :

$$W_2(P_{\boldsymbol{Y}}, P_{\tilde{\boldsymbol{Y}}^{[s]}}) \leq W_2(P_{\boldsymbol{Y}}, P_{\hat{\boldsymbol{Y}}^{[s]}}) + W_2(P_{\hat{\boldsymbol{Y}}^{[s]}}, P_{\tilde{\boldsymbol{Y}}^{[s]}}) \tag{2.17}$$

Avec :

$$\begin{aligned} W_2(P_{\boldsymbol{Y}}, P_{\hat{\boldsymbol{Y}}^{[s]}}) &= \inf_{\gamma\in\Gamma(P_{\boldsymbol{Y}}, P_{\hat{\boldsymbol{Y}}^{[s]}})} \mathbb{E}\left[\|Y-\hat{Y}\|_2^2\right]^{\frac{1}{2}} \\ &\leq \inf_{\gamma\in\Gamma(P_{\boldsymbol{X}}, P_{\hat{\boldsymbol{X}}^{[s]}})} \mathbb{E}\left[\|\boldsymbol{f}(X)-\boldsymbol{f}(\hat{X})\|_2^2\right]^{\frac{1}{2}} \\ &\leq L_{2,2} \inf_{\gamma\in\Gamma(P_{\boldsymbol{X}}, P_{\hat{\boldsymbol{X}}^{[s]}})} \mathbb{E}\left[\|X-\hat{X}\|_2^2\right]^{\frac{1}{2}} \\ &\leq L_{2,2} W_2(P_{\boldsymbol{X}}, P_{\hat{\boldsymbol{X}}^{[s]}}) \\ &\leq L_{2,2}\epsilon_{0,1} \sum_{k=0}^{s-1} \sqrt{\lambda_\infty^{[k]}} \end{aligned} \tag{2.18}$$

en utilisant le lemme 2.1.5.

Et :

$$\begin{aligned} W_2\left(P_{\hat{\boldsymbol{Y}}^{[s]}}, P_{\tilde{\boldsymbol{Y}}^{[s]}}\right) &\leq \sum_{i=1}^{M} w_i W_2\left(P_{\hat{\boldsymbol{Y}}^{[s]},i}, P_{\tilde{\boldsymbol{Y}}^{[s]},i}\right) \\ &\leq \sum_{i=1}^{M} w_i \left(W_2(P_{\hat{\boldsymbol{Y}}^{[s]},i}, P_{\bar{\boldsymbol{Y}}^{[s]},i}) + W_2(P_{\bar{\boldsymbol{Y}}^{[s]},i}, P_{\tilde{\boldsymbol{Y}}^{[s]},i})\right) \end{aligned} \tag{2.19}$$

Avec :

$$W_2(P_{\tilde{\boldsymbol{Y}}^{[s]},i}, P_{\hat{\boldsymbol{Y}}^{[s]},i}) = \left(\|\mu_{\bar{\boldsymbol{Y}}^{[s]},i} - \mu_{\hat{\boldsymbol{Y}}^{[s]},i}\|_2^2 + Tr(\Sigma_{\bar{\boldsymbol{Y}}^{[s]},i} + \Sigma_{\hat{\boldsymbol{Y}}^{[s]},i} - 2\left(\Sigma_{\hat{\boldsymbol{Y}}^{[s]},i}^{\frac{1}{2}} \Sigma_{\bar{\boldsymbol{Y}}^{[s]},i} \Sigma_{\hat{\boldsymbol{Y}}^{[s]},i}^{\frac{1}{2}}\right)^{\frac{1}{2}}\right)\right)^{\frac{1}{2}} \tag{2.20}$$

Et :

$$\begin{aligned} W_2^2(P_{\hat{\boldsymbol{Y}}^{[s]},i}, P_{\bar{\boldsymbol{Y}}^{[s]},i}) &= \inf_{\gamma\in\Gamma(P_{\hat{\boldsymbol{Y}}^{[s]},i}, P_{\bar{\boldsymbol{Y}}^{[s]},i})} \mathbb{E}\left[\|\hat{\boldsymbol{Y}}^{[s]} - \bar{\boldsymbol{Y}}^{[s]}\|_2^2\right] \\ &\leq \mathbb{E}_{\boldsymbol{X}\sim P_{\hat{\boldsymbol{X}}^{[s]},i}}\left[\|\boldsymbol{f}(\boldsymbol{X}) - \bar{\boldsymbol{f}}_{\boldsymbol{i}}(\boldsymbol{X})\|_2^2\right] \end{aligned} \tag{2.21}$$

En choisissant γ entièrement défini par $\boldsymbol{X} \sim P_{\hat{\boldsymbol{X}}^{[s]},i}$ et tel que $(\hat{\boldsymbol{Y}}^{[s]}, \bar{\boldsymbol{Y}}^{[s]}) = (\boldsymbol{f}(\boldsymbol{X}), \bar{\boldsymbol{f}}_{\boldsymbol{i}}(\boldsymbol{X}))$.

Cela nous fournit le résultat recherché.

□

Garanties théoriques de convergence

Nous montrons ici la convergence de cette borne supérieure au sein de la proposition 2.1.6.

Proposition 2.1.6. *Soit $R = rang(\Sigma_{\boldsymbol{X}})$ et $(\lambda_1, ..., \lambda_R)$ les valeurs propres non nulles de $\Sigma_{\boldsymbol{X}}$ classées par ordre décroissant, alors $\exists C \in \mathbb{R}_+$ tel que :*

$$B_s \underset{s\to\infty}{\longrightarrow} L_{2,2}\epsilon_{0,1}\left(C + \frac{R\sqrt{\lambda_R}}{1-\tilde{\sigma}}\right),$$

avec $C = \sum_{k=0}^{r_R}\sqrt{\lambda_\infty^{[k]}}$, et $r_R = \sum_{i=1}^{R-1}\left\lceil \frac{\ln\lambda_i - \ln\lambda_R}{\ln\tilde{\sigma}^2}\right\rceil$

Cela nous fournit immédiatement le corollaire suivant.

Corollaire 2.1.7. *Avec les notations de la proposition 2.1.6 :*

$$\limsup W_2(P_{\boldsymbol{Y}}, P_{\tilde{\boldsymbol{Y}}^{[s]}}) \leq L_{2,2}\epsilon_{0,1}\left(C + \frac{R\sqrt{\lambda_R}}{1-\tilde{\sigma}}\right).$$

Appliquer un nombre infini de fois l'opérateur de division engendre la convergence de notre critère vers une constante. Cette constante dépend directement de l'erreur type de l'opérateur de division $\epsilon_{0,1}$, de la constante de Lipschitz $L_{2,2}$ du réseau de neurones $\boldsymbol{f}$, du niveau de division n et de l'incertitude d'entrée initiale traduit par les valeurs propres initiales de $\Sigma_{\boldsymbol{X}}$. On montre que cette borne de convergence dépend de l'erreur initiale commise lors de la division de la distribution initiale comme mixture de gaussiennes. $\epsilon_{0,1}$ peut être réduit en utilisant un opérateur de division avec un nombre plus important de composantes (plus grand n) et de plus faibles variances (plus faible $\tilde{\sigma}$). Le nombre d'itérations utilisant ce nouvel opérateur de division sera également réduit, et donc cette borne tient avec le nouvel $\epsilon_{0,1}$ et l'ancienne constante.

Afin de démontrer cette convergence, nous fournissons et démontrons le lemme suivant :

Lemme 2.1.8. *Soit $x = [x_1, ..., x_d]$ un vecteur de $\mathbb{R}_+^d$. Soit g la fonction telle que : $g(x) = [x_1, ..., \tilde{\sigma}^2 x_j, ..., x_d]$, avec $j = \text{argmax}_i\, x_i$ et $0 \leq \tilde{\sigma} \leq 1$, et $g^{[k]}$ la concaténation de g, k fois. Sans perte de généralité, on suppose que $\forall i \in \{1...d-1\} \quad x_i > x_d$. Alors :*

$$\sum_{k=1}^{s}\sqrt{\|g^{[k]}(x)\|_\infty} \underset{s\to\infty}{\longrightarrow} \left(C + \frac{d\sqrt{x_d}}{1-\tilde{\sigma}}\right) \tag{2.22}$$

où $C = \sum_{k=0}^{r_d}\sqrt{\|g^{[k]}\|_\infty}$, et $r_d = \sum_{i=1}^{d-1}\left\lceil \frac{\ln x_i - \ln x_d}{\ln\tilde{\sigma}^2}\right\rceil$

Preuve du lemme 2.1.8. On note $x^{[k]} = g^{[k]}(x)$. Pour tout $i \in \{1...d-1\}$, on note N_i le plus petit entier tel que $(\tilde{\sigma}^2)^{N_i} x_i < x_d$.

On a $N_i = \left\lceil \frac{\ln x_i - \ln x_d}{\ln \tilde{\sigma}^2} \right\rceil$. Donc après application de $r_d = \sum_{i=1}^{d-1} N_i = \sum_{i=1}^{d-1} \left\lceil \frac{\ln x_i - \ln x_d}{\ln \tilde{\sigma}^2} \right\rceil$ fois g à x, on obtient $\forall i \in \{1...d-1\} \quad \tilde{\sigma}^2 x_d \leq x_i^{[r_d]} \leq x_d$ (l'inégalité de gauche peut être obtenue en raisonnant par l'absurde).

Plus généralement : $\forall m \in \mathbb{N} \quad \forall i \in \{1...d-1\} \quad (\tilde{\sigma}^2)^{m+1} x_d \leq x_i^{[r_d+md]} \leq (\tilde{\sigma}^2)^m x_d$ (par récurrence).
Et : $\forall m \in \mathbb{N} \quad \forall i \in \{1...d-1\} \quad \forall p \in \{0...d-1\} \quad x_i^{[r_d+md+p]} \leq x_i^{[r_d+md]}$

$$\begin{aligned}
\sum_{k=0}^{\infty} \sqrt{\|g^{[k]}(x)\|_\infty} &= \sum_{k=0}^{r_d} \sqrt{\|g^{[k]}(x)\|_\infty} + \sum_{k=0}^{\infty} \sum_{p=0}^{d-1} \sqrt{\|g^{[r_d+kd+p]}(x)\|_\infty} \\
&\leq \sum_{k=0}^{r_d} \sqrt{\|g^{[k]}(x)\|_\infty} + d \sum_{k=0}^{\infty} \sqrt{\|g^{[r_d+kd]}(x)\|_\infty} \\
&\leq \sum_{k=0}^{r_d} \sqrt{\|g^{[k]}(x)\|_\infty} + d \sum_{k=0}^{\infty} \sqrt{(\tilde{\sigma}^2)^k x_d} \\
&\leq \sum_{k=0}^{r_d} \sqrt{\|g^{[k]}(x)\|_\infty} + \frac{d\sqrt{x_d}}{1-\tilde{\sigma}} \qquad (2.23)
\end{aligned}$$

finissant la preuve. □

Preuve de la proposition 2.1.6. On note $V_s = \sum_i w_i E_{\boldsymbol{X} \sim P_{\hat{\boldsymbol{X}}^{[s]},i}} \left[\|\boldsymbol{f}(\boldsymbol{X}) - \bar{\boldsymbol{f}}_{\boldsymbol{i}}(\boldsymbol{X})\|_2^2 \right]^{\frac{1}{2}} + \Big(\|\mu_{\tilde{\boldsymbol{Y}}^{[s]},i} - \mu_{\bar{\boldsymbol{Y}}^{[s]},i}\|_2^2 +$ $\mathrm{Tr}\left(\Sigma_{\bar{\boldsymbol{Y}}^{[s]},i} + \Sigma_{\tilde{\boldsymbol{Y}}^{[s]},i} - 2\left(\Sigma^{\frac{1}{2}}_{\tilde{\boldsymbol{Y}}^{[s]},i} \Sigma_{\bar{\boldsymbol{Y}}^{[s]},i} \Sigma^{\frac{1}{2}}_{\tilde{\boldsymbol{Y}}^{[s]},i} \right)^{\frac{1}{2}} \right) \Big)^{\frac{1}{2}}$.

Dans la suite, on montre que $V_s \underset{s \to \infty}{\longrightarrow} 0$.

$$\begin{aligned}
\|\mu_{\tilde{\boldsymbol{Y}}^{[s]},i} - \mu_{\bar{\boldsymbol{Y}}^{[s]},i}\|_2^2 &= \|E_{\boldsymbol{X} \sim P_{\hat{\boldsymbol{X}}^{[s]},i}} \left[\boldsymbol{f}(\boldsymbol{X}) - \bar{\boldsymbol{f}}_{\boldsymbol{i}}(\boldsymbol{X}) \right] \|_2^2 \\
&\leq E_{\boldsymbol{X} \sim P_{\hat{\boldsymbol{X}}^{[s]},i}} \left[\|\boldsymbol{f}(\boldsymbol{X}) - \bar{\boldsymbol{f}}_{\boldsymbol{i}}(\boldsymbol{X})\|_2^2 \right] \qquad (2.24)
\end{aligned}$$

Où $\bar{\boldsymbol{f}}_{\boldsymbol{i}}$ est la linéarisation locale de Taylor au premier ordre de $\boldsymbol{f}$ en $\mu_{\hat{\boldsymbol{X}}^{[s]},i}$. Avec $J_{\boldsymbol{f}}(\mu_{\hat{\boldsymbol{X}}^{[s]},i})$ la

jacobienne de $\boldsymbol{f}$ en $\mu_{\hat{\boldsymbol{X}}^{[s]},i}$, on a $\bar{\boldsymbol{f}}_{\boldsymbol{i}}(X) = \boldsymbol{f}(\mu_{\hat{\boldsymbol{X}}^{[s]},i}) + J_{\boldsymbol{f}}(\mu_{\hat{\boldsymbol{X}}^{[s]},i})(\boldsymbol{X} - \mu_{\hat{\boldsymbol{X}}^{[s]},i})$. Donc :

$$\begin{aligned}
E_{\boldsymbol{X}\sim P_{\hat{\boldsymbol{X}}^{[s]},i}}\left[\|\boldsymbol{f}(\boldsymbol{X}) - \bar{\boldsymbol{f}}_{\boldsymbol{i}}(\boldsymbol{X})\|_2^2\right] &= E_{\boldsymbol{X}\sim P_{\hat{\boldsymbol{X}}^{[s]},i}}\left[\|\boldsymbol{f}(\boldsymbol{X}) - \boldsymbol{f}(\mu_{\hat{\boldsymbol{X}}^{[s]},i}) - J_{\boldsymbol{f}}(\mu_{\hat{\boldsymbol{X}}^{[s]},i})(\boldsymbol{X} - \mu_{\hat{\boldsymbol{X}}^{[s]},i})\|_2^2\right] \\
&\leq 2(E_{\boldsymbol{X}\sim P_{\hat{\boldsymbol{X}}^{[s]},i}}\left[\|\boldsymbol{f}(\boldsymbol{X}) - \boldsymbol{f}(\mu_{\hat{\boldsymbol{X}}^{[s]},i})\|_2^2\right] \\
&+ E_{\boldsymbol{X}\sim P_{\hat{\boldsymbol{X}}^{[s]},i}}\left[\|J_{\boldsymbol{f}}(\mu_{\hat{\boldsymbol{X}}^{[s]},i})(\boldsymbol{X} - \mu_{\hat{\boldsymbol{X}}^{[s]},i})\|_2^2\right]) \\
&\leq 2(L_{2,2}^2 + |||J_{\boldsymbol{f}}(\mu_{\hat{\boldsymbol{X}}^{[s]},i})|||^2)E_{\boldsymbol{X}\sim P_{\hat{\boldsymbol{X}}^{[s]},i}}\left[\|\boldsymbol{X} - \mu_{\hat{\boldsymbol{X}}^{[s]},i}\|_2^2\right] \\
&\leq 2(L_{2,2}^2 + |||J_{\boldsymbol{f}}(\mu_{\hat{\boldsymbol{X}}^{[s]},i})|||^2)E_{\boldsymbol{X}\sim P_{\hat{\boldsymbol{X}}^{[s]},i}}\left[\sum_{k=1}^{d}(\boldsymbol{X}_k - \mu_{\hat{\boldsymbol{X}}^{[s]},i,k})^2\right] \\
&\leq 2(L_{2,2}^2 + |||J_{\boldsymbol{f}}(\mu_{\hat{\boldsymbol{X}}^{[s]},i})|||^2)\sum_{k=1}^{d}E_{\boldsymbol{X}\sim P_{\hat{\boldsymbol{X}}^{[s]},i}}\left[(\boldsymbol{X}_k - \mu_{\hat{\boldsymbol{X}}^{[s]},i,k})^2\right] \\
&\leq 2(L_{2,2}^2 + |||J_{\boldsymbol{f}}(\mu_{\hat{\boldsymbol{X}}^{[s]},i})|||^2)\mathrm{Tr}(\Sigma_{\hat{\boldsymbol{X}}^{[s]},i}) \qquad (2.25)
\end{aligned}$$

où $|||\cdot|||$ dénote la norme matricielle, et Tr l'opérateur trace.

$$\mathrm{Tr}\left(\Sigma_{\bar{\boldsymbol{Y}}^{[s]},i} + \Sigma_{\tilde{\boldsymbol{Y}}^{[s]},i} - 2\left(\Sigma_{\tilde{\boldsymbol{Y}}^{[s]},i}^{\frac{1}{2}}\Sigma_{\bar{\boldsymbol{Y}}^{[s]},i}\Sigma_{\tilde{\boldsymbol{Y}}^{[s]},i}^{\frac{1}{2}}\right)^{\frac{1}{2}}\right) \leq \mathrm{Tr}\left(\Sigma_{\bar{\boldsymbol{Y}}^{[s]},i} + \Sigma_{\tilde{\boldsymbol{Y}}^{[s]},i}\right) \qquad (2.26)$$

Et :

$$\begin{aligned}
\mathrm{Tr}\left(\Sigma_{\tilde{\boldsymbol{Y}}^{[s]},i}\right) &= \mathrm{Tr}\left(\mathrm{Cov}(\tilde{Y}^{[s],i})\right) = \mathrm{Tr}\left(\mathrm{Cov}\left(\boldsymbol{f}(\hat{X}^{[s],i})\right)\right) \\
&= \mathrm{Tr}\left(\mathrm{Cov}\left(\boldsymbol{f}(\hat{X}^{[s],i}) - E_{X'}[\boldsymbol{f}(X')]\right)\right) \\
&\leq E_{\hat{X}^{[s],i}}[\|\boldsymbol{f}(\hat{X}^{[s],i}) - E_{X'}[\boldsymbol{f}(X')]\|_2^2] \\
&= E_{\hat{X}^{[s],i}}[\|E_{X'}[\boldsymbol{f}(\hat{X}^{[s],i}) - \boldsymbol{f}(X')]\|_2^2] \\
&= L_{2,2}^2 E_{\hat{X}^{[s],i}}[\|E_{X'}[\hat{X}^{[s],i} - X']\|_2^2] \\
&\leq L_{2,2}^2 E_{\hat{X}^{[s],i}}[E_{X'}[\|\hat{X}^{[s],i} - X'\|_2^2]] \\
&= L_{2,2}^2 E_{\hat{X}^{[s],i}}[E_{X'}[\|(\hat{X}^{[s],i} - E_{\hat{X}^{[s],i}}[\hat{X}^{[s],i}]) - (X' - E_{X'}[X'])\|_2^2]] \\
&\leq 2L_{2,2}^2(E_{\hat{X}^{[s],i}}[\|\hat{X}^{[s],i} - E_{\hat{X}^{[s],i}}[\hat{X}^{[s],i}]\|_2^2] + E_{X'}[\|X' - E_{X'}[X']\|_2^2]) \\
&= 4L_{2,2}^2 E_{\hat{X}^{[s],i}}[\|\hat{X}^{[s],i} - E_{\hat{X}^{[s],i}}[\hat{X}^{[s],i}]\|_2^2] \\
&= 4L_{2,2}^2\mathrm{Tr}\left(\mathrm{Cov}\left(\hat{X}^{[s],i} - E_{\hat{X}^{[s],i}}[\hat{X}^{[s],i}]\right)\right) = 4L_{2,2}^2\mathrm{Tr}\left(\Sigma_{\hat{\boldsymbol{X}}^{[s]},i}\right) \qquad (2.27)
\end{aligned}$$

Où X' est une copie i.i.d de $\hat{X}^{[s],i}$. Finalement :

$$\begin{aligned}
\mathrm{Tr}\left(\Sigma_{\bar{\boldsymbol{Y}}^{[s]},i}\right) &= \mathrm{Tr}\left(\mathrm{Cov}\left(\bar{Y}^{[s],i}\right)\right) = \mathrm{Tr}\left(\mathrm{Cov}\left(\bar{\boldsymbol{f}}(\hat{X}^{[s],i})\right)\right) \\
&= \mathrm{Tr}\left(\mathrm{Cov}\left(\boldsymbol{f}(\mu_{\hat{\boldsymbol{X}}^{[s]},i}) + J_{\boldsymbol{f}}(\mu_{\hat{\boldsymbol{X}}^{[s]},i})(\hat{X}^{[s],i} - \mu_{\hat{\boldsymbol{X}}^{[s]},i})\right)\right) \\
&= \mathrm{Tr}\left(\mathrm{Cov}\left(J_{\boldsymbol{f}}(\mu_{\hat{\boldsymbol{X}}^{[s]},i})(\hat{X}^{[s],i} - \mu_{\hat{\boldsymbol{X}}^{[s]},i})\right)\right) \\
&\leq E_{\hat{X}^{[s],i}}[\|J_{\boldsymbol{f}}(\mu_{\hat{\boldsymbol{X}}^{[s]},i})(\hat{X}^{[s],i} - \mu_{\hat{\boldsymbol{X}}^{[s]},i})\|_2^2] \\
&\leq |||J_{\boldsymbol{f}}(\mu_{\hat{\boldsymbol{X}}^{[s]},i})|||^2 E_{\hat{X}^{[s],i}}[\|\hat{X}^{[s],i} - \mu_{\hat{\boldsymbol{X}}^{[s]},i}\|_2^2] \\
&= |||J_{\boldsymbol{f}}(\mu_{\hat{\boldsymbol{X}}^{[s]},i})|||^2 \mathrm{Tr}\left(\mathrm{Cov}\left(\hat{X}^{[s],i} - E_{\hat{X}^{[s],i}}[\hat{X}^{[s],i}]\right)\right) \\
&= |||J_{\boldsymbol{f}}(\mu_{\hat{\boldsymbol{X}}^{[s]},i})|||^2 \mathrm{Tr}\left(\mathrm{Cov}\left(\Sigma_{\hat{\boldsymbol{X}}^{[s]},i}\right)\right)
\end{aligned} \tag{2.28}$$

Et $\forall i \in \{1...M\}$: $\Sigma_{\hat{\boldsymbol{X}}^{[s]},i} = \Sigma_{\hat{\boldsymbol{X}}^{[s]},0}$, en notant $J_{[s]} = \max_i |||J_{\boldsymbol{f}}(\mu_{\hat{\boldsymbol{X}}^{[s]},i})|||$ et avec $\sum_i \omega_i = 1$:

$$V_s \leq \sqrt{2\left(L_{2,2}^2 + J_{[s]}^2\right)\mathrm{Tr}\left(\Sigma_{\hat{\boldsymbol{X}}^{[s]},0}\right)} + \sqrt{\left(6L_{2,2}^2 + 3J_{[s]}^2\right)\mathrm{Tr}\left(\Sigma_{\hat{\boldsymbol{X}}^{[s]},0}\right)} \tag{2.29}$$

$$V_s \underset{s\to\infty}{\longrightarrow} 0 \tag{2.30}$$

Car $\Sigma_{\hat{\boldsymbol{X}}^{[s]},i} \underset{s\to\infty}{\longrightarrow} 0$ et $J_{[s]} \leq L_{2,2}$ par l'hypothèse Lipschitz de $\boldsymbol{f}$.

En utilisant le lemme 2.1.8 on obtient le résultat désiré. □

Remarque 9.

La preuve de la proposition 2.1.6 introduit la jacobienne de $\boldsymbol{f}$. Elle émet l'hypothèse que $\boldsymbol{f}$ est C^1 de telle sorte que $J_{\boldsymbol{f}}(x)$ est la matrice jacobienne de $\boldsymbol{f}$ en $x \in \mathbb{R}^d$. La preuve nécessite uniquement que $\bar{\boldsymbol{f}}_{\boldsymbol{i}}$ soit une application linéaire de la forme $\boldsymbol{x} \mapsto \boldsymbol{f}(\mu_{\hat{\boldsymbol{X}}^{[s]},i}) + A(\boldsymbol{x} - \mu_{\hat{\boldsymbol{X}}^{[s]},i})$, avec une borne sur $|||A|||$. C'est le cas des gradients calculés par les algorithmes standard de rétropropagation avec des fonctions d'activation Lipschitz telles que ReLU où Leaky ReLU.

2.1.5 Implémentation pratique

Les sous-sections suivantes présentent les détails de la mise en œuvre pratique de la méthode proposée.

Division gaussienne

Nous rappelons ici que la division d'une distribution gaussienne multivariée consiste à diviser en "petits morceaux" la distribution dans une direction spécifique, suivant les formules suivantes :

$$w_i = w\tilde{w}_i, \quad \boldsymbol{\mu_i} = \boldsymbol{\mu} + \tilde{\mu}_i\sqrt{\lambda_j}\boldsymbol{\nu_j}, \quad \tilde{\boldsymbol{\Lambda}} = diag(\lambda_1, \ldots, \tilde{\sigma}^2\lambda_j, \ldots, \lambda_k) \tag{2.31}$$

où $(\lambda_r)_{1\leq r\leq R}$ sont les valeurs propres de $\boldsymbol{\Sigma}$, j est la direction de division[1], $\tilde{w}_i$, $\tilde{\mu}_i$ et $\tilde{\sigma}$ sont des paramètres fixes, présentés dans la table 2.1.

TABLE 2.1 – Table des paramètres de division pour une distribution gaussienne univariée, centrée et réduite.

# composants	$\tilde{w}_i$	$\tilde{\beta}_i$	$\tilde{\sigma}$	$\epsilon_{0,1}(\times 10^{-5})$
2	0.6364	0.0		
	0.1818	$\pm$1.0579	0.005	5.57
3	0.4444	0.0		
	0.2455	$\pm$0.9332	0.008	4.42
	0.0323	$\pm$1.9776		
4	0.3048	0.0		
	0.2410	$\pm$0.7056		
	0.0948	$\pm$1.4992	0.019	4.18
	0.0118	$\pm$2.4601		

Cette table apparaît originellement dans les travaux de (DeMars et al., 2013) puis fut complétée par le travail de (Zhang and Shin, 2021), à laquelle nous avons ajouté les valeurs de $\epsilon_{0,1}$. L'ensemble de ces paramètres peuvent être obtenu en résolvant le problème d'optimisation suivant (en utilisant, par exemple, la méthode *fmincon* de Matlab (**?**)) :

$$\begin{aligned} &\min_{\tilde{w}_i,\tilde{\beta}_i,\tilde{\sigma}} D_{KL}(\mathcal{N}(0,1))\|S^{[1]}(\mathcal{N}(0,1))) + \alpha n\tilde{\sigma} \\ &\text{subject to } \textstyle\sum_{i=1}^{n}\tilde{w}_i = 1, \quad 0 < \tilde{\sigma} < 1 \end{aligned} \tag{2.32}$$

où α est un facteur d'échelle et n le nombre de composants dans la mixture résultante.

Direction de division : Dans les équations précédentes, j correspond à la direction dans laquelle on *divise* la distribution gaussienne. Cette direction peut être définie de différentes manières :

1. **Direction de plus forte incertitude :** La direction de division correspond à la direction de plus grande incertitude, c'est-à-dire la direction ayant la valeur propre la plus élevée de la matrice de variance covariance. C'est le choix par défaut dans les méthodes de l'état de l'art car elle est simple d'implémentation et ne rajoute pas de complexité calculatoire. De plus, il

1. l'axe selon lequel les termes PDF du mélange gaussien sont divisés

est intéressant de noter que dans ce paradigme de division, toutes les composantes divisées en même temps ont la même matrice de covariance, et donc les mêmes valeurs propres. Cela permet notamment de réduire la mémoire employée par l'algorithme,

2. **Direction de plus forte non-linéarité :** Faubel and Klakow (2010) introduisent la direction de plus forte non-linéarité comme le vecteur propre associé à la plus forte valeur propre de $\Psi = \sum_{j=0}^{n-1} \eta_j \phi_j \phi_j^T$ où $\eta_j = \frac{1}{2} \left\| Y_{i,[j+d],L-1} + Y_{i,[j],L-1} - 2Y_{i,[0],L-1} \right\|^2$ et $\phi_j = \frac{Z_{i,[j]} - Z_{i,[0]}}{\left\| Z_{i,[j]} - Z_{i,[0]} \right\|}$, et $Z_{i,[j]}$ sont les échantillons UT présentées dans la sous-section suivante, et $Y_{i,[j+d],L-1} = \boldsymbol{f}(Z_{i,[j]})$.
3. **Direction de plus forte non-gaussianité :** Straka et al. (2016) introduisent la direction de plus forte non-gaussianité en se basant sur le calcul du moment d'ordre 3 de la distribution gaussienne propagée au sein de la fonction non-linéaire considérée.

Dans l'ensemble des expériences présentées dans la section 2.2, nous avons sélectionné comme direction de division la direction de plus forte incertitude. Néanmoins, et comme le montrera la sous-section 2.2.4, pour des bruits diagonaux comme le bruit gaussien, il est préférable d'utiliser des alternatives à la direction de plus forte incertitude.

Appariement des moments et estimation de la borne supérieure

Unscented Transform : Pour estimer la borne supérieure de la proposition 2.1.3, nous proposons de nous appuyer sur l'échantillonnage UT. UT est une technique d'échantillonnage qui permet d'estimer les deux premiers moments de la distribution $P_{\hat{\boldsymbol{X}}^{[s]},i} = \mathcal{N}(\mu_{\hat{\boldsymbol{X}}^{[s]},i}, \Sigma_{\hat{\boldsymbol{X}}^{[s]},i})$ à travers une transformation non linéaire $\boldsymbol{f}$ en utilisant $2d+1$ échantillons pondérés appelés points sigma, définis comme suit :

$$
\begin{aligned}
Z_{i,[0]} &= \mu_{\hat{\boldsymbol{X}}^{[s]},i} & \boldsymbol{\omega}_0 &= \frac{\kappa}{d+\kappa} \\
Z_{i,[j]} &= \mu_{\hat{\boldsymbol{X}}^{[s]},i} + (\sqrt{(d+\kappa)}\boldsymbol{U}_{\hat{\boldsymbol{X}}^{[s]},i})_{[j]} & \boldsymbol{\omega}_j &= \frac{1}{2(d+\kappa)} \\
Z_{i,[j+d]} &= \mu_{\hat{\boldsymbol{X}}^{[s]},i} - (\sqrt{(d+\kappa)}\boldsymbol{U}_{\hat{\boldsymbol{X}}^{[s]},i})_{[j]} & \boldsymbol{\omega}_{j+d} &= \frac{1}{2(d+\kappa)}
\end{aligned}
$$

$\forall j \in \{1, \cdots, d\}$, où $\Sigma_{\hat{\boldsymbol{X}}^{[s]},i} = \boldsymbol{U}_{\hat{\boldsymbol{X}}^{[s]},\boldsymbol{i}} \boldsymbol{U}^{\boldsymbol{T}}_{\hat{\boldsymbol{X}}^{[s]},\boldsymbol{i}}$, $(\sqrt{(d+\kappa)}\boldsymbol{U}_{\hat{\boldsymbol{X}}^{[s]},i})_{[j]}$ est la j-e colonne de $\sqrt{(d+\kappa)}\boldsymbol{U}_{\hat{\boldsymbol{X}}^{[s]},i}$, $\boldsymbol{\omega}_j$ est le poids associé au j-e point sigma , et κ est un paramètre libre, généralement fixé à $\kappa = d-3$. Les points sigma sont utilisés pour estimer $\mu_{\tilde{\boldsymbol{Y}}^{[s]},i}$, $\Sigma_{\tilde{\boldsymbol{Y}}^{[s]},i}$, $\mu_{\bar{\boldsymbol{Y}}^{[s]},i}$ et $\Sigma_{\bar{\boldsymbol{Y}}^{[s]},i}$ par propagation à travers $\boldsymbol{f}$ et $\bar{\boldsymbol{f}}_{\boldsymbol{i}}$ en suivant la procédure détaillée dans la section 2.1.5.

Propagation analytique Il est également possible d'estimer les deux premiers moments de la sortie par propagation analytique. Pour les réseaux entièrement connectés et les CNN, nous fournissons la formule analytique permettant de propager les deux premiers moments d'une distribution gaussienne à travers la fonction d'activation ReLU, Leaky ReLU (Maas et al., 2013) et ELU (Clevert et al.,

2015) [2].

Proposition 2.1.9. *Soit X une variable aléatoire avec $X \sim \mathcal{N}_d(\boldsymbol{\mu}, \boldsymbol{\Sigma})$, $\boldsymbol{\mu} \in \mathbb{R}^d$, $\boldsymbol{\Sigma} \in \mathbb{S}_d^+(\mathbb{R})$ avec rang$(\boldsymbol{\Sigma}) = r$, $r \in \mathbb{N}^*$ et f étant la fonction linéaire rectifiée, alors pour la variable aléatoire $\boldsymbol{Y} = (Y_1, ..., Y_d) = f(\boldsymbol{X}) = (f(X_1), ..., f(X_d))$, pour tous k,k' dans $\{1...d\}$:*

$$\begin{aligned}
\mathbb{E}[Y_k] &= c_k\phi_k + \mu_k\Phi_k && (2.33)\\
\mathbb{E}[Y_k^2] &= (\mu_k^2 + c_k^2)\Phi_k + \mu_k c_k \phi_k && (2.34)\\
\mathbb{E}[Y_k Y_{k'}] &= c_k\alpha\phi_{k'}\phi_{NS} + (\mu_k\mu_{k'} + c_k\beta)(\Phi_{k'} - \Phi_{kk'}) \\
&+ \mu_k c_{k'}\phi_{k'}\Phi_S + \mu_{k'}c_k\phi_k\Phi_{NS} && (2.35)
\end{aligned}$$

La propagation analytique des deux premiers moments d'une distribution gaussienne au sein d'un réseau de neurone sous hypothèse de linéarité permet d'augmenter la précision de l'estimation (on ne recourt plus aux approximations induites par l'échantillonnage UT) mais conduit à une augmentation du temps de calcul.

Preuve de la Proposition 2.1.9. Soit $X \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ une variable aléatoire gaussienne dans $\mathbb{R}^n$ et $\boldsymbol{Y} = f(\boldsymbol{X})$ avec f la fonction d'activation ReLU. Par diagonalisation : $\exists Q \in \mathbb{O}_n(\mathbb{R}) \exists \Lambda \in \mathbb{D}_n(\mathbb{R}) : \boldsymbol{\Sigma} = Q\Lambda Q^T$, et en notant r le rang de $\boldsymbol{\Sigma}$, on a $\boldsymbol{\Sigma} = Q_r\Lambda_r Q_r^T$ où $Q_r \in \mathbb{R}^{n\times r}$ correspond aux r premières colonnes de Q.
On note la variable aléatoire $R = Q_r Z + \mu$ où $Z \sim \mathcal{N}(0, \Lambda_r)$ i.e. $R \sim \mathcal{N}(\mu, \Sigma)$. X et R sont égaux en termes de distribution.
On note f_i telle que :

$$\begin{aligned}
f_i \colon \mathbb{R}^n &\to \mathbb{R}_+ \\
x &\mapsto x_i \mathbb{1}_{x_i \geq 0}
\end{aligned}$$

Alors :

$$\begin{aligned}
\mathbb{E}[Y_i] &= \mathbb{E}_{z\sim p(Z)}\left[f_i(Q_r z + \mu)\right] \\
&= \mathbb{E}_{z\sim p(Z)}\left[(z^T q_i + \mu_i)\mathbb{1}_{(z^T q_i + \mu_i \geq 0)}\right] \\
&= \int_{-\infty}^{+\infty} (z^T q_i + \mu_i)\mathbb{1}_{(z^T q_i + \mu_i \geq 0)}\phi_{0,\Lambda_r}(z)dz && (2.36)
\end{aligned}$$

où ϕ_{0,Λ_r} est la fonction de densité de probabilité de la distribution gaussienne multivariée de moyenne zéro et de matrice de variance/covariance Λ_r.

2. Les formules pour Leaky ReLU et ELU ainsi que les détails des variables et les preuves sont fournis en annexe G.

Notons :

$$Z_i^+ = \left\{ z \in \mathbb{R}^r : z^T q_i + \mu_i \geq 0 \right\},$$

Donc :

$$\mathbb{E}[Y_i] = \int_{Z_i^+} z^T q_i \phi_{0,\Lambda_r}(z) dz + \mu_i \int_{Z_i^+} \phi_{0,\Lambda_r}(z) dz \tag{2.37}$$

Avec :

$$\int_{Z_i^+} z^T q_i \phi_{0,\Lambda_r}(z) dz = (2\pi)^{-\frac{r}{2}} |\Lambda_r|^{-\frac{1}{2}} \int_{Z_i^+} z^T q_i \exp\left(-\frac{1}{2} z^T \Lambda_r z \right) dz \tag{2.38}$$

On note $a = \sqrt{\Lambda_r^{-1}} z$, i.e. $z = \sqrt{\Lambda_r} a$; $|J| = |\Lambda_r|^{\frac{1}{2}}$ et

$$\bar{Z}_i^+ = \left\{ a \in \mathbb{R}^r : a^T \sqrt{\Lambda_r} q_i + \mu_i \geq 0 \right\}$$

On effectue une intégration par partie sur le demi-espace de $\mathbb{R}^r$:

$$\int_{Z_i^+} z^T q_i \phi_{0,\Lambda_r}(z) dz = (2\pi)^{-\frac{r}{2}} \int_{\bar{Z}_i^+} a^T \sqrt{\Lambda_r} q_i \exp\left(-\frac{1}{2} a^T a \right) da \tag{2.39}$$

En effectuant une rotation par rapport à l'hyperplan d'équation $a^T q_i = 0$, on trouve une matrice unitaire B telle que $B\sqrt{\Lambda_r} qi = \left\| \sqrt{\Lambda_r} q_i \right\| e_r$ où $e_r = [0, 0, ..., 1]^T \in \mathbb{R}^r$, alors

$$\tilde{Z}_i^+ = \left\{ b \in \mathbb{R}^r : b \left\| \sqrt{\Lambda_r} q_i \right\| e_r + \mu_i \geq 0 \right\} = \left\{ b \in \mathbb{R}^r : b_r \geq -\frac{\mu_i}{\left\| \sqrt{\Lambda_r} q_i \right\|} \right\} \tag{2.40}$$

$$\begin{aligned}
\int_{Z_i^+} z^T q_i \phi_{0,\Lambda_r}(z) dz &= (2\pi)^{-\frac{r}{2}} \int_{\tilde{Z}_i^+} b^T \left\| \sqrt{\Lambda_r} q_i \right\| e_r \exp\left(-\frac{1}{2} b^T b \right) db \\
&= (2\pi)^{-\frac{r}{2}} \int_{-\infty}^{+\infty} ... \int_{-\infty}^{+\infty} \int_{-\frac{\mu_i}{\left\| \sqrt{\Lambda_r} q_i \right\|}}^{+\infty} \left\| \sqrt{\Lambda_r} q_i \right\| b_r \exp\left(-\frac{1}{2} b^T b \right) db_r \\
&= \left\| \sqrt{\Lambda_r} q_i \right\| (2\pi)^{-\frac{1}{2}} \int_{-\frac{\mu_i}{\left\| \sqrt{\Lambda_r} q_i \right\|}}^{+\infty} b_r \exp\left(-\frac{1}{2} b_r^2 \right) db_r \\
&= \left\| \sqrt{\Lambda_r} q_i \right\| \phi_{0,1}\left(\frac{\mu_i}{\left\| \sqrt{\Lambda_r} q_i \right\|} \right)
\end{aligned} \tag{2.41}$$

De la même manière :

$$\int_{Z_i^+} \phi_{0,\Lambda_r}(z) dz = \int_{-\frac{\mu_i}{\left\| \sqrt{\Lambda_r} q_i \right\|}}^{+\infty} \phi_{0,1}(b_r) db_r = \Phi_{0,1}\left(\frac{\mu_i}{\left\| \sqrt{\Lambda_r} q_i \right\|} \right) \tag{2.42}$$

Alors, en notant $c_i = \left\|\sqrt{\Lambda_r} q_i\right\|$, on a :

$$\boxed{\mathbb{E}[Y_i] = c_i \phi_{0,1}\left(\frac{\mu_i}{c_i}\right) + \mu_i \Phi_{0,1}\left(\frac{\mu_i}{c_i}\right)} \tag{2.43}$$

Estimation des variances / covariances Soit f_{ij} la fonction définie par :

$$\begin{aligned} f_{ij} \colon \mathbb{R}^n &\to \mathbb{R}_+ \\ x &\mapsto x_i \mathbb{1}_{x_i \geq 0} x_j \mathbb{1}_{x_j \geq 0} \end{aligned}$$

$$\begin{aligned} \mathbb{E}[Y_i Y_j] &= \mathbb{E}_{z \sim p(Z)}\left[f_{ij}(Q_r z + \mu)\right] \\ &= \mathbb{E}_{z \sim p(Z)}\left[(z^T q_i + \mu_i)\mathbb{1}_{(z^T q_i + \mu_i \geq 0)}(z^T q_j + \mu_j)\mathbb{1}_{(z^T q_j + \mu_j \geq 0)}\right] \\ &= \int_{-\infty}^{+\infty} (z^T q_i + \mu_i)\mathbb{1}_{(z^T q_i + \mu_i \geq 0)}(z^T q_j + \mu_j)\mathbb{1}_{(z^T q_j + \mu_j \geq 0)} \phi_{0,\Lambda_r}(z) dz \end{aligned} \tag{2.44}$$

Notons :

$$Z_{ij}^+ = \left\{ z \in \mathbb{R}^r : z^T q_i + \mu_i \geq 0,\ z^T q_j + \mu_j \geq 0 \right\}$$

$$\begin{aligned} \mathbb{E}[Y_i Y_j] &= \int_{Z_{ij}^+} z^T q_i z^T q_j \phi_{0,\Lambda_r}(z) dz + \mu_i \int_{Z_{ij}^+} z^T q_j \phi_{0,\Lambda_r}(z) dz \\ &+ \mu_j \int_{Z_{ij}^+} z^T q_i \phi_{0,\Lambda_r}(z) dz + \mu_i \mu_j \int_{Z_{ij}^+} \phi_{0,\Lambda_r}(z) dz \end{aligned} \tag{2.45}$$

$$\mathbb{E}[Y_i Y_j] = (*) + \mu_i (**)_j + \mu_j (**)_i + \mu_i \mu_j (***) \tag{2.46}$$

Cas où i=j : (Estimation des variances)

$$\mathbb{E}[Y_i^2] = \int_{Z_i^+} (z^T q_i)^2 \phi_{0,\Lambda_r}(z) dz + 2\mu_i \int_{Z_i^+} z^T q_i \phi_{0,\Lambda_r}(z) dz + \mu_i^2 P(Z \in Z_i^+) \tag{2.47}$$

avec :

$$\int_{Z_i^+} (z^T q_i)^2 \phi_{0,\Lambda_r}(z) dz = c_i^2 (2\pi)^{-\frac{1}{2}} \int_{-\frac{\mu_i}{c_i}}^{+\infty} b_r^2 \exp\left(-\frac{1}{2} b_r^2\right) db_r \tag{2.48}$$

Par intégration par partie, on obtient :

$$\boxed{\mathbb{E}[Y_i^2] = (\mu_i^2 + c_i^2)\Phi_{0,1}\left(\frac{\mu_i}{c_i}\right) + \mu_i c_i \phi_{0,1}\left(\frac{\mu_i}{c_i}\right)} \tag{2.49}$$

Cas général où $i \neq$ j : (Estimation des covariances) On doit effectuer une intégration par partie sur le quart d'espace de $\mathbb{R}^r$. En notant comme précédemment $a = \sqrt{\Lambda_r^{-1}}z$, on a :

$$Z_{ij}^+ = \left\{a \in \mathbb{R}^r : a^T\sqrt{\Lambda_r}q_i + \mu_i \geq 0,\ a^T\sqrt{\Lambda_r}q_j + \mu_j \geq 0\right\}$$

Il est utile de noter que $Z_{ij}^+ = Z_i^+ \cap Z_j^+$.

Une nouvelle fois, nous pouvons trouver une matrice unitaire B telle que :

$$\begin{cases} B\sqrt{\Lambda_r}q_i = c_i e_r \\ B\sqrt{\Lambda_r}q_j = \alpha e_{r-1} + \beta e_r \end{cases} \tag{2.50}$$

On suppose maintenant que $\alpha \neq 0$. Sinon, si $\alpha = 0$, alors $\beta = c_j$ et

$$\tilde{Z}_{ij}^+ = \left\{b \in \mathbb{R}^r : b_r \geq m = \max\left(-\frac{\mu_i}{c_i}, -\frac{\mu_j}{c_j}\right)\right\},$$

ce qui correspond au cas précédent de l'intégration sur le demi-espace de R^r :

$$\boxed{\mathbb{E}[Y_i Y_j] = (\mu_i\mu_j + c_i c_j)\Phi_{0,1}(-m) + (\mu_i c_j + \mu_j c_i + m c_i c_j)\phi_{0,1}(m)} \tag{2.51}$$

On peut trouver une matrice B en choisissant r-2 vecteurs orthonormaux dans $Z_i^+ \cap Z_j^+$, un vecteur dans Z_i^+ et un vecteur orthogonale pour compléter la base (par Gram Schmidt par exemple).

De plus, les matrices unitaires préservent la norme L_2 par multiplication :

$$\begin{cases} \alpha^2 + \beta^2 = c_j^2 \\ \left\|B(\sqrt{\Lambda_r}q_i - \sqrt{\Lambda_r}q_j)\right\| = \left\|\sqrt{\Lambda_r}q_i - \sqrt{\Lambda_r}q_j\right\| \end{cases}$$

Donc :

$$\begin{cases} \beta = \frac{c_i^2 + c_j^2 - c_{i-j}^2}{2c_i} \\ \alpha = \pm\sqrt{c_j^2 - \beta^2} \end{cases}$$

où :

$$\begin{cases} c_i = \left\| \sqrt{\Lambda_r} q_i \right\| \\ c_j = \left\| \sqrt{\Lambda_r} q_j \right\| \\ c_{i-j} = \left\| \sqrt{\Lambda_r} q_i - \sqrt{\Lambda_r} q_j \right\| \end{cases}$$

Sans perte de généralité, on peut choisir $\alpha = \sqrt{c_j^2 - \beta^2}$. On obtient :

$$\begin{aligned} \tilde{Z}_{ij}^{+} &= \left\{ b \in \mathbb{R}^r : b_r \geq -\frac{\mu_i}{ci} \ , \ \alpha b_{r-1} + \beta b_r \geq -\mu_j \right\} \\ &= \left\{ b \in \mathbb{R}^r : b_r \geq -\frac{\mu_i}{ci} \ , \ b_{r-1} \geq -\frac{(\mu_j + \beta b_r)}{\alpha} \right\} \end{aligned} \tag{2.52}$$

En utilisant des intégrations par parties, des changements de variables classiques ainsi que les équations 10,010.1 et 10,010.8 de Owen (1980), on obtient :

$$(***) = \Phi_{0,1}\left(\frac{\mu_j}{c_j}\right) - \Phi_{\begin{bmatrix}0\\0\end{bmatrix},\begin{bmatrix}1 & -\frac{\beta}{c_j}\\ -\frac{\beta}{c_j} & 1\end{bmatrix}}\left(\left[\frac{\mu_j}{c_j}, -\frac{\mu_i}{c_i}\right]^T\right) \tag{2.53}$$

$$(**)_i = c_i \phi_{0,1}\left(\frac{\mu_i}{c_i}\right)\Phi_{0,1}\left(\frac{\mu_j - \beta\frac{\mu_i}{c_i}}{\alpha}\right) + \frac{\beta c_i}{c_j}\phi_{0,1}\left(\frac{\mu_j}{c_j}\right)\Phi_{0,1}\left(\left(\frac{c_j\mu_i}{c_i\alpha} - \frac{\beta\mu_j}{\alpha c_j}\right)\right) \tag{2.54}$$

$$(**)_j = c_j \phi_{0,1}\left(\frac{\mu_j}{c_j}\right)\Phi_{0,1}\left(\left(\frac{c_j\mu_i}{c_i\alpha} - \frac{\beta\mu_j}{\alpha c_j}\right)\right) + \beta\phi_{0,1}\left(\frac{\mu_i}{c_i}\right)\Phi_{0,1}\left(\frac{\mu_j}{\alpha} - \frac{\mu_i\beta}{c_i\alpha}\right) \tag{2.55}$$

$$\begin{aligned} (*) &= c_i\alpha\phi_{0,1}\left(\frac{\mu_j}{c_j}\right)\phi_{0,1}\left(\frac{\beta\mu_j}{\alpha c_j} - \frac{\mu_i c_j}{c_i\alpha}\right) - \beta\mu_i\phi_{0,1}\left(\frac{\mu_i}{c_i}\right)\Phi_{0,1}\left(\frac{\mu_j}{\alpha} - \frac{\beta\mu_i}{\alpha c_i}\right) \\ &+ c_i\beta(***) - \phi_{0,1}\left(\frac{\mu_j}{c_j}\right)\frac{c_i\beta\mu_j}{c_j}\Phi_{0,1}\left(\left(\frac{c_j\mu_i}{c_i\alpha} - \frac{\beta\mu_j}{\alpha c_j}\right)\right) \end{aligned} \tag{2.56}$$

En posant :

$$\phi_j = \phi_{0,1}\left(\frac{\mu_j}{c_j}\right), \phi_i = \phi_{0,1}\left(\frac{\mu_i}{c_i}\right), \Phi_j = \Phi_{0,1}\left(\frac{\mu_j}{c_j}\right), \tag{2.57}$$

$$\Phi_{ij} = \Phi_{\begin{bmatrix}0\\0\end{bmatrix},\begin{bmatrix}1 & -\frac{\beta}{c_j}\\ -\frac{\beta}{c_j} & 1\end{bmatrix}}\left(\left[\frac{\mu_j}{c_j}, -\frac{\mu_i}{c_i}\right]^T\right), \Phi_{NS} = \Phi_{0,1}\left(\frac{\mu_j}{\alpha} - \frac{\beta\mu_i}{\alpha c_i}\right), \tag{2.58}$$

$$\Phi_S = \Phi_{0,1}\left(\left(\frac{c_j\mu_i}{c_i\alpha} - \frac{\beta\mu_j}{\alpha c_j}\right)\right), \phi_{NS} = \phi_{0,1}\left(\frac{\beta\mu_j}{\alpha c_j} - \frac{\mu_i c_j}{c_i\alpha}\right), \tag{2.59}$$

On obtient finalement le résultat recherché :

$$\boxed{\mathbb{E}[Y_i Y_j] = c_i \alpha \phi_j \phi_{NS} + (\mu_i \mu_j + c_i \beta)(\Phi_j - \Phi_{ij}) + \mu_i c_j \phi_j \Phi_S + \mu_j c_i \phi_i \Phi_{NS}} \tag{2.60}$$

□

Remarque 10.

L'estimation d'une fonction cumulative de distribution en 2 dimensions peut être coûteux en temps de calcul. Pour y remédier, de très bonnes approximations existent (Scott (2015)) et ne font intervenir que des fonctions usuelles telles que l'exponentielle, la racine carrée ou encore la fonction d'erreur.

Comment obtenir $\bar{\boldsymbol{f}}_{\boldsymbol{i}}$:

Formellement, un réseau de neurones peut être considéré comme une fonction paramétrique non linéaire $\boldsymbol{f}(\cdot, \boldsymbol{\theta})$, résultat de la concaténation de L blocs élémentaires h_i, chacun composé d'une transformation linéaire et d'une fonction d'activation non linéaire : $\mathbf{Y} = h_{L-1}\left(..h_1\left(h_0\left(\mathbf{X}, \theta_0\right)\right)\right) = \boldsymbol{f}(\mathbf{X}, \boldsymbol{\theta})$.

Pour obtenir $\bar{\boldsymbol{f}}_{\boldsymbol{i}}$, nous effectuons une succession de linéarisations de Taylor au premier ordre : $\bar{\boldsymbol{Y}} = \bar{h}_{L-1}\left(...\bar{h}_1\left(\bar{h}_0\left(\mathbf{X}, \theta_0\right)\right)\right) = \bar{\boldsymbol{f}}(\mathbf{X}, \boldsymbol{\theta})$, en utilisant les $2d+1$ points sigma $Z_{i,[j]}$ obtenus dans la section 2.1.5. En notant $Y_{i,[j],l} = h_l(h_{l-1}(...h_0(Z_{i,[j]}, \theta_0)))$ et $\mu_{i,j,l} = \sum_{j=1}^{2d+1} w_i Y_{i,[j],l}$, et $Y_{i,[j],0} = Z_{i,[j]}$ alors pour la couche l, nous obtenons $\bar{h}_l$ par linéarisation de Taylor du premier ordre :

$$\bar{h}_l(X) = h_l(\mu_{i,j,l-1}) + J_{h_l}(\mu_{i,j,l-1})(X - \mu_{i,j,l-1}) \tag{2.61}$$

où $J_{h_l}(\mu_{i,j,l-1})$ désigne la matrice jacobienne de h_l en $\mu_{i,j,l-1}$ [3]. Pour finir :

3. La matrice jacobienne n'est pas définie partout pour les réseaux de neurones utilisant ReLU ou Leaky ReLU, elle est calculée en pratique à l'aide de la bibliothèque de calcul classique (**?**)

$$\mu_{\bar{\boldsymbol{Y}}^{[s]},i} \approx \sum_{j=1}^{2d+1} \boldsymbol{\omega}_j \bar{\boldsymbol{f}}_{\boldsymbol{i}}(Z_{i,[j]}), \qquad \mu_{\tilde{\boldsymbol{Y}}^{[s]},i} \approx \sum_{j=1}^{2d+1} \boldsymbol{\omega}_j \boldsymbol{f}(Z_{i,[j]})$$

$$\Sigma_{\bar{\boldsymbol{Y}}^{[s]},i} \approx \sum_{j=1}^{2d+1} \boldsymbol{\omega}_j (\bar{\boldsymbol{f}}_{\boldsymbol{i}}(Z_{i,[j]}) - \mu_{\bar{\boldsymbol{Y}}^{[s]},i})(\bar{\boldsymbol{f}}_{\boldsymbol{i}}(Z_{i,[j]}) - \mu_{\bar{\boldsymbol{Y}}^{[s]},i})^T$$

$$\Sigma_{\tilde{\boldsymbol{Y}}^{[s]},i} \approx \sum_{j=1}^{2d+1} \boldsymbol{\omega}_j (\boldsymbol{f}(Z_{i,[j]}) - \mu_{\tilde{\boldsymbol{Y}}^{[s]},i})(\boldsymbol{f}(Z_{i,[j]}) - \mu_{\tilde{\boldsymbol{Y}}^{[s]},i})^T$$

$$\mathbb{E}_{X \sim P_{\tilde{\boldsymbol{X}}^{[s]},i}} \left[\|\boldsymbol{f}(X) - \bar{\boldsymbol{f}}_{\boldsymbol{i}}(X)\|_2^2 \right] \approx \sum_{j=1}^{2d+1} \omega_j \|\boldsymbol{f}(Z_{i,[j]}) - \bar{\boldsymbol{f}}_{\boldsymbol{i}}(Z_{i,[j]})\|_2^2$$

Haute dimensionalité

Comme la méthodologie proposée considère le réseau de neurones comme une boîte noire, le goulot d'étranglement en termes d'utilisation de la mémoire et de temps de calcul réside dans les dimensions d'entrée et de sortie (indépendamment de la largeur du réseau), qui peuvent être élevées. Ce problème se pose avant tout dans les architectures de traitement d'images et de réseaux de neurones convolutifs. Plus précisément, notre méthode repose sur l'inverse de la matrice de covariance d'entrée $\Sigma_{\boldsymbol{X}}$ présentant une complexité de calcul en $\mathcal{O}(d^3)$ (où d est la dimension de $\boldsymbol{X}$). Pour améliorer la scalabilité de la méthode proposée, nous appliquons la méthodologie au sous-espace actif de $\Sigma_{\boldsymbol{X}}$ (Constantine et al., 2014). La sélection du sous-espace actif consiste à réduire la dimension du problème en conservant les r ($r \ll d$) premières valeurs propres les plus importantes de $\Sigma_{\boldsymbol{X}}$, à l'aide d'algorithmes tels que la méthode de Lanczos redémarrée implicitement (Calvetti et al., 1994). Pour améliorer encore les performances, nous pouvons appliquer une étape de *rodage* consistant à effectuer initialement s_0 itérations de division, où s_0 est fixé manuellement.

Algorithme

Ci-dessous, nous présentons la version complète de WGMprop sous forme algorithmique 1 :

Hyperparamètres

WGMprop nécessite le réglage de plusieurs hyperparamètres. Ainsi, le seuil de division est un hyperparamètre crucial pour cette méthode. Actuellement, cet hyperparamètre doit être réglé manuellement mais pourrait être automatisé en mettant notamment en place un système de gain basé sur le calcul de la borne supérieure présentée en 2.1.3. Néanmoins, grâce à l'expérimentation, nous avons facilement pu identifier une valeur de seuil (dans ce cas, $0,0001$) qui a bien fonctionné pour toutes nos expériences. De plus, nous recommandons un niveau de division à 7 composantes car

Algorithm 1 WGMprop

Input:

— Distribution d'entrée gaussienne, $P_{\boldsymbol{X}}$
— Le seuil de division, T_{split}
— Le nombre de pré-division, s_0
— Dimension maximale du sous-espace actif, r
— Le niveau de division, n

Output:

— Le MG de sortie estimé $P_{\tilde{\boldsymbol{Y}}^{[s]}} = (\omega_{\hat{\boldsymbol{Y}}^{[s]},i}, \mu_{\hat{\boldsymbol{Y}}^{[s]},i}, \Sigma_{\hat{\boldsymbol{Y}}^{[s]},i})$

1: **Sélection du sous-espace actif :** $\Sigma_{\boldsymbol{X}} = U_r \Lambda_r U_r^T$
2: **Rodage :** Effectuer s_0 divisions $S^{[s_0]}(P_{\boldsymbol{X}}; n) = P_{\hat{\boldsymbol{X}}^{[s_0]}} = (\omega_{\hat{\boldsymbol{X}}^{[s_0]},i}, \mu_{\hat{\boldsymbol{X}}^{[s_0]},i}, \Sigma_{\hat{\boldsymbol{X}}^{[s_0]},i})$ (Eq. 2.31)
3: **Split&Merge :** Pour chaque composante $P_{\hat{\boldsymbol{X}}^{[s_0]},i}$ de $P_{\hat{\boldsymbol{X}}^{[s_0]}}$:
4: **Échantillonnage UT :** Générer $Z_{i,[j]}$ et ω_j (Sec. 2.1.5)
5: **Propagation :** Estimer $P_{\tilde{\boldsymbol{Y}}^{[s]},i}$ et $P_{\tilde{\boldsymbol{Y}}^{[s]},i}$ (Sec. 2.1.5)
6: **Critère de division :** Calculée B_s en utilisant l'équation (2.8)
7: **Division :** Si $B_s > T_{split}$ alors $P_{\hat{\boldsymbol{X}}^{[s]},i}$ (Éq. 2.31)
8: **Itération :** Appliquer les étapes (4)-(7) sur les composantes divisées de $P_{\hat{\boldsymbol{X}}^{[s]},i}$ jusqu'à $B_s < T_{split}$
9: **Estimation de sortie :** Estimer $(\mu_{\hat{\boldsymbol{Y}}^{[s]},i}, \Sigma_{\hat{\boldsymbol{Y}}^{[s]},i})_{1 \leq i \leq M}$ par appariement des moments

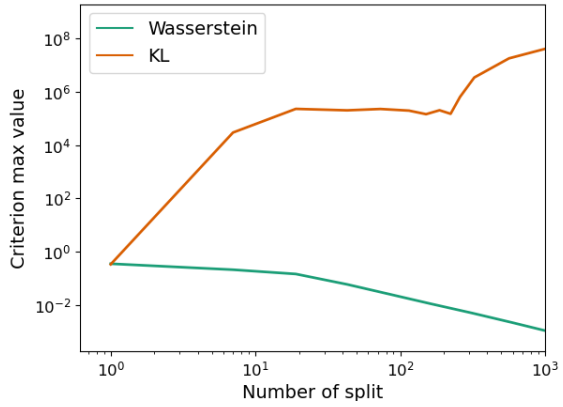
présente la plus faible erreur type $\epsilon_{0,1}$, comme le montre la table 2.1. Enfin, le nombre de pré-divisions dépend de l'application et de l'intensité du bruit. Une valeur comprise entre 0 et 3 semble raisonnable.

2.2 Expériences

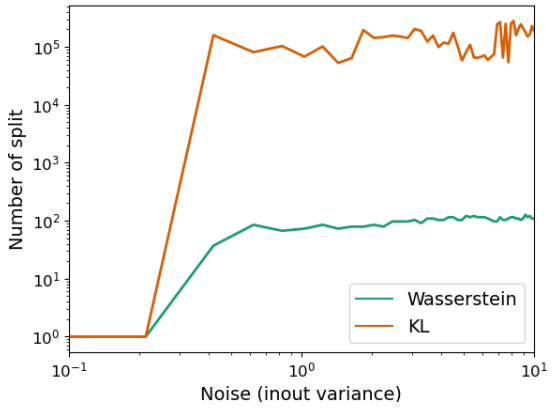
Nous présentons tout d'abord une courte expérience de propagation à travers une unique fonction d'activation ReLU afin de comparer la détection de cette non linéarité par un critère de Wasserstein et un critère basé sur la divergence de Kullback-Leibler. Puis, nous détaillerons une expérience réalisée sur l'ensemble de données MNIST, où la dimension d'entrée permet une comparaison approfondie avec une référence de Monte Carlo (i.e. sans s'appuyer sur une technique de réduction de dimension). Puis, nous présentons des expériences menées sur des bases de données plus complexes telles que CIFAR-10, CIFAR-100 et Camelyon, offrant un large éventail de scénarios s'approchant de cas d'applications réels, démontrant ainsi le caractère générique de notre méthodologie. Enfin, nous présenterons une courte expérience permettant de comparer les différentes stratégies de sélection de la direction de division. L'annexe E présente également une comparaison de différentes variantes de notre algorithme.

2.2.1 Comparaison des critères de division pour une unique activation (ReLU).

Pour illustrer les limites théoriques exposées dans l'introduction (complétée par l'annexe C), nous menons une expérience consistant à propager une distribution gaussienne univariée à travers une fonction d'activation ReLU en utilisant le critère basé sur la divergence KL (avec approximation Leaky ReLU de pente 10^{-3}). Cette mini-expérience est primordiale puisqu'elle constitue la brique élémentaire d'une propagation effectuée couche par couche, comme lors de l'utilisation de la divergence KL.



(a) Évolution du critère de division à chaque étape de la procédure itérative.



(b) Nombre de divisions nécessaires pour propager $\mathcal{N}(1, 0, \sigma^2)$ à travers une ReLU

Tout d'abord, la figure 2.4a montre l'évolution de la valeur maximale du critère à chaque étape de la procédure itérative. Le critère de Wasserstein que nous proposons diminue au fur et à mesure que le nombre de divisions augmente jusqu'à tomber sous le seuil spécifié, mettant fin à la procédure. Au contraire, le critère KL devient très important et la procédure ne se termine donc pas. Enfin, la figure 2.4b montre l'évolution du nombre de divisions nécessaires lors de la propagation d'une gaussienne $\mathcal{N}(1.0, \sigma^2)$ à travers une ReLU. On observe que cela conduit à un très grand nombre de divisions, ce qui rend la propagation d'une distribution gaussienne à travers la première couche d'un réseau ReLU déjà impraticable.

2.2.2 Application de WGMprop à la base de données MNIST

La tâche de classification MNIST consiste en la classification des vignettes, de taille $28 \times 28 \times 1$ soit 784 pixels, en accord avec le chiffre contenu au sein de l'image. La figure 2.5 présente des exemples de ce jeu de données pour chacune des 10 classes.

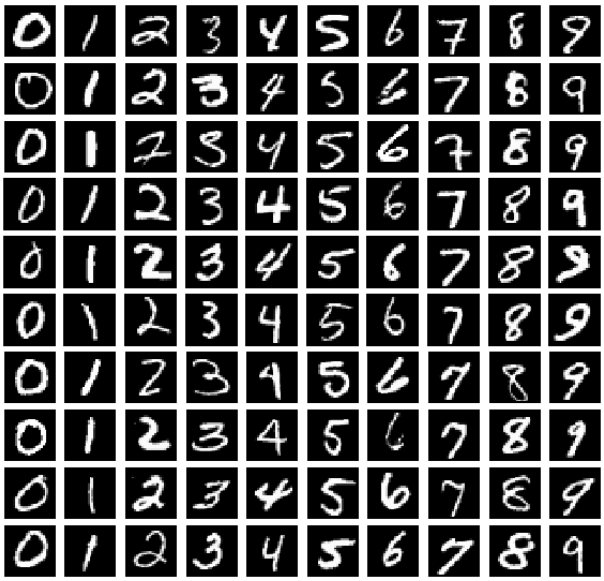


FIGURE 2.5 – Exemples d'images du jeu de données MNIST pour chaque classe.

Paramètres expérimentaux

Dans cette section, notre méthode est testée dans le cadre d'une tâche de régression MNIST en examinant la loi des logits lorsque les images d'entrée sont corrompues par différents types de bruit (bruit gaussien, flou et bruit de contraste) à différents niveaux d'intensité (I_1, I_2, I_3 [4] se référant à une intensité faible, moyenne et élevée, respectivement). Ces bruits ont été sélectionnés afin de couvrir une large gamme de scénarios en termes d'amplitude de covariance. En effet, alors que le bruit gaussien corrompt les pixels indépendamment (faibles covariances), le flou corrompt chaque pixel en fonction de son voisinage (covariances moyennes) et le bruit de contraste affecte l'ensemble de l'image (fortes covariances). Le réseau de neurones utilisé est un perceptron à 4 couches contenant 200 neurones chacune. En pratique, nous générons 10^4 échantillons bruités, dont nous extrayons un vecteur moyen ainsi que la matrice de variance et de covariance associée. Ensuite, nous propageons la distribution normale résultante (par appariement des moments) à travers le réseau en comparant notre méthode (WGMprop) à la méthode LPN (Gast and Roth, 2018) et à une propagation utilisant l'échantillonnage UT à travers le réseau complet (Abdelaziz et al., 2015), référencé comme UT@FN dans la suite. Nous n'avons pas été en mesure d'intégrer dans nos expériences les travaux de (Zhang and Shin, 2021) pour les raisons présentées précédemment (ainsi qu'au sein de l'annexe C). La figure 2.6 présente un échantillon d'entrée corrompu par différents types de bruits et niveaux d'intensité. La référence MC est construite sur 10^6 d'échantillons générés pour toutes les expériences.

4. **Bruit gaussien** : $\mu = (0, 0, 0)$, $\sigma^2 = (0.002, 0.01, 0.05)$; **Flou** : $\mu = (0, 0, 0)$, $\sigma^2 = (1, 16, 49)$; **Bruit de contraste** : $\mu = (1, 1, 1)$, $\sigma^2 = (0.04, 1, 25)$. (μ, σ) correspondent aux paramètres définis pour I_1, I_2 et I_3.

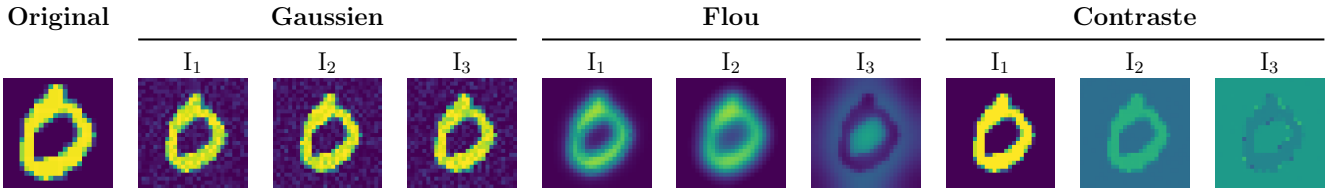


FIGURE 2.6 – Image d'entrée corrompue par différents types de bruit (bruit gaussien, bruit de flou et de contraste) à différents niveaux d'intensité (I_1, I_2, I_3).

Résultats : comparaison avec l'état de l'art

Une évaluation quantitative des performances a été réalisée sur 500 images test de l'ensemble de données MNIST, en comparant leurs performances à la référence MC, qui montre un temps de prédiction moyen de 15,91 secondes ($\sim 6,5$s pour la génération de l'échantillon et $\sim$ 9s pour la propagation). L'analyse repose sur le calcul des mesures de performance : Erreur absolue moyenne en pourcentage, MAPE, de la moyenne estimée ($\mathbf{MAPE}_{\text{MEAN}}$) et de l'écart-type estimé ($\mathbf{MAPE}_{\text{STD}}$), distance de 2-Wasserstein (2W), Intersection sur l'Union de l'intervalle de prédiction à 95% (IOU_{95}) sur la marginale principale[5] et le temps de prédiction moyen (TIME). Le tableau 2.2 présente les résultats pour l'intensité de bruit la plus élevée, cas expérimentale le plus difficile (nécessitant un plus grand nombre de divisions).

TABLE 2.2 – Comparaison numérique des performances des méthodes de l'état de l'art et de WGMprop sur les trois bruits étudiés (Gaussien, Flou et Contraste) à l'intensité I_3. Écart-type en ().

			PERFORMANCE CRITERIA				
Bruit	**Méthode**	**# Gaussienne (↓)**	**2W ($\times 10^3$) (↓)**	**$\text{MAPE}_{\text{MEAN}}$ (↓)**	**MAPE_{STD} (↓)**	**IOU_{95} (↑)**	**TIME (s) (↓)**
Gaussien	LPN	1	2.39(3.11)	4.26(4.82)	197.08(411.19)	0.342(0.156)	0.02(0.00)
	UT	1	0.03(0.05)	0.21(0.26)	6.07(7.98)	0.859(0.117)	0.06(0.01)
	Zhang et al.	10000∗	-/-	-/-	-/-	-/-	-/-
	WGMprop	349	0.02(0.04)	0.16(0.20)	5.69(8.44)	0.876(0.113)	0.48(0.04)
Flou	LPN	1	39.77(19.38)	13.61(9.48)	88.52(3.43)	0.145(0.037)	0.02(0.00)
	UT	1	17.44(12.76)	17.00(10.89)	30.74(17.03)	0.604(0.101)	0.05(0.01)
	Zhang et al.	10000∗	-/-	-/-	-/-	-/-	-/-
	WGMprop	635	0.26(0.21)	0.17(0.15)	0.48(0.51)	0.920(0.034)	0.58(0.03)
Contraste	LPN	1	88.15(33.15)	53.66(23.12)	66.55(10.03)	0.343(0.099)	0.02(0.00)
	UT	1	95.85(45.12)	66.88(25.57)	44.18(26.74)	0.446(0.162)	0.04(0.01)
	Zhang et al.	10000∗	-/-	-/-	-/-	-/-	-/-
	WGMprop	691	0.25(0.23)	0.18(0.16)	0.26(0.29)	0.938(0.041)	0.81(0.10)

*Le nombre de divisions a atteint la limite mémoire dès la première ReLU.

5. la marginale principale est définie comme la marginale présentant la moyenne la plus élevée.

D'un point de vue général, les méthodes LPN et UT@FN sont plus rapides que WGMprop au détriment de la précision. En effet, quelle que soit la complexité du bruit d'entrée, le temps de prédiction moyen de ces méthodes reste constant (≈ 0.02s) alors que leurs erreurs de prédiction augmentent considérablement (l'IOU_{95} passe de 0.859 pour un bruit gaussien et une propagation UT@FN à 0.446 pour un bruit de contraste). En effet, dans le cas d'une entrée corrompue par un bruit gaussien, la PDF de sortie reste proche d'une gaussienne (valeur de 2-Wasserstein de $0,03 \times 10^{-5}$ pour la méthode UT@FN) alors que, pour des bruits plus complexes, l'hypothèse gaussienne sur la PDF de sortie ne tient plus.

Pour illustrer ce phénomène, la figure 2.7 affiche à la fois les PDF de la marginale principale de sortie estimées et la PDF de référence (MC) sur l'image n°66 (de l'ensemble de données de test) corrompue par l'ensemble des bruits à l'intensité la plus élevée (I_3). Dans le cas de bruits de flou et de contraste, les méthodes UT@FN et LPN souffrent d'une mauvaise représentation de la forme en raison de leur hypothèse gaussienne. La mauvaise représentation de la PDF de sortie dans le cas d'un bruit gaussien par la méthode LPN semble suggérer le fait que négliger les covariances dans la propagation couche par couche entraîne une faible précision d'estimation des statistiques de la distribution de sortie. On note également que WGMprop capture bien la forme de la PDF de sortie. C'est donc un bon candidat pour obtenir une estimation fiable de l'intervalle de prédiction à 95% tout en maintenant un temps d'exécution raisonnable (un maximum de $0,81$s pour le bruit de contraste).

Bien que le temps de prédiction moyen augmente légèrement avec la complexité du bruit d'entrée (plus de composantes gaussiennes dans le mélange sont nécessaires), les prédictions de la méthode WGMprop restent très précises : $\mathbf{IOU}_{95}$ supérieur à 0,876, $\mathbf{MAPE}_{\text{MEAN}}$ inférieur à 0,18%. En outre, la figure 2.8 présente un diagramme en boîte présentent les distributions MAPE sur les différents centiles (1, $2,5$, 25, 50, $97,5$ et 99) de la PDF de sortie pour les trois méthodes comparées pour un bruit de contraste d'intensité élevée. Comme attendu, les méthodes fondées sur l'hypothèse gaussienne présentent des erreurs élevées sur l'ensemble considéré. L'utilisation d'une mixture de gaussiennes permet une représentation fidèle de la PDF de sortie et donc une estimation fiable des différents centiles. Nous constatons une erreur plus importante pour les centiles inférieurs. Pour les percentiles plus élevés, notre approche présente de faibles valeurs MAPE ainsi qu'une faible dispersion de ces erreurs.

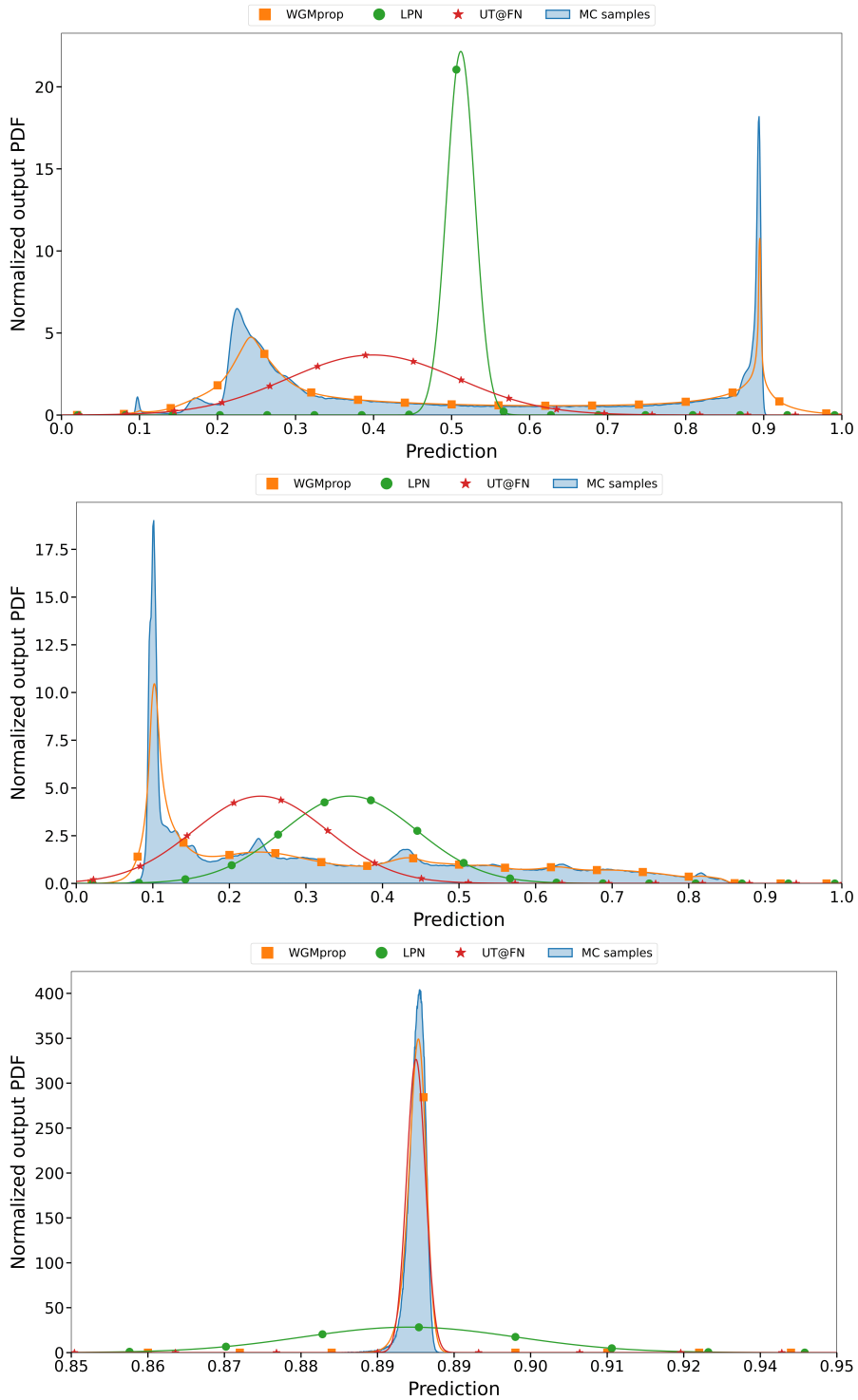


FIGURE 2.7 – PDF de sortie estimées (marginale principale) pour l'échantillon n°66 corrompu par un bruit de flou (**Haut**), un bruit de contraste (**Milieu**) et un bruit additif gaussien (**Bas**) (intensité I_3). La PDF de sortie estimée à l'aide de la propagation de Monte Carlo (référence) est indiquée en bleu.

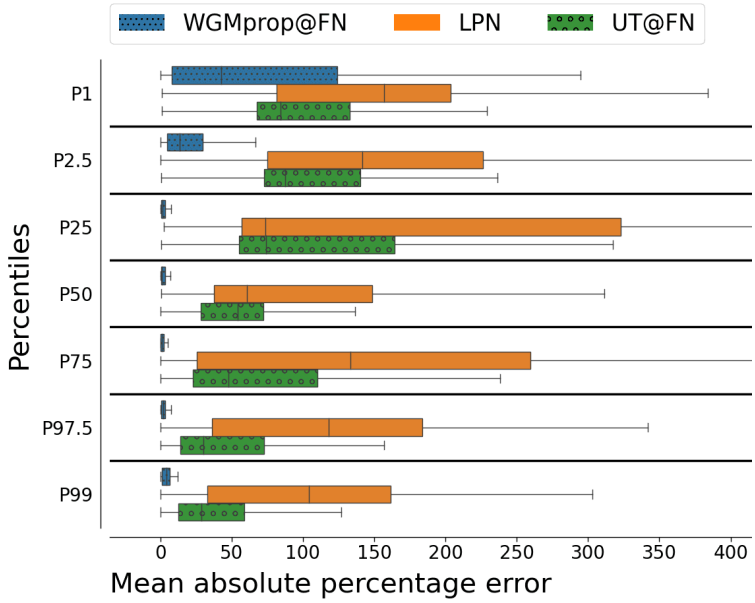


FIGURE 2.8 – Graphiques MAPE pour les percentiles 1, 2,5, 25, 50, 75, 97,5 et 99. Les images d'entrée ont été dégradées par un bruit de contraste (I_3).

2.2.3 Application de WGMprop aux bases de données CIFAR-10, CIFAR-100 et CAMELYON

Dans cette sous-section, nous testons notre méthodologie sur des ensembles de données et des architectures de réseau plus complexes, détaillés dans le tableau 2.3.

La tâche de classification CIFAR-10 consiste en la classification des vignettes, de taille $32 \times 32 \times 3$ pixels, en accord avec l'animal contenu au sein de l'image. La figure 2.9 présente des exemples de ce jeu de données pour chacune des 10 classes. La base de données CIFAR-100 est analogue à la base de données CIFAR-10 mais présente 100 classes différentes.

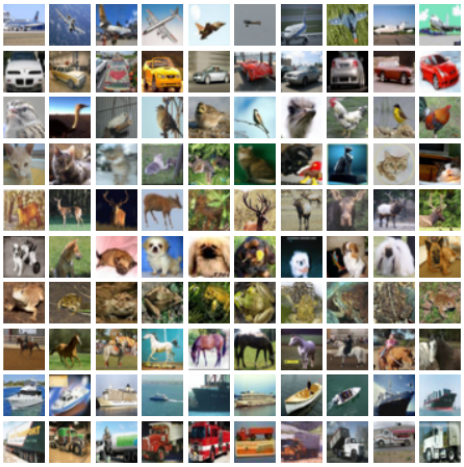


FIGURE 2.9 – Exemples d'images du jeu de données CIFAR-10 pour chaque classe.

La base de données CAMELYON, quant à elle, consiste en la classification binaire d'images de cellules de ganglion lymphatique afin de détecter la présence de cellules cancéreuses. La figure 2.10 présente des exemples de ce jeu de données.

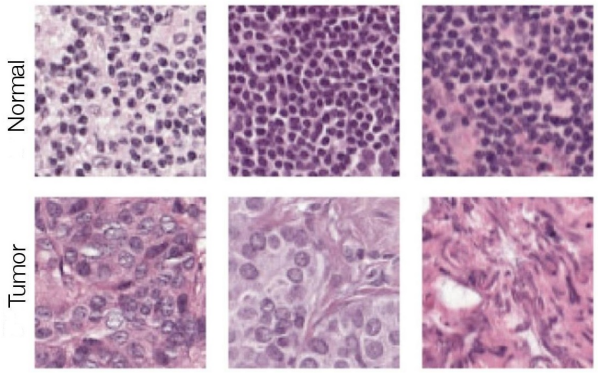


FIGURE 2.10 – Exemples d'images du jeu de données Camelyon pour chaque classe.

Paramètres expérimentaux

La table 2.3 présente les caractéristiques des expériences présentées dans cette sous-section.

TABLE 2.3 – Détails expérimentaux.

Données	**Architecture**	**Dimension d'entrée**	**Dimension de sortie**	**Couche de sortie**	**Nb paramètres**	s_0
CIFAR-10	$VGG8$	$32 \times 32 \times 3$	10	Linear	$2,397,216$	3
CIFAR-100	$VGG13$	$32 \times 32 \times 3$	100	Softmax	$10,171,936$	1
Camelyon	$VGG10$	$96 \times 96 \times 3$	2	Sigmoïd	$10,615,328$	1

Cette série d'expériences nous permet d'explorer différents scénarios en variant les dimensions d'entrée et de sortie, le nombre de paramètres dans le réseau et le type de dernière couche utilisée dans le réseau (couche linéaire pour la régression, softmax pour la classification multiclasse et sigmoïde pour la classification binaire). Pour ces expériences, nous conservons le tirage de Monte Carlo comme référence et mettons en œuvre UT@FN en plus de notre méthode (WGMprop). Dans la section suivante, nous présentons les résultats obtenus avec le bruit de flou de moyenne intensité. Pour ces problèmes de haute dimension, nous fixons la dimension maximale du sous-espace à 30. Le nombre de pré-divisions est indiqué par la colonne s_0.

Results

Le tableau 2.4 présente les résultats en utilisant les mêmes métriques que pour le problème de classification MNIST.

TABLE 2.4 – Comparaison numérique des performances des méthodes de l'état de l'art et de WGMprop sur un bruit de flou à l'intensité I_3. Écart-type en ().

			PERFORMANCE CRITERIA				
Dataset	**Method**	**# Gaussian (↓)**	**2W ($\times 10^3$) (↓)**	**MAPE$_{\text{MEAN}}$ (↓)**	**MAPE$_{\text{STD}}$ (↓)**	**IOU$_{95}$ (↑)**	**TIME (s) (↓)**
CIFAR-10	MC (reference)						38.83
	UT@FN	1	12.15(14.32)	16.96(13.58)	30.60(16.74)	0.634(0.122)	0.39(0.10)
	WGMprop@FN	629	0.64(0.88)	0.63(0.63)	1.73(1.65)	0.877(0.061)	1.77(0.16)
CIFAR-100	MC (reference)						49.92
	UT@FN	1	1.39(2.18)	60.33(43.67)	35.61(20.33)	0.320(0.227)	0.41(0.12)
	WGMprop@FN	709	0.16(0.32)	3.45(3.33)	2.84(2.99)	0.783(0.121)	6.54(0.55)
Camelyon	MC (reference)						359.21
	UT@FN	1	6.96(12.32)	43.93(87.48)	145.78(345.12)	0.131(0.181)	23.04(5.72)
	WGMprop@FN	309	0.25(0.40)	3.80(12.93)	25.14(59.98)	0.783(0.184)	28.68(7.18)

Nous observons une augmentation nette du temps d'exécution pour la référence MC et les deux méthodes comparées. En effet, les trois méthodes partagent un temps incompressible de sélection du sous-espace actif (environ 0.35 secondes pour les jeux de données CIFAR-10 et CIFAR-100 et environ 22 secondes pour le jeu de données Camelyon). WGMprop maintient un temps de propagation compétitif par rapport à UT@FN, qui nécessite la propagation de seulement $2r + 1$ échantillons, où r représente la dimension du sous-espace actif. WGMprop maintient des métriques globales de haute performance pour les ensembles de données CIFAR-10 et CIFAR-100, étant 22 fois plus rapide que Monte Carlo dans le cas du CIFAR-10 et 8 fois plus rapide pour le CIFAR-100. Enfin, pour l'ensemble de données Camelyon, qui est une tâche de classification binaire avec une sigmoïde utilisée comme dernière couche du réseau, WGMprop montre de bonnes performances pour les mesures 2W et $\mathbf{MAPE}_{\text{STD}}$. Cependant, la configuration de classification binaire peut produire des PDF de sortie extrêmement concentrés autour de sa moyenne, où la notion d'écart-type et de percentile ne semble plus pertinente. Cela explique très probablement les résultats obtenus pour $\mathbf{MAPE}_{\text{STD}}$ et IOU_{95}.

Les figures 2.11-2.12 proposent de visualiser les distributions de sorties de la marginale principale sur les échantillons n°2 et 89 du jeu de test de la base CIFAR-10 pour les trois types de bruit à intensité maximale.

Il est intéressant de noter que pour l'image n°2 et en présence d'un bruit gaussien de forte intensité, la méthode proposée ne permet pas de capturer convenablement la forme de la distribution de sortie. La sous-section 2.2.4 discute de ce problème et propose des solutions possibles.

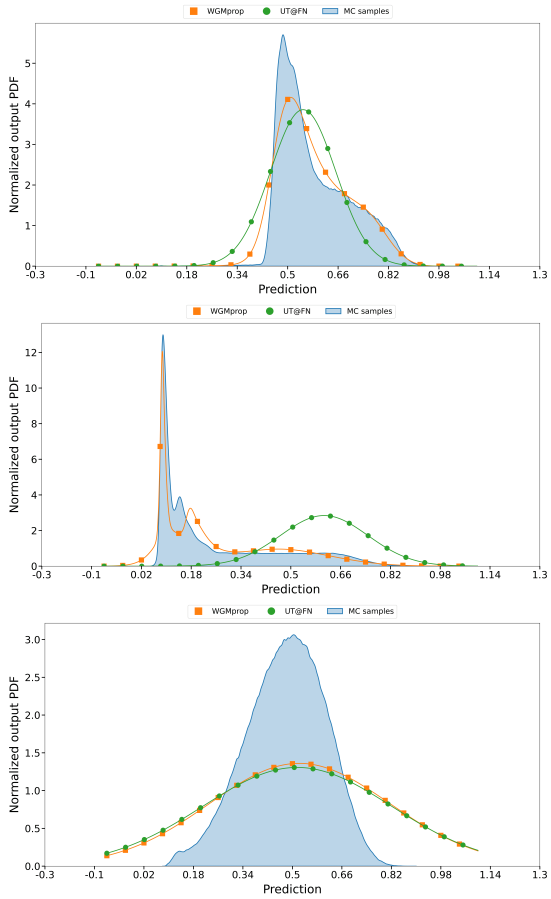


FIGURE 2.11 – PDFs de sortie estimée (marginale principale) pour l'image n°2 du jeu de données CIFAR-10 corrompu par un bruit de flou (**Haut**), bruit de contraste (**Milieu**) et un bruit additif gaussien (**Bas**) (Intensité I_3). La référence de Monte-Carlo est présentée en bleu.

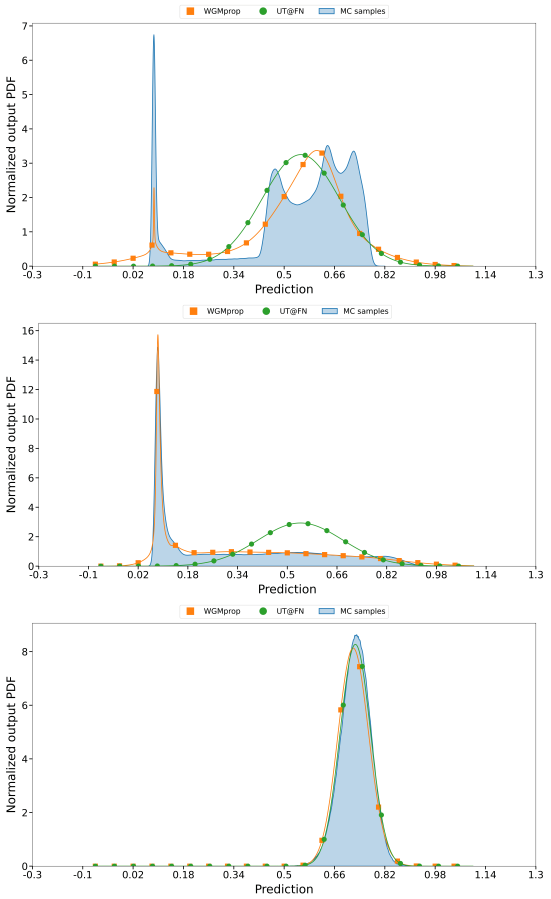


FIGURE 2.12 – PDFs de sortie estimée (marginale principale) pour l'image n°89 du jeu de données CIFAR-10 corrompu par un bruit de flou (**Haut**), bruit de contraste (**Milieu**) et un bruit additif gaussien (**Bas**) (Intensité I_3). La référence de Monte-Carlo est présentée en bleu.

2.2.4 Comparaison des directions de division dans le cas d'un bruit gaussien

Nous venons d'observer que notre méthodologie peut présenter des résultats moins satisfaisants en présence d'un bruit gaussien sur certains échantillons. Cela peut être expliqué par le fait que l'intégralité des valeurs propres d'entrée ont la même intensité. Ainsi, lors de l'étape de division, sélectionner la direction présentant la valeur propre la plus élevée revient à sélectionner aléatoirement la direction de division. Il est probable qu'une sélection aléatoire de cette direction ne permette pas de trouver la direction la plus impactée par le caractère non linéaire du réseau. Le nombre de composantes au sein de la mixture augmentant exponentiellement en fonction du nombre d'étapes de division, le budget mémoire alloué est atteint avant que la direction à privilégier soit sélectionnée

lors de la division.

Pour palier à ce problème, nous étudions sur un bruit gaussien, les trois méthodes de sélection de direction qui ont été présentées à la section 2.1.5. Nous incluons également dans notre comparaison un critère de sélection similaire à celle de plus forte non-linéarité et définie au sein de la définition 2.2.1.

Définition 2.2.1. Soit $Z_{i,[j]}$, les échantillons UT de $P_{\hat{\boldsymbol{X}}^{[s]},i} = \mathcal{N}(\mu_{\hat{\boldsymbol{X}}^{[s]},i}, \Sigma_{\hat{\boldsymbol{X}}^{[s]},i})$. Soit $\boldsymbol{f}$, le réseau de neurones et $Y_{i,[j],L-1} = \boldsymbol{f}(Z_{i,[j]})$.

La direction j_{split} de plus forte non-linéarité est définie comme :

$$j_{\text{split}} = \underset{j}{\text{argmax}} \left\| Y_{i,[j+d],L-1} + Y_{i,[j],L-1} - 2Y_{i,[0],L-1} - \boldsymbol{f}(0) \right\|$$

Cette direction s'obtient facilement en exploitant l'hypothèse de linéarité locale de $\boldsymbol{f}$. En effet, si $\boldsymbol{f}$ est linéaire, alors $\boldsymbol{f}(Z_{i,[j+d]} + Z_{i,[j]} - 2Z_{i,[0],L-1}) = \boldsymbol{f}(0) \approx \boldsymbol{f}(Z_{i,[j+d]}) + \boldsymbol{f}(Z_{i,[j]}) - 2\boldsymbol{f}(Z_{i,[0],L-1})$. Ce choix de direction sera présentée sous la référence "M" par la suite.

Nous reprenons notre démarche expérimentale appliquée à la base de données CIFAR-10, et nous comparons nos résultats sur 100 images des quatre méthodes de sélection précédemment présentées :

TABLE 2.5 – Comparaison des critères de sélection de direction de division pour WGMProp et un bruit gaussien de moyenne intensité. Les écarts type sont présentés en ().

			CRITÈRE DE PERFORMANCE					
Bruit	**Intensité**	**Méthode**	**# Gaussienne (↓)**	**2W (↓)**	**MEAN (↓)**	**STD (↓)**	**IOU$_{95}$ (↑)**	**TIME (s) (↓)**
Gaussien	I_3	Plus forte incertitude	69.64(91.58)	0.06(0.28)	1.78(1.60)	7.30(6.21)	0.679(0.135)	10.31(11.62)
		Non-linéarité (Faubel and Klakow, 2010)	63.88(82.27)	0.06(0.34)	1.74(1.63)	9.74(7.59)	0.685(0.138)	9.48(10.45)
		M	61.48(79.81)	0.05(0.26)	1.75(1.79)	10.88(10.26)	0.699(0.165)	9.79(10.76)
		Non-gaussiannité (Straka et al., 2016)	72.52(105.40)	0.07(0.34)	1.71(1.62)	9.12(7.63)	0.682(0.152)	10.73(13.58)

Nous observons que l'introduction d'un critère de sélection de direction dans le cas d'un bruit diagonale isotrope (telle que le bruit gaussien) permet d'améliorer légèrement l'estimation finale. Notre critère précédemment présenté est celui montrant les meilleurs résultats en termes de distance de Wasserstein **2W** et de l'estimation de l'intervalle **IOU$_{95}$**. C'est également le critère présentant le plus faible nombre de gaussiennes nécessaires au sein de la mixture d'entrée. En revanche, cette amélioration reste légère. Pour améliorer sensiblement ces résultats, il serait opportun de diminuer le seuil T_{split}. En effet, il semblerait que l'erreur finale sur la prédiction provient de l'accumulation de petites erreurs présentes pour l'ensemble des composantes de la mixture. Enfin, nous notons que la direction de plus forte non-gaussianité présente les meilleurs résultats en terme d'estimation de la moyenne de la distribution de sortie ainsi que son écart-type.

Dans le cas d'un bruit diagonal isotrope, nous recommandons donc l'utilisation d'un critère estimant la meilleure distribution de division telle que notre approche précédemment présentée ou l'approche de Straka et al. (2016).

2.3 Discussion

Bien que précise, robuste et générique, notre méthode basée sur les modèles de MG nécessite l'ajustement empirique des hyperparamètres pour l'étape de division. Un mauvais choix de seuil conduit à une augmentation significative de l'utilisation de la mémoire et du temps de calcul, l'automatisation de son réglage pour les performances attendues en termes de précision pourrait être une amélioration significative de la méthode proposée. En effet, dans les expériences présentées, les hyperparamètres ont été fixés empiriquement. De plus, pour une matrice de covariance avec un sous-espace actif de haute dimension, un compromis entre la précision et la performance doit être trouvé en sélectionnant sa dimension. Nous avons également mis en lumière qu'il est nécessaire pour certains types de bruits la sélection de la direction de division afin d'améliorer les performances sans pour autant augmenter drastiquement la complexité calculatoire.

Dans ce chapitre, nous avons abordé la question de la propagation de l'incertitude de l'entrée pendant la prédiction en propageant la distribution d'entrée au sein du réseau de neurones à l'aide d'une mixture de gaussiennes dont le nombre de composantes est contrôlé par un critère de 2-Wasserstein. L'analyse théorique de cet algorithme a conduit à fournir des garanties théoriques de convergence mais également de l'erreur effectuée sur les estimations des éléments statistiques de la distribution de sortie. Nous avons montré, grâce à un vaste choix d'expériences, le caractère générique, robuste et adaptatif de notre proposition sur de nombreuses taches et scénarios. Notre proposition offre également un temps d'exécution très compétitif comparé au tirage classique de Monte Carlo, et constitue une alternative de choix pour certaines applications.

En effet, ce paradigme tient toute sa place dans la "boite à outil" du métrologue désirant étudier l'impact de différent types de bruits et d'intensités sur le réseau de neurones étudié. Ce cas d'application en condition laboratoire permet d'offrir au métrologue une généricité adéquate tout en fournissant une erreur associée à l'estimation, constituant une vraie valeur ajoutée. Enfin, nous pouvons également souligner la pertinence de cet algorithme dans des application de type embarqué, où le budget temps/mémoire est très restreint. Ainsi, WGMprop offre la possibilité d'utiliser l'intégralité du budget temps, en fournissant une nouvelle fois une erreur associée à l'estimation.

Ainsi, WGMprop est un très bon compromis entre le tirage "à l'aveugle" de Monte Carlo et des méthodes faisant des hypothèses fortes sur la forme de la distribution de sortie. WGMprop peut

également être très utile afin de propager l'incertitude au sein de fonctions plus classiques rencontrées en métrologie. On peut voir la méthode WGMprop comme une méthode permettant d'effectuer une propagation type GUM en propageant les variances de façon linéaire. WGMprop permet ainsi de "découper" la distribution d'entrée afin de vérifier les hypothèses de propagations explicitées en **??**.

Enfin, nous avons pu démontrer le caractère générique de notre approche en considérant les réseaux de neurones comme une boite noire. Ainsi, WGMprop a été implémenté avec succès sur l'intégralité des réseaux convolutifs de l'API Tensorflow.

Chapitre 3

Vers une caractérisation de la méthode des ensembles profonds pour la quantification de l'incertitude des paramètres

Pour des modèles paramétriques tels que des réseaux de neurones, l'estimation des paramètres à l'aide d'une base de données d'exemples constitue, en général, une tâche difficile. Cette estimation étant soumise à différentes sources d'incertitude, il est important de pouvoir les quantifier et les propager afin d'estimer la distribution prédictive résultante. Cette dernière provient de l'intrication de deux sources distinctes (cf. section 1.5.6) : l'incertitude d'échantillonnage (cf. section 1.5.2), provenant d'un manque d'informations au sein de la base d'apprentissage, et l'incertitude de répétabilité, résultante de la stochasticité inhérente des algorithmes de descente de gradient utilisés (SGD (Ruder, 2016), RMSprop (Tieleman et al., 2012), Adam (Kingma and Ba, 2014), ...). L'état de l'art sur cette problématique est dense et florissante (cf. section 1.2.2), avec de nombreuses méthodes permettant d'obtenir des jeux de paramètres distincts mais dont la performance globale reste quasi-constante. Dans ce chapitre, nous explorons cette problématique en proposant, tout d'abord, une étude comparative de ces différentes méthodes, qui sera complétée par une courte méta-analyse, permettant une mise en perspective des résultats obtenus. Cette étude permettra notamment de conclure de la supériorité de la méthodologie des ensembles profonds (Lakshminarayanan et al., 2017) sur une majorité des métriques sélectionnées. Les ensembles profonds sont alors à l'état de l'art par leurs performances mais également par leur simplicité d'implémentation (il s'agit simplement d'une approche de Monte-Carlo de l'entraînement). D'un point de vue métrologique, les ensembles profonds revêtent également une importance de premier plan, puisqu'ils permettent, par définition, de quantifier l'incertitude de répétabilité (voir entre autres la section 1.5.2). Ils présentent néanmoins deux désavantages majeurs :

— l'obtention d'un jeu de paramètres peut être très coûteux en temps, certaines taches nécessitant parfois jusqu'à plusieurs jours voir semaines d'entraînement,
— le manque de quantification de l'erreur des estimateurs proposés, pouvant être lié au nombre de réseaux nécessaires au sein de l'ensemble considéré.

Les travaux récents de (Ainsworth et al., 2022), suggèrent la possibilité d'une caractérisation locale des ensembles profonds. En effet, ils étudient les permutations d'un réseau de neurones, laissant le prédicteur inchangé, mais pouvant le rendre linéairement interpolable avec un second.

Malheureusement, ce phénomène n'étant pas vérifié dans un cadre général, les travaux présentés dans ce chapitre étudient la possibilité d'approcher ces ensembles profonds localement, en empruntant des "tunnels" de faible valeur de coût. Cette méthodologie poursuit l'état de l'art sur la caractérisation du "paysage" des paramètres d'un réseau de neurones, entamé, par exemple, par les travaux de Garipov et al. (2018), Wortsman et al. (2021) et Benton et al. (2021), qui caractérisent localement ce paysage par des éléments géométriques tels qu'une droite, une courbe de Bézier ou encore un simplexe de faible dimension. Notre approche, générique et simple d'implémentation, offre notamment un coût calculatoire réduit au temps d'entraînement et des performances encourageantes comparées aux autres méthodes de l'état de l'art, mais reste encore en deçà des ensembles profonds. Ainsi, la section 3.1 présente une étude comparative effectuée sur un large panel de méthodes de l'état de l'art, et la section 3.2 détaille nos développements méthodologiques. En conclusion, nous formulerons des recommandations concernant la quantification de l'incertitude des paramètres d'un réseau de neurones, ainsi que les perspectives et les développements futurs de nos travaux.

3.1 Analyse comparative des méthodes à l'état de l'art sur les tâches MNIST et CIFAR-10.

Dans cette section, nous présentons une étude comparative des principales méthodes de l'état de l'art : ensemble profond (Lakshminarayanan et al., 2017) (DE), SGLD (Welling and Teh, 2011), SGHMC (Chen et al., 2014), cSGHMC (cyclic-SGHMC (Loshchilov and Hutter, 2016)), cSGLD (Loshchilov and Hutter, 2016), MCD (Gal and Ghahramani, 2016), SWAG (Maddox et al., 2019) et bootstrap (Breiman, 1996) (DE bootstrapé).

L'expérience présentée ici porte sur deux tâches de classification : MNIST (LeCun, 1998) et CIFAR-10 (Krizhevsky et al., 2009). Pour réaliser ces tâches, deux réseaux de neurones ont été choisis, un perceptron multi-couches (Haykin, 1998) (MLP) et un VGG16 (Simonyan and Zisserman, 2014). Ci-dessous, le détail des architectures :

— **MLP :** réseau constitué de 4 blocs, chacun composé d'une couche entièrement connectée de

1000 neurones, d'une couche de batch normalisation (Ioffe and Szegedy, 2015) et d'une ReLU. Le réseau est muni d'une couche de Softmax comme fonction d'activation de sortie. Ce réseau comprend 3 806 010 paramètres.

— **VGG16 :** réseau VGG à l'architecture classique (Simonyan and Zisserman, 2014) muni de couches de batch normalisation et de fonction d'activation ReLU. Ce réseau comprend 37 558 848 paramètres.

Pour chaque méthode, nous obtenons des jeux de paramètres des réseaux à l'aide des hyper-paramètres spécifiés au sein du tableau 3.1 :

TABLE 3.1 – Hyperparamètres selectionnés pour les méthodes de l'état de l'art.

Méthode	optimizer	weight_decay	lr_init (MLP/VGG16)	prior_std	nb cycle	num_samples_per_cycle	$p_{dropout}$
DE	SGD	0.0005	0.1 / 0.01	-/-	20	1	-/-
BOOTSTRAP	SGD	0.0005	0.1 / 0.01	-/-	20	1	-/-
SGHMC	SGD	-/-	0.1 / 0.01	0.2	1	20	-/-
SGLD	SGD	-/-	0.1 / 0.01	0.2	1	20	-/-
cSGHMC	SGD	-/-	0.1 / 0.01	0.2	4	5	-/-
cSGLD	SGD	-/-	0.1 / 0.01	0.2	4	5	-/-
MCD	SGD	0.0005	0.1 / 0.01	-/-	1	1	0.25
SWAG	SGD	0.0005	0.1 / 0.01	-/-	1	20	-/-

La programmation du taux d'apprentissage est faite par un recuit-cosinus (Loshchilov and Hutter, 2016). Seule la méthode SWAG suit une programmation du taux d'apprentissage linéaire par morceaux comme présenté dans l'article original (Maddox et al., 2019).

Enfin, les 8 méthodes sélectionnées sont comparées en utilisant les métriques ci-dessous :

— **NLL (Negative Log-Likelihood)**

L'opposé du logarithme de la vraisemblance est employé comme fonction de coût conventionnelle dans une tâche de classification multi-classes. Il permet d'évaluer la performance globale d'un réseau.

— **Précision**

La précision, exprimée en pourcentage, mesure le pourcentage de classifications correctes.

— **ECE (Expected Calibration Error)**

Cette mesure permet d'appréhender dans quelle mesure le réseau peut être excessivement ou insuffisamment confiant dans ses prédictions. Un réseau parfaitement calibré présente un taux de bonnes classifications de x% pour une valeur de prédiction (après la fonction softmax) de 0.x.

— **Entropie$^+$, Entropie$^-$**

L'entropie des prédictions du réseau pour les bonnes ($^+$) et les mauvaises prédictions ($^-$). En général, une entropie élevée est souhaitée pour les mauvaises prédictions, tandis qu'une

entropie faible est préférable pour les bonnes prédictions.

Résultats

Nous analysons tout d'abord la vitesse de convergence de la précision d'un ensemble de N prédicteurs de nos deux réseaux, auxquels nous ajoutons un ResNet20 (He et al., 2016). La figure 3.1 présente ainsi l'évolution de cette quantité en fonction du nombre de ses constituants.

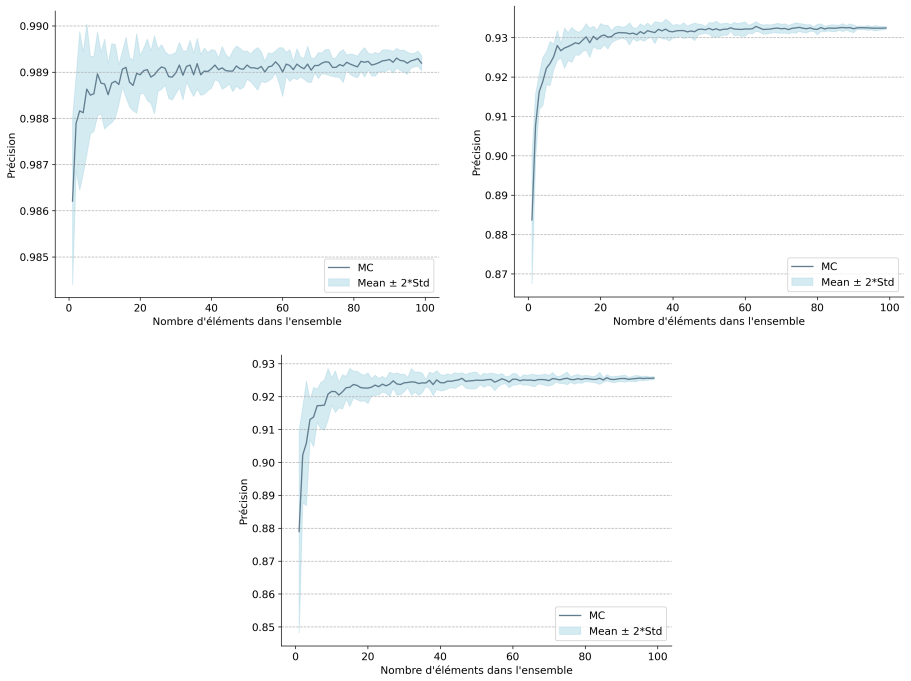


FIGURE 3.1 – Évolution de la précision d'un ensemble en fonction du nombre de ses constituants pour les tâches de classification MNIST à l'aide d'un MLP (**Gauche**), CIFAR-10 à l'aide d'un VGG16 (**Milieu**) et d'un ResNet20 (**Droite**).

Pour la tâche de classification MNIST à l'aide d'un perceptron multi-couches et pour le CIFAR-10 à l'aide d'un VGG16 et d'un ResNet20, la précision de l'ensemble converge très rapidement. On observe une courbe classique en "L" avec un coude situé pour un nombre de prédicteurs compris entre 15 et 20. Ce nombre peut sembler surprenant à la vue du nombre de paramètres des réseaux. La tâche de classification CIFAR-10, étant plus complexe que la tâche de classification MNIST, on s'attendait également à une convergence plus lente. Enfin, il faut atteindre un nombre de réseaux plus élevé, de l'ordre de la centaine, afin d'obtenir un écart-type réduit pour l'ensemble de ces trois réseaux (Pour le cas MNIST, on observe une réduction d'un facteur 4 de l'écart-type entre un ensemble constitué de 10 à 100 prédicteurs). **Dans la suite de ce chapitre, nous constituerons des ensembles de 20 prédicteurs.**

Analyse de la précision : La figure 3.2 présente l'évolution de la précision en fonction du nombre d'éléments dans l'ensemble pour les 8 méthodes sélectionnées. Le DE présente la meilleure performance sur la tâche MNIST et la deuxième meilleure performance sur la tâche CIFAR-10, juste derrière sa variante bootstrapé. La méthode de Monte-Carlo Dropout démontre une performance faible pour un petit nombre d'éléments dans l'ensemble, mais celle-ci s'améliore considérablement à mesure que le nombre d'éléments augmente. Enfin, les méthodes ne proposant pas de ré-entraînement (SGLD, SGHMC et SWAG) présentent une amélioration limitée en termes de performance par rapport aux méthodes ensemblistes. Néanmoins, la méthode cyclic-SGHMC permet d'atteindre de bonnes performances sur les deux tâches étudiées, bien qu'inférieure à la méthode DE (Deep Ensemble).

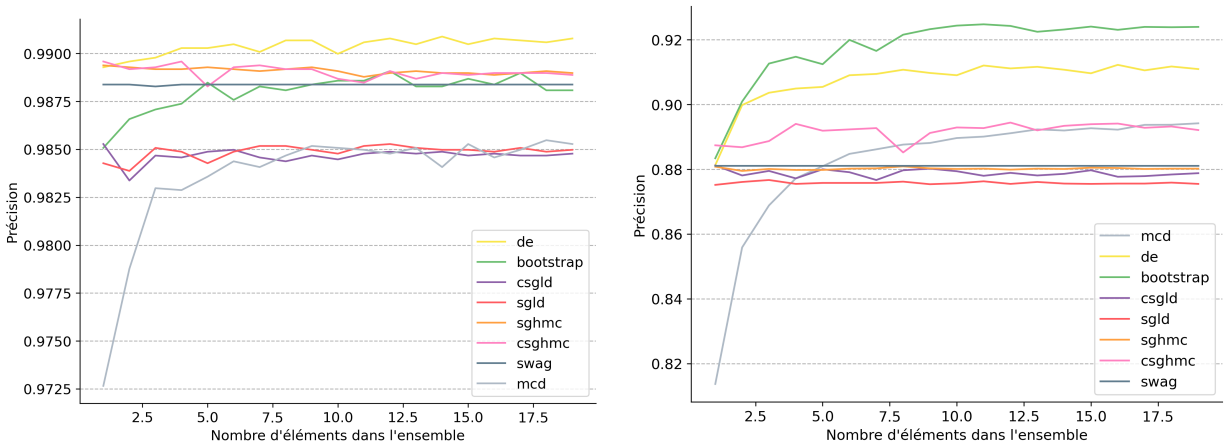


FIGURE 3.2 – Évolution de la précision d'un ensemble en fonction du nombre de ses constituants pour les méthodes de l'état de l'art sur les tâches de classification MNIST (**Gauche**) et CIFAR-10 (**Droite**).

Analyse de la calibration : Les figures 3.3 et 3.4 présentent pour chaque méthode, l'erreur de calibration moyenne (en bas à droite de chaque graphe) ainsi que la répartition de cette erreur sur les différentes plages de prédiction [1]. Dans le cadre de la tâche de classification MNIST, le MCD présente le meilleur score de calibration (0.17), suivie du DE (0.33) puis du cSGHMC (0.37). Dans ce cas, le MCD présente une bonne calibration pour l'ensemble des plages de prédiction à l'exception de la plage 0.3-0.4 (aucune de ses prédictions n'a obtenu un score compris entre 0.0 et 0.3). Dans ce cas précis, le MCD est trop peu confiant dans ses prédictions. Nous observons le même phénomène pour le DE. Enfin, malgré un score ECE plus élevé pour les méthodes cSGLD, SWAG et SGHMC, l'erreur

1. Une calibration parfaite engendrerait un graphe parfaitement diagonal (absence de zones colorées) ainsi qu'un score ECE nul. Les zones colorées en dessous de la diagonale traduisent un réseau trop confiant, à l'inverse, des zones colorées au dessus de la diagonale traduisent un réseau qui n'est pas assez confiant.

de calibration est globalement homogène sur l'ensemble des plages de prédiction. Pour la tâche CIFAR-10, c'est une nouvelle fois le MCD qui présente nettement la meilleure calibration (1.19), là où le DE reste toutefois performant (4.06) au regard du reste du panel de méthodes. On souligne le fait que la *très* bonne calibration du MCD s'observe sur l'ensemble des plages de prédictions, rendant la méthode attractive. Dans ce cas, les scores de prédictions pourraient directement être interprétés en probabilité de bonne prédiction. Contrairement au cas MNIST, la méthode Boostrap arrive, ici, en deuxième position sur cette métrique (3.02). Sur cette tâche plus complexe, et à l'exception du MCD, l'ensemble des méthodes tendent ici à être trop confiantes. Les résultats obtenus pour les méthodes MCMC sont contrintuitifs. En effet, une meilleure calibration était attendue compte tenu de leur plus faible précision.

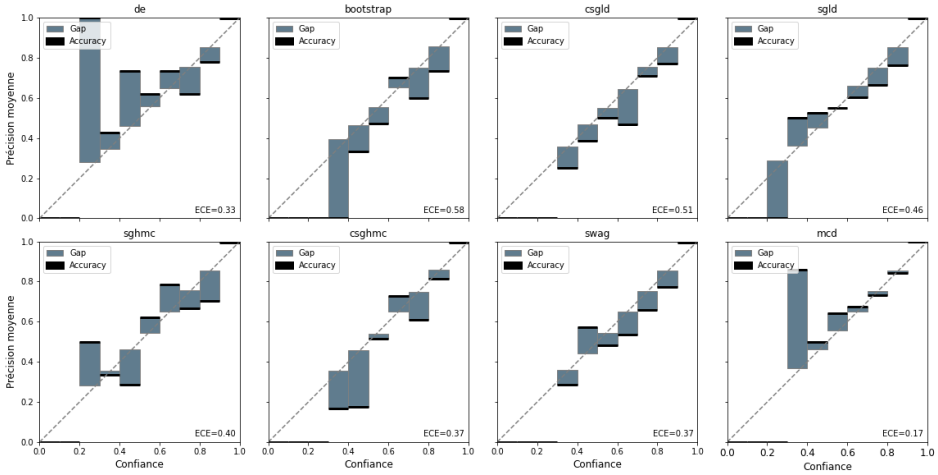


FIGURE 3.3 – Diagrammes de calibration pour la tâche MNIST (de **gauche** à **droite** et de **bas** en **haut** : DE, Bootstrap, cSGLD, SGLD, SGHMC, cSGHMC, SWAG, MCD).

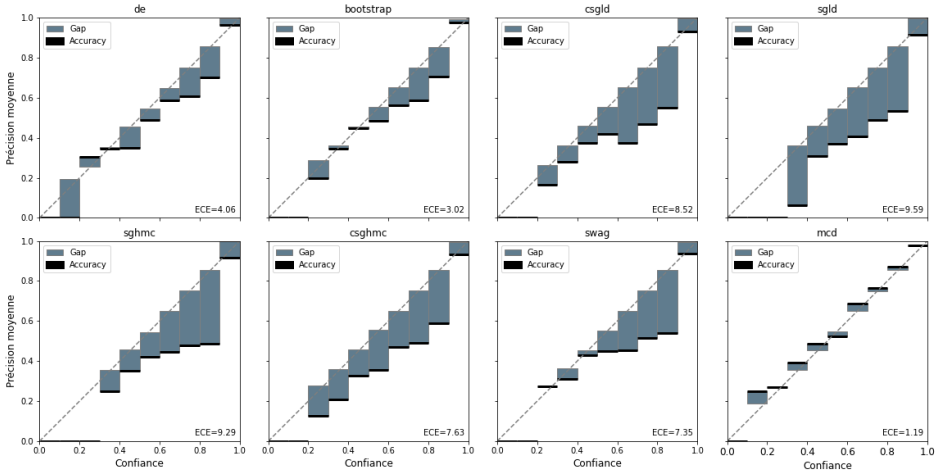


FIGURE 3.4 – Diagrammes de calibration pour la tâche CIFAR-10 (de **gauche** à **droite** et de **bas** en **haut** : DE, Bootstrap, cSGLD, SGLD, SGHMC, cSSGHMC, SWAG, MCD).

Analyse de l'entropie : La figure 3.5 présente la répartition de l'entropie des méthodes comparées sur leurs bonnes et mauvaises prédictions [2]. Bien que nous fournissions ici le graphe sur les mauvaises prédictions effectuées sur la base de données MNIST, il est difficilement interprétable dû au très faible nombre d'échantillons mal classifiés (100 échantillons pour une précision moyenne de 99%). En revanche, et à la vue des autres courbes de résultats, le comportement du MCD se détâche de toutes les autres méthodes en montrant une entropie bien supérieure. Cependant, pour la tâche CIFAR-10, le DE semble offrir le meilleur comportement (attendu), en montrant une forte entropie sur les mauvaises prédictions (juste derrière le MCD) et une faible entropie sur les bonnes prédictions. On note également que l'ensemble des méthodes, hors MCD, présentent une entropie similaire sur les bonnes prédictions.

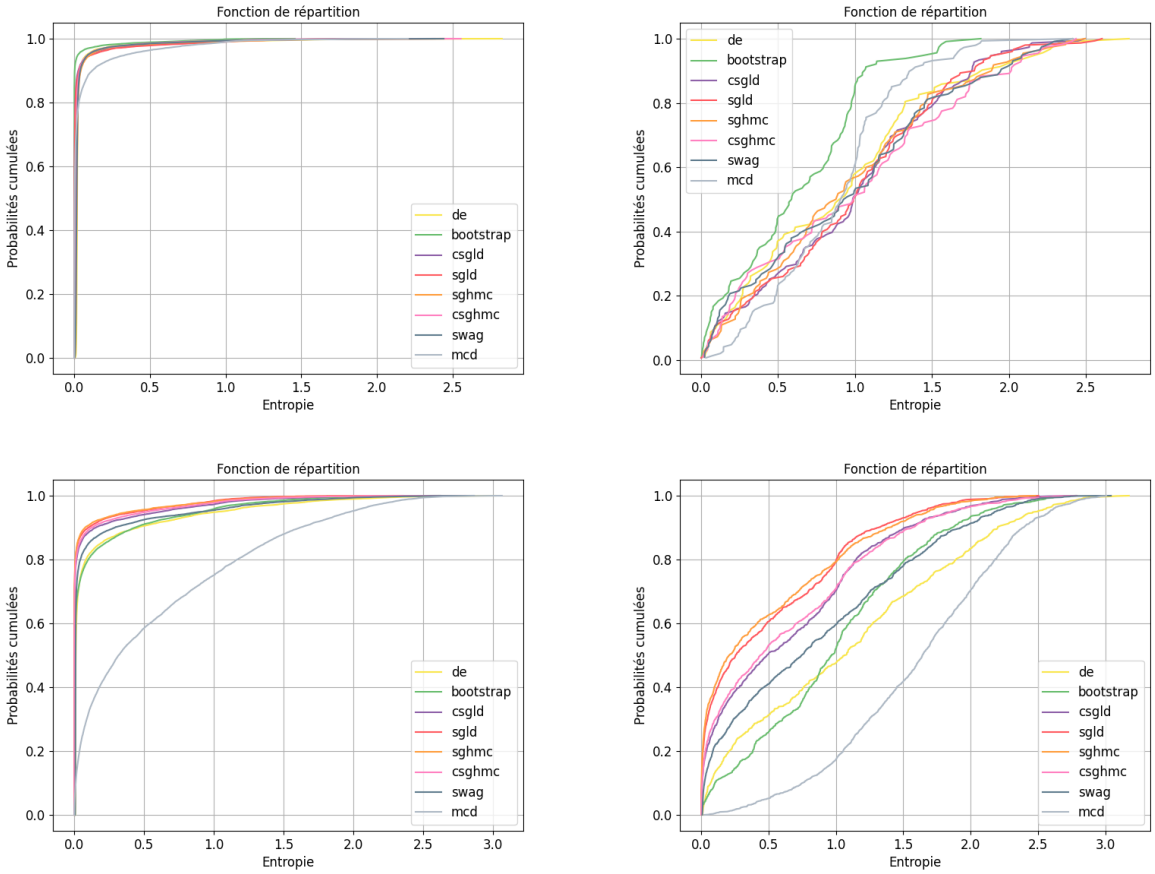


FIGURE 3.5 – Fonction de répartition de l'entropie prédictive des différentes méthodes comparées sur les bonnes (**Gauche**) et mauvaises (**Droite**) prédictions pour notre cas MNIST (**Haut**) et CIFAR-10 (**Bas**).

2. Pour rappel, le comportement désiré consiste en une faible entropie sur les bonnes prédictions et une entropie élevée sur les mauvaises prédictions.

Vue d'ensemble : Les tableaux 3.2 et 3.3 retranscrivent l'ensemble des métriques sélectionnées pour nos deux tâches. Aux métriques déjà présentées s'ajoutent le critère MCE (pour Max Calibration Error ou erreur de calibration maximale en français) ainsi que la métrique AUROC, calculée à partir de l'aire en dessous de la courbe ROC. Cette dernière permet de montrer le compromis d'un réseau entre le taux de vrais positifs et le taux de faux positifs. Un réseau parfait atteint un score AUROC de 1.0. Pour la tâche de classification MNIST, l'ensemble des méthodes de l'état de l'art montrent des performances équivalentes pour les métriques sélectionnées. On retrouve notamment que les ensembles profonds permettent d'atteindre les meilleures performances en termes de précision alors que le MCD permet d'atteindre la meilleure calibration sur la tâche considérée. Pour la tâche de classification plus complexe (ici CIFAR-10), on peut tirer des conclusions similaires bien que les différences soient plus notables entre les méthodes du panel. Le MCD obtient le meilleur score de calibration et d'entropie sur ses mauvaises classifications, alors que les méthodes ensemblistes (DE et sa variante bootstrappée) permettent d'atteindre les meilleures performances en termes de précision.

TABLE 3.2 – Critères de performance pour les méthodes de l'état de l'art sur la tâche MNIST (réseau MLP).

Méthode	NLL (↓)	Précision (↑)	ECE (↓)	MCE (↓)	Entropie$^-$ (↑)	Entropie$^+$ (↓)	AUROC (↑)
DE	0.03586	0.99079	0.00327	0.72097	0.91131	0.04432	0.99990
BOOTSTRAP	0.03626	0.98858	0.00575	0.39345	0.61972	0.01527	0.99991
SGHMC	0.03948	0.98898	0.00401	0.21649	0.94593	0.04382	0.99982
SGLD	0.05033	0.98498	0.00463	0.28935	0.96999	0.03488	0.99983
cSGHMC	0.03748	0.98888	0.00367	0.28174	0.98128	0.03483	0.99988
cSGLD	0.04888	0.98487	0.00512	0.17642	0.96771	0.03079	0.99984
MCD	0.04364	0.98528	0.00173	0.48855	0.86834	0.05825	0.99985
SWAG	0.04090	0.98838	0.00372	0.12781	0.94783	0.04052	0.99984

TABLE 3.3 – Critères de performance pour les méthodes de l'état de l'art sur la tâche CIFAR-10 (réseau VGG16).

Méthode	NLL (↓)	Précision (↑)	ECE (↓)	MCE (↓)	Entropie$^-$ (↑)	Entropie$^+$ (↓)	AUROC (↑)
DE	0.33029	0.91095	0.04057	0.19425	1.08832	0.15043	0.99282
BOOTSTRAP	0.25069	0.92318	0.03016	0.16412	0.98609	0.13001	0.99616
SGHMC	0.67970	0.88040	0.09285	0.36777	0.48842	0.05961	0.98671
SGLD	0.68649	0.87550	0.09589	0.32398	0.50023	0.06021	0.98699
cSGHMC	0.51535	0.89212	0.07634	0.26532	0.63701	0.07123	0.99011
cSGLD	0.57712	0.87880	0.08517	0.30319	0.65073	0.07931	0.98799
MCD	0.50329	0.82652	0.01189	0.06100	1.60504	0.12159	0.98361
SWAG	0.50490	0.88110	0.07352	0.31305	0.85737	0.59570	0.98760

Détection des échantillons hors distribution : Une des premières applications du BMA (cf. équation A.2) est la détection des échantillons hors-distribution. Cette notion est primordiale en apprentissage statistique du fait de la taille restreinte des bases de données d'entraînement à la vue de la variété et de la complexité des données que l'on peut rencontrer dans des cas "réels". Nous présentons, au sein de la figure 3.6, la fonction de répartition de l'entropie prédictive sur le jeu de données FMNIST (Xiao et al., 2017) des réseaux entraînés sur la tâche MNIST. Le jeu de données FMNIST est constitué d'un ensemble d'images contenant en son centre des éléments vestimentaires. L'ensemble des images de ce jeu est donc hors-distribution. Sur cet exemple, la méthode Boostrap présente les moins bons résultats en accord avec les précédents résultats sur les réseaux MNIST où cette dernière présentait la moins bonne calibration. On note une nouvelle fois que le DE est la méthode offrant les meilleures performances, accompagnée dans ce cas par les méthodes SGLD et cSGLD.

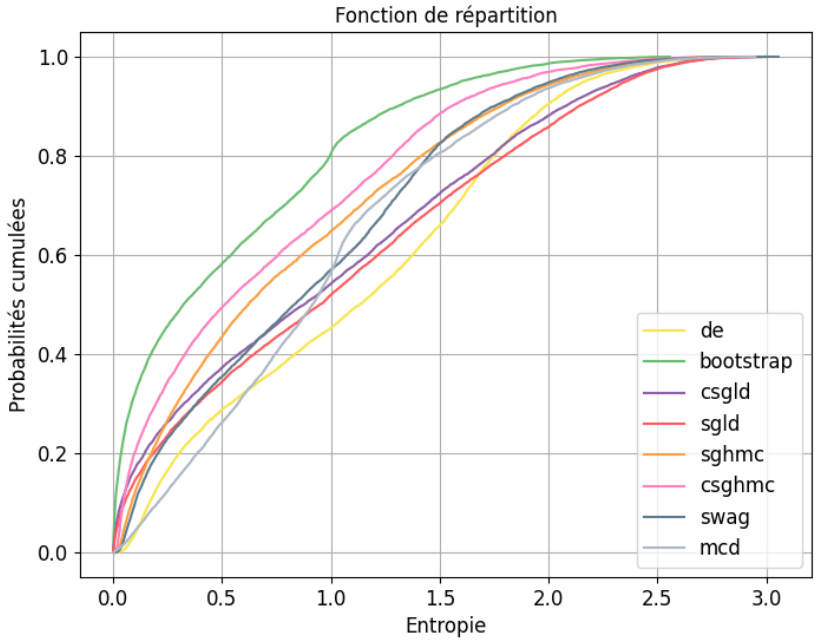


FIGURE 3.6 – Fonction de répartition de l'entropie prédictive de données hors-distributions des réseaux entraînés sur la tâche MNIST.

Méta-Analyse

L'analyse comparative précédente a montré de hautes performances du DE pour l'ensemble des métriques étudiées. Cependant, notre cadre expérimental reste restreint à la vue de la variété possible de tâches, architectures et métriques. Ainsi, nous proposons ici une vision plus élargie en proposant une courte méta-analyse incluant des études comparatives des plus récentes de l'état

de l'art : (Vadera et al., 2020; Staber and Da Veiga, 2022; Gustafsson et al., 2020; Basora et al., 2023). Ce nombre limité d'études s'explique en partie par la difficile implémentation des différentes méthodes, de la charge calculatoire en termes de temps d'entraînement et de mémoire de stockage associée. De plus, le manque de référence à laquelle se comparer rend toute analyse extrêmement difficile, obligeant d'aucun à s'appuyer sur des métriques "secondaires". Enfin, la quasi-impossibilité d'isoler une source d'incertitude particulière en rendant toutes les autres constantes augmente une nouvelle fois la complexité de la tâche.

Vadera et al. (2020) proposent une analyse comparative similaire à la nôtre en incluant notamment un réseau ResNet50 afin de traiter les tâches CIFAR-10 et CIFAR-100. Cette analyse n'inclut malheureusement pas le DE et sa variante bootstrapée. Dans leurs travaux, les auteurs obtiennent des résultats similaires pour l'ensemble des méthodes et des cas expérimentaux considérés. Les résultats sont disparates en fonction de la tâche, de l'architecture ou de la métrique considérée. À la vue de leurs résultats, il est difficile d'obtenir des conclusions générales et définitives. Cependant, nous pouvons noter quelques points intéressants :

— Les méthodes SGLD/SGHMC semblent montrer les meilleures performances quant à la détection des échantillons hors-distributions. Nous avons pu établir une conclusion similaire précédemment.
— Sur la tâche la plus complexe implémentée (CIFAR-100 muni d'un ResNet50), la méthode cSGHMC présente les meilleures performances sur l'ensemble des métriques utilisées (Précision, NLL, ECE, score Brier (Brier, 1950)).
— La méthodologie SWAG montre globalement des performances plus en retrait.
— La méthode MCD propose des performances globales homogènes sur l'ensemble des cas étudiés.

Il semble important de noter que leurs résultats font suite à une recherche approfondie des hyperparamètres optimaux pour chacune des méthodes.

Staber and Da Veiga (2022) proposent une analyse comparative dans un cadre de régression. Cette analyse n'inclut pas la méthode Bayes-By-Backprop mais intègre de nombreuses variantes des méthodes de SGLD et SGHMC. Cette étude, menée sur un "petit" perceptron comprenant entre 2 651 et 20 501 paramètres, permet une comparaison avec la méthode HMC qui constitue leur référence. Cette comparaison se fait, entre autres, en quantifiant la distance avec cette référence HMC, aussi bien dans l'espace des paramètres, que des prédictions, ainsi que la validité des intervalles de confiance fournis par les différentes méthodes étudiées. Plusieurs conclusions sont à retenir de leur étude :

— Les méthodes SGLD, SGHMC (et leurs variantes) ainsi que le DE permettent d'obtenir de bons intervalles de confiance ainsi que de bonnes performances à la prédiction.
— La méthode SWAG présente d'excellentes performances en termes de régression, mais tend à

sous-estimer l'incertitude prédictive.
— Les méthodes laplaciennes et SGLD (avec préconditionnement (Li et al., 2016)) donnent de bonnes approximations dans l'espace des paramètres.
— Le DE offre la meilleure approximation à la référence HMC aussi bien dans l'espace des paramètres que des prédictions.
— La sélection des hyper-paramètres permettant d'obtenir des intervalles de confiance valides reste une tâche difficile.

Gustafsson et al. (2020) proposent une comparaison entre le MCD et le DE sur des tâches de vision assistée par ordinateur telle que la segmentation sémantique de scène. Les auteurs présentent également un cas jouet permettant une comparaison avec une référence HMC intégrant les méthodes bayésiennes telles que SGLD et SGHMC. Ces travaux montrent que le DE fournit des estimations d'incertitude plus fiables en pratique. Les auteurs attribuent le succès du DE à sa capacité, due à l'initialisation aléatoire, à capturer l'aspect multimodale présent dans le postérieure $p(\theta|\mathcal{D}_N)$ des réseaux de neurones. De plus, le DE semble être une nouvelle fois la méthode permettant la meilleure approximation de la référence HMC. Sur les tâches en haute dimension, le DE présente de meilleures performances et une meilleure calibration (ECE) que les ensembles construit à l'aide du MCD.

Basora et al. (2023) comparent une nouvelle fois les méthodes de l'état de l'art (BBB (Blundell et al., 2015), MCD (Gal and Ghahramani, 2016), DE (Lakshminarayanan et al., 2017), Radial BNN (Farquhar et al., 2020) et Flipout (Wen et al., 2018)) sur la prévision de durée de vie restante de composants dans un cadre de maintenance. Cette comparaison se distingue des précédentes en incluant le concept d'incertitude aléatoire hétéroscédastique reprenant la dichotomie classique de la communauté de l'apprentissage profond entre incertitude *épistémique* et *aléatoire*. Dans un vaste champ expérimental, les auteurs soulignent le fait qu'aucune des méthodes étudiées semblent réellement performantes pour la détection des données hors distribution. De plus, ce travail met également en lumière qu'il est difficile d'obtenir des conclusions fermes et définitives de par la disparité des résultats en fonction des données et métriques considérées. Cependant, les auteurs concluent que, malgré la complexité accrue des réseaux de neurones bayésiens (BBB), ces derniers ne sont pas nettement plus performants que les méthodes alternatives plus simples (DE et MCD). Les auteurs soulignent néanmoins que ces deux dernières méthodes peuvent fournir des estimations conservatives de l'incertitude prédictive. Ce travail a également implémenté une recherche avancée des hyperparamètres optimaux pour chacune des méthodes étudiées.

L'analyse comparative menée en section 3.1 ainsi que les études comparatives de l'état de l'art cités ci-dessus, montrent que le DE et le MCD offrent le meilleur compromis sur l'ensemble des

mesures de performance. De plus, leur généricité et leur simplicité d'implémentation [3] les rendent attrayantes pour une utilisation dans des applications industrielles.

3.2 Exploration locale du paysage des paramètres

Dans cette section, nous désirons explorer la possibilité d'**approcher localement la méthode des ensembles profonds**, qui permet, par construction, de quantifier l'incertitude de répétabilité, inhérente aux algorithmes d'optimisation stochastique [4], **en empruntant des "tunnels" présentant de faibles valeurs de coût au sein du "paysage" des paramètres**. Cette exploration est motivée par 1) la volonté d'obtenir un ensemble de prédicteurs en un temps réduit comparé aux ensembles profonds tout en atteignant leurs performances et 2) comprendre et caractériser le "paysage" des paramètres.

3.2.1 État de l'art

Ainsworth et al. (2022) émettent la conjecture suivante :

Si l'invariance des réseaux de neurones par permutations est considérée, alors, les solutions SGD n'auront probablement pas de "barrière" dans l'interpolation linéaire entre elles.

Soient deux réseaux de neurones A et B de paramètres $\hat{\theta}_A$ et $\hat{\theta}_B$, alors la fonction de coût barrière $B\left(\hat{\theta}_A, \hat{\theta}_B\right)$ est définie par :

$$B\left(\hat{\theta}_A, \hat{\theta}_B\right) = \sup_{\alpha} \left[\mathcal{L}\left(\alpha\hat{\theta}_A + (1-\alpha)\hat{\theta}_B\right)\right] - \left[\alpha\mathcal{L}\left(\hat{\theta}_A\right) + (1-\alpha)\mathcal{L}\left(\hat{\theta}_B\right)\right] \tag{3.1}$$

où $\mathcal{L}$ est la fonction de coût utilisée lors de l'entraînement et $\alpha \in [0, 1]$.

Ainsi, on dit que les modes $\hat{\theta}_A$ et $\hat{\theta}_B$ appartiennent au même bassin local s'il n'y a pas de barrière entre eux ; i.e. si $B\left(\hat{\theta}_A, \hat{\theta}_B\right) \approx 0$. Ces chemins (ici linéaires) présentant une faible valeur de fonction de coût sont appelés des "tunnels de faible coût". Enfin, on appelle "rebasement" d'un réseau B sur un réseau A, l'application de permutations à B, lui permettant d'appartenir au même bassin local que A. Plusieurs méthodes sont proposées afin de trouver des permutations au sein du réseau de neurones laissant invariant les prédictions. Si un réseau de neurones présente de telles permutations, alors le "paysage" des paramètres se résume en un unique bassin d'attraction.

3. ces méthodes ne requièrent aucune augmentation du nombre de paramètres aussi bien lors de l'entraînement qu'à l'inférence

4. l'annexe A propose une réflexion sur la capactité du DE à quantifier l'incertitude d'échantillonnage 1.5.2

Malheureusement, la conjecture 3.2.1 n'est en général pas vérifiée [5]. Quelques travaux s'intéressent alors au sujet de la connectivité (non-linéaire) des différents modes des réseaux de neurones et de leur représentation dans l'espace des prédictions. Ainsi, Garipov et al. (2018) proposent une paramétrisation de ces tunnels par une courbe de Bézier ne présentant pas de barrière, au sens de l'équation 3.1. Pour rappel, sachant un jeu de données, le paysage des paramètres est la surface définie par la valeur moyenne de la fonction coût en fonction des paramètres du réseau. Pour une paramétrisation par une courbe de Bézier quadratique (2 degrés de liberté), l'exploration de ce paysage des paramètres est donc effectuée sur un sous-espace bi-dimensionnel. La figure 3.7 illustre le paysage des paramètres pour un ResNet-164 entraîné sur la tâche CIFAR-100 avec l'entropie croisée régularisée L_2 comme fonction de coût.

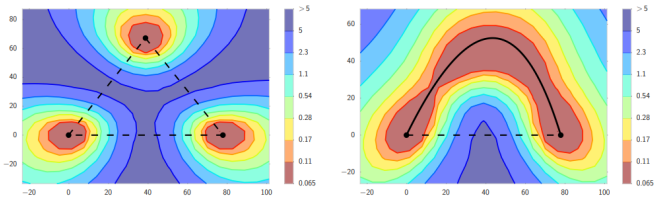


FIGURE 3.7 – Représentation bi-dimensionnelle du paysage des paramètres d'un ResNet-164 sur la tâche CIFAR-100 avec l'entropie croisée régularisée L_2 comme fonction de coût. Trois modes locaux pour des réseaux formés indépendamment (**Gauche**), Une courbe de Bézier quadratique reliant les deux modes inférieurs du panneau de gauche le long d'une trajectoire de fonction de coût quasi-constante (**Droite**). La figure est reprise de (Garipov et al., 2018)

Wortsman et al. (2021) proposent une approche concurrente, utilisant une ligne droite, une courbe de bézier ou encore un simplexe comportant m sommets pour "caractériser" un bassin d'attraction. Ces méthodes ont pour principale limite la paramétrisation de la géométrie locale dont le nombre de degrés de liberté est directement limité par la mémoire disponible. On note également les travaux de Benton et al. (2021) qui proposent également une caractérisation par simplexe.

Les méthodes reposant sur des simplexes présentent deux freins majeurs à leurs utilisations :

— La construction de simplexes nécessite un nombre de réseaux égal au nombre de sommets souhaité. En pratique, il parait difficile d'augmenter la mémoire requise d'un facteur supérieur à 2.

— L'intégralité de l'intérieur du simplexe se devant de présenter une faible valeur de coût, le volume de ce dernier est nécessairement réduit.

Dans la suite de ce chapitre, nous présentons une nouvelle approche, notée WEUQ (Weight Exploration for Uncertainty Quantification ou Exploration des paramètres pour la quantification

5. l'annexe H présente une première approche naïve de caractérisation du simplexe dont les sommets consistent en différents entraînements rebasés au sein du même bassin d'attraction

d'incertitude) pour explorer les "tunnels de faible coût" du paysage des paramètres du réseau de neurones en s'affranchissant de la paramétrisation par éléments géométriques.

3.2.2 Exploration locale du paysage des paramètres par marche aléatoire (WEUQ)

Dans cette section, nous proposons une exploration itérative en marche aléatoire droite à partir d'un mode local trouvé par optimisation SGD. La motivation première de ce développement est de concurrencer les performances des approches ensemblistes à moindre coût (temps d'entraînement).

Exploration en marche aléatoire droite

Nous rappelons qu'un réseau de neurones est le résultat de la concaténation de L blocs élémentaires notés $(h_l)_{0\leq l\leq L-1}$: $\mathbf{Y} = h_{L-1}\left(..h_1\left(h_0\left(\mathbf{X}, \theta_0\right)\right)\right) = \boldsymbol{f}(\mathbf{X}, \boldsymbol{\theta})$.
Ainsi, chaque couche linéaire h_l du réseau de neurones peut être décomposée de la manière suivante :

$$h_l = h_l^{nt} + \alpha h_l^t$$

où h_l^{nt} est une couche paramétrique statique représentant le point de départ, et h_l^t représente le vecteur directionnel de l'exploration qui est appris lors de l'entraînement courant.
Nous notons de la même manière

$$f_\alpha = f^{nt} + \alpha f^t$$

où $\alpha \in [0, 1]$ est un scalaire permettant de se balader sur la droite définie par f^{nt} et f^t. Lors de l'itération courante, f^t est initialisé de façon aléatoire, suivant le même paradigme d'initialisation que l'entraînement initial.

La mise à jour des paramètres s'effectue alors uniquement sur les paramètres encodant le vecteur directionnel d'exploration f^t. On obtient pour l'optimisation SGD classique, de pas d'apprentissage η, les équations de mise à jour suivantes :

$$h_l^{nt} := h_l^{nt} \tag{3.2}$$

$$h_l^t := h_l^t - \eta \frac{\partial \mathcal{L}_{WEUQ}}{\partial h_l^t} \tag{3.3}$$

L'exploration étant itérative, le premier point de départ f^{nt} contient donc le mode local trouvé par optimisation SGD.

La direction de recherche initiale étant aléatoirement choisie, nous proposons une alternative

à l'exploration WEUQ, nommé SWEUQ, consistant à effectuer plusieurs explorations autour du même mode local trouvé par entraînement SGD, comme illustré par la figure 3.8 :

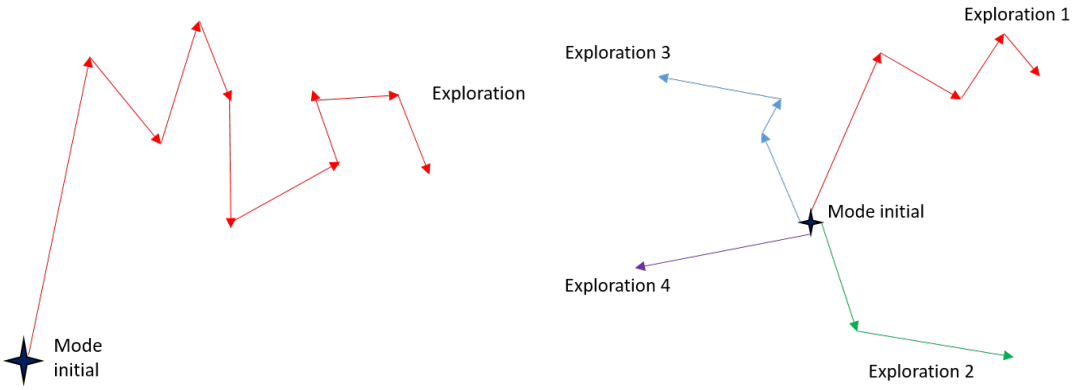


FIGURE 3.8 – Illustration d'une exploration WEUQ (**Gauche**) et d'une exploration SWEUQ (**Droite**).

Fonction de coût WEUQ

Pour explorer des tunnels de faible coût, la fonction de coût WEUQ intègre un terme de "barrière", noté $\mathcal{L}_{barrière}$, comme introduit en section 3.2.1. De plus, pour reproduire la capacité des ensembles profonds à générer naturellement une diversité élevée de prédicteurs (cf. annexe I [6] et (Gustafsson et al., 2020)), nous proposons d'ajouter un terme de régularisation, noté $\mathcal{L}_{diversité}$, pour explorer des "zones" favorisant la diversité de l'ensemble [7]. Ce terme de régularisation permettra également d'éviter la solution triviale $f^t = 0$. Enfin, nous désirons rendre ce terme de régularisation adaptatif, c'est à dire s'adaptant à la dynamique d'entraînement : $\mathcal{L}_{diversité} = \mathcal{L}_{diversité}(\gamma)$. Ainsi, la fonction de coût prendra la forme suivante :

$$\mathcal{L}_{WEUQ} = \mathcal{L}_{barrière} - \mathcal{L}_{diversité}(\gamma)$$

Établissement de $\mathcal{L}_{barrière}$

La fonction de coût naïve considérée est définie par :

$$\mathcal{L}_{barrière} = \int_0^1 \mathcal{L}(f^{nt} + \alpha f^t) d\alpha \tag{3.4}$$

où $\mathcal{L}$ est la fonction de coût utilisée lors de l'entraînement initial.

6. L'annexe I propose une étude de la performance des ensembles profonds sous l'angle de la diversité d'un ensemble de prédicteurs. Nous montrons que l'exploration multimodale du paysage des paramètres par méthodes de "restart" permet d'augmenter la diversité de l'ensemble permettant une augmentation de la performance.

7. En pratique, l'ajout d'un terme de régularisation sur la diversité peut rendre l'entraînement difficile. En effet, augmenter la diversité conduit à produire des prédictions différentes, potentiellement au détriment de la performance individuel du réseau appris.

Expérimentalement, nous avons observé que la fonction $\mathcal{L}(f^{nt} + \alpha f^t)$ est une fonction croissante en α. Il devient d'autant plus difficile de trouver des réseaux à mesure que α s'approche de 1.0. Nous proposons alors une version modifiée de la fonction de coût précédente permettant de pondérer l'erreur commise en fonction de la position α :

$$\mathcal{L}_{barrière} = \int_0^1 \alpha \mathcal{L}(f^{nt} + \alpha f^t) d\alpha \tag{3.5}$$

L'intégrale précédente étant intractable, nous proposons l'approximation suivante :

$$\mathcal{L}_{barrière} \approx \frac{1}{\sum_i \alpha_i} \sum_{i=1}^{M} \alpha_i \mathcal{L}(h_l^{nt} + \alpha_i h_l^t) \tag{3.6}$$

où $\alpha = \{\alpha_i\}_{1 \leq i \leq M}$ est un ensemble de valeurs comprises entre 0.0 et 1.0.

Établissement de $\mathcal{L}_{diversité}$

Dans le cadre d'une tâche de classification, le vecteur de sortie est normalisé grâce à la fonction d'activation *softmax*. Un ensemble de prédicteurs est d'autant plus performant que ses éléments constitutifs effectuent leurs **erreurs** sur des **entrées différentes** (cf. annexe I). Cette observation conduit à proposer le terme de régularisation présenté au sein de la définition 3.2.1.

Définition 3.2.1. Soit $p \in \mathbb{N}^*$ et notons L_p la p-distance usuelle. Soit $\mathcal{D}_N = (\mathbf{x}_i, \mathbf{y}_i)_{1 \leq i \leq N}$ une base de données d'entraînement. On note $\bar{\mathbf{y}}_i$ la prédiction moyenne de l'ensemble E sur la i^e entrée, et $\mathbf{y}_i^k$ la prédiction du k^e réseau de E sur l'entrée i. Enfin, nous notons $\mathbf{y}_{i,j}$, la j^e colonne du vecteur $\mathbf{y}_i$, $\mathbf{y}_{i \setminus j}$ le vecteur prédit sur l'entrée i privé de la colonne j et on note j^* la colonne correspondant à la vraie classe de l'entrée courante. Nous définissons notre mesure de diversité d'un ensemble E (composé de M réseaux) comme la quantité $\mathbb{D}$ définit par :

$$\mathbb{D}(E) = \frac{1}{NM} \sum_{k=1}^{M} \sum_{i=1}^{N} \gamma_i^k L_p \left(\hat{\mathbf{y}}_{i \setminus j^*}^k, \bar{\mathbf{y}}_{i \setminus j^*} \right) \tag{3.7}$$

avec :

$$\gamma_i^k = \tanh \frac{1}{2} (L_p \left(\mathbf{y}_{i,j^*}^k, 1.0 \right) + L_p \left(\bar{\mathbf{y}}_{i,j^*}, 1.0 \right)) \tag{3.8}$$

Lorsque la prédiction de l'ensemble est erronée, on a $L_p \left(\bar{\mathbf{y}}_{i,j^*}, 1.0 \right) >> 0$, donc $\tanh L_p \left(\bar{\mathbf{y}}_{i \setminus j^*}, 1.0 \right) \approx 1.0$. Au contraire, lorsque la prédiction de l'ensemble est correcte, i.e. $L_p \left(\bar{\mathbf{y}}_{i \setminus j^*}, 1.0 \right) \approx 0.0$, on obtient $\tanh L_p \left(\bar{\mathbf{y}}_{i \setminus j^*}, 1.0 \right) \approx 0.0$. Ainsi, le terme γ donne une "importance" supérieure aux entrées sur lesquels nos réseaux présentent une faible performance sans dégrader les prédictions sur les entrées correctement classifiées.

Cette définition vérifie une propriété intuitive, détaillée au sein de la proposition 3.2.2.

Proposition 3.2.2. *Soit E un ensemble de prédicteurs au sein d'une tâche de classification dans $\mathbb{R}^{d'}$ et $\mathbb{D}$ une métrique de diversité définie précédemment, munie d'une distance (appelée distance élémentaire) L_p avec $p \in \mathbb{N}^*$.*

$$\mathbb{D}(E) = 0 \leftrightarrow \forall k \in [0...M] \quad \forall i \in [0...N] \quad \mathbf{y}_i^k = \bar{\mathbf{y}}_i \tag{3.9}$$

Preuve de la proposition 3.2.2. Soit E un ensemble de prédicteurs au sein d'une tâche de classification dans $\mathbb{R}^{d'}$ et $\mathbb{D}$ une métrique de diversité définie précédemment, muni d'une distance L_p avec $p \in \mathbb{N}^*$.

Immédiatement, on a :

$$\forall k \in [0...M] \quad \forall i \in [0...N] \quad \mathbf{y}_i^k = \bar{\mathbf{y}}_i \rightarrow \mathbb{D}(E) = 0$$

Maintenant, si $\mathbb{D}(E) = 0$ alors pour tout $k \in [0...M]$ et pour tout $i \in [0...N]$, on a $\gamma_i^k = 0$ ou $L_p\left(\mathbf{y}_{i\backslash j^*}^k, \bar{\mathbf{y}}_{i\backslash j^*}\right) = 0$. Si $\gamma_i^k = 0$ alors $\mathbf{y}_{i,j^*}^k = \bar{\mathbf{y}}_{i,j^*}^k = 1.0$, or $\sum_j \mathbf{y}_{i,j}^k = \sum_j \bar{\mathbf{y}}_{i,j} = 1.0$ donc $\forall j \in [0...d']\backslash j^* \quad \mathbf{y}_{i,j}^k = \bar{\mathbf{y}}_{i,j}^k = 0.0$. Alternativement, si $L_p\left(\mathbf{y}_{i\backslash j^*}^k, \bar{\mathbf{y}}_{i\backslash j^*}\right) = 0$ alors $\forall j \in [0...d']\backslash j^* \quad \mathbf{y}_{i,j}^k = \bar{\mathbf{y}}_{i,j}$ et donc $\mathbf{y}_{i,j^*}^k = \bar{\mathbf{y}}_{i,j^*}$ par sommation à 1.0. □

L'annexe J étudie la pertinence de ce critère de diversité au regard de la précision d'un ensemble de prédicteurs.

Remarque 11.

Il est difficile d'évaluer une différence des prédicteurs en observant leurs différences de paramètres. Par les phénomènes de symétries et de permutations, deux réseaux peuvent être très éloignés au sein d'une distance (par exemple, les distance L_p) entre leurs paramètres, mais pour autant être identiques en termes de prédiction. C'est pourquoi, il ne nous semble pas pertinent d'introduire au sein de $\mathcal{L}_{WEUQ}$ un terme de régularisation sur les paramètres. D'autres travaux ne font pas le même choix, à l'image de Wortsman et al. (2021) qui ajoutent une similarité cosinus entre deux jeux de paramètres au sein de la fonction de coût.

La fonction de coût de la méthode WEUQ est finalement définie par :

$$\mathcal{L}_{WEUQ} = \frac{1}{\sum_i \alpha_i} \sum_{i=1}^{M} \alpha_i \mathcal{L}(h_l^{nt} + \alpha_i h_l^t) + \mathbb{D}(\{f^{nt}, f^{nt} + f^t\}) \tag{3.10}$$

où $\mathcal{L}$ est la fonction de coût utilisée lors de l'apprentissage classique conduisant au mode local (représenté par f^{nt}), $\mathbb{D}$ une mesure de diversité précédemment détaillée, et $\{a, b\}$ désigne l'ensemble de prédicteurs composé des prédicteurs a et b.

Algorithme WEUQ

Nous présentons au sein de l'algorithme 2 de la méthode WEUQ.

Algorithm 2 WEUQ

1: **Choix des α :** $\alpha_0 \sim \mathcal{U}(0,1)$,
2: **Ré-initialisation :** Mise du point de départ dans f_l^{nt} et initialisation de f_l^t,
3: **Optimisation :** Pour chaque époch et pour chaque mini-batch du jeu d'entraînement :
4: **Calcul du pas d'apprentissage :** Calcul du pas d'apprentissage
5: **Propagation en avant :** Prédiction du réseau sur le mini-batch courant
6: **Calcul de la fonction de coût :** Calcul la fonction de coût $\mathcal{L}_{WEUQ}$
7: **Rétro-propagation :** Calcul des gradients de la fonction de coût respectivement à (f^t),
8: **Mise à jour des paramètres :** $w^t := w^t - \lambda \frac{\partial \mathcal{L}_{WEUQ}}{\partial w^t}$
9: **Estimation de sortie :** Garder les poids présentant la plus faible valeur de coût sur le jeu de validation

Nous incorporons également les versions "restartées", nommée respectivement *M-WEUQ* et *M-SWEUQ*, des deux approches. Les variantes M-WEUQ, SWEUQ et M-SWEUQ sont obtenues directement à partir de cet algorithme. À l'instar de Bayes-by-Backprop (Blundell et al., 2015), les méthodes proposées augmentent la mémoire nécessaire d'un facteur 2 au moment de l'entraînement. En revanche, elles ne requièrent pas de mémoire supplémentaire à l'inférence.

3.2.3 Expériences

Le cadre expérimental est identique à celui présenté en section 3.1. Dans le cadre de la comparaison numérique avec la méthode des ensembles profonds, une taille d'ensemble contenant 20 prédicteurs (réseaux) a été maintenue. Ainsi, chacune des méthodes (DE, WEUQ, SWEUG, M-WEUQ et M-SWEUQ) dispose de 20 prédicteurs :

— WEUQ : 1 branche d'exploration de 20 prédicteurs.
— SWEUQ : 4 branches d'exploration de 5 prédicteurs.
— M-WEUQ : 4 entraînements WEUQ (de 5 prédicteurs) autour de 4 modes différents.
— M-SWEUQ : 4 entraînements SWEUQ chacun comprenant 4 branches de 1 prédicteur auquel on ajoute le mode local.

Le nombre d'époch maximal pour un entraînement classique est de 300 alors que le nombre d'époch maximal pour la méthode WEUQ est établie à 30 (pour un coût calculatoire par époch constant).

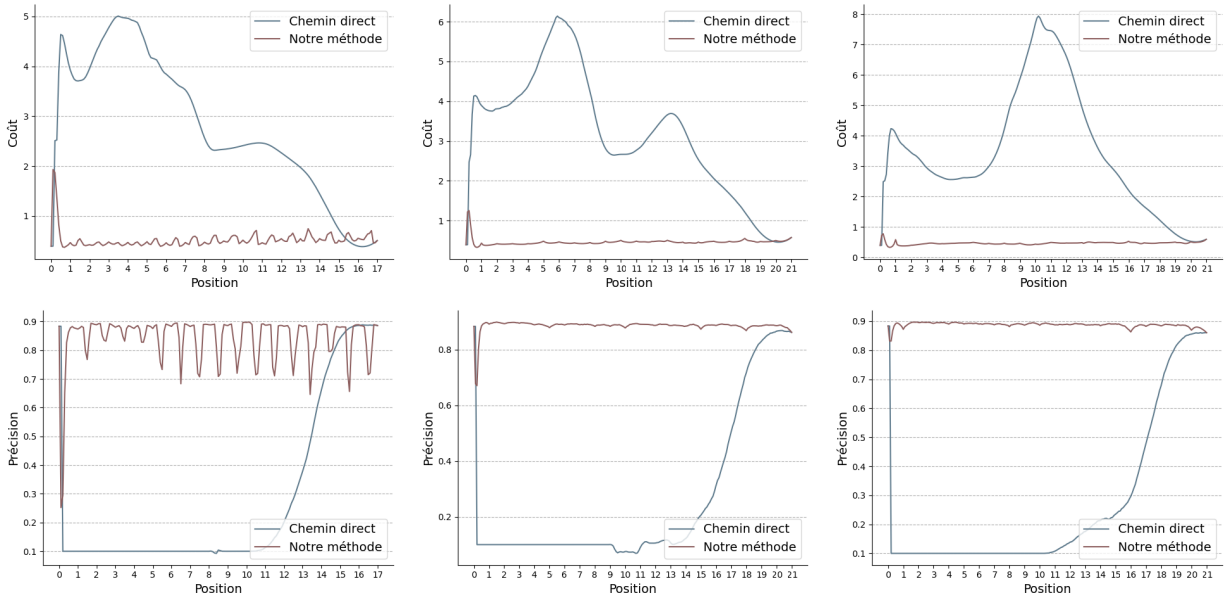


FIGURE 3.9 – Évolution du coût (**en haut**) et de la précision (**en bas**) entre le point de départ et celui trouvé après plusieurs itérations de notre algorithme WEUQ, selon un chemin direct (en **bleu**) et en suivant le chemin trouvé lors de notre exploration WEUQ (en **rouge**). Différentes stratégies de valeurs α sont présentées : $\alpha = \{0.25, 0.5, 0.75, 1.0\}$ (**à gauche**), $\alpha = \{\alpha_0, 1.0\}$ et $\alpha_0 \sim \mathcal{U}(0, 1)$ (**au milieu**) et $\alpha = \{0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0\}$ pour la première itération (**à droite**).

La figure 3.9 présente l'évolution du coût (**en haut**) et de la précision (**en bas**) entre le point de départ et celui trouvé après plusieurs itérations de la méthode WEUQ, selon un chemin direct (consistant à interpoler naïvement les deux réseaux) et en suivant le chemin trouvé lors de l'exploration WEUQ. Cette figure présente également trois variantes de la méthode WEUQ : une première approche avec $\alpha = [0.25, 0.5, 0.75, 1.0]$ (**à gauche**), une deuxième avec $\alpha = \{\alpha_0, 1.0\}$ et $\alpha_0 \sim \mathcal{U}(0, 1)$ (**au milieu**) et une troisième, identique à la deuxième approche à l'exception de la première itération où $\alpha = \{0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0\}$ (**à droite**).
On note que la méthode WEUQ permet de trouver des chemins de faible coût, là où une interpolation directe et naïve présente une forte valeur de barrière. L'échantillonnage de α_0 au sein d'une distribution uniforme permet d'obtenir des chemins quasiment sans barrière tout en maintenant un coût calculatoire faible. Enfin, la première itération présente un petit pic de barrière que l'on peut réduire en augmentant le nombre de valeurs prises par α.

Dans la suite, seule la 3^e variante est considérée, c'est à dire $\alpha = \{\alpha_0, 1.0\}$ avec $\alpha_0 \sim \mathcal{U}(0, 1)$ et $\alpha = \{0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0\}$ pour la première itération.

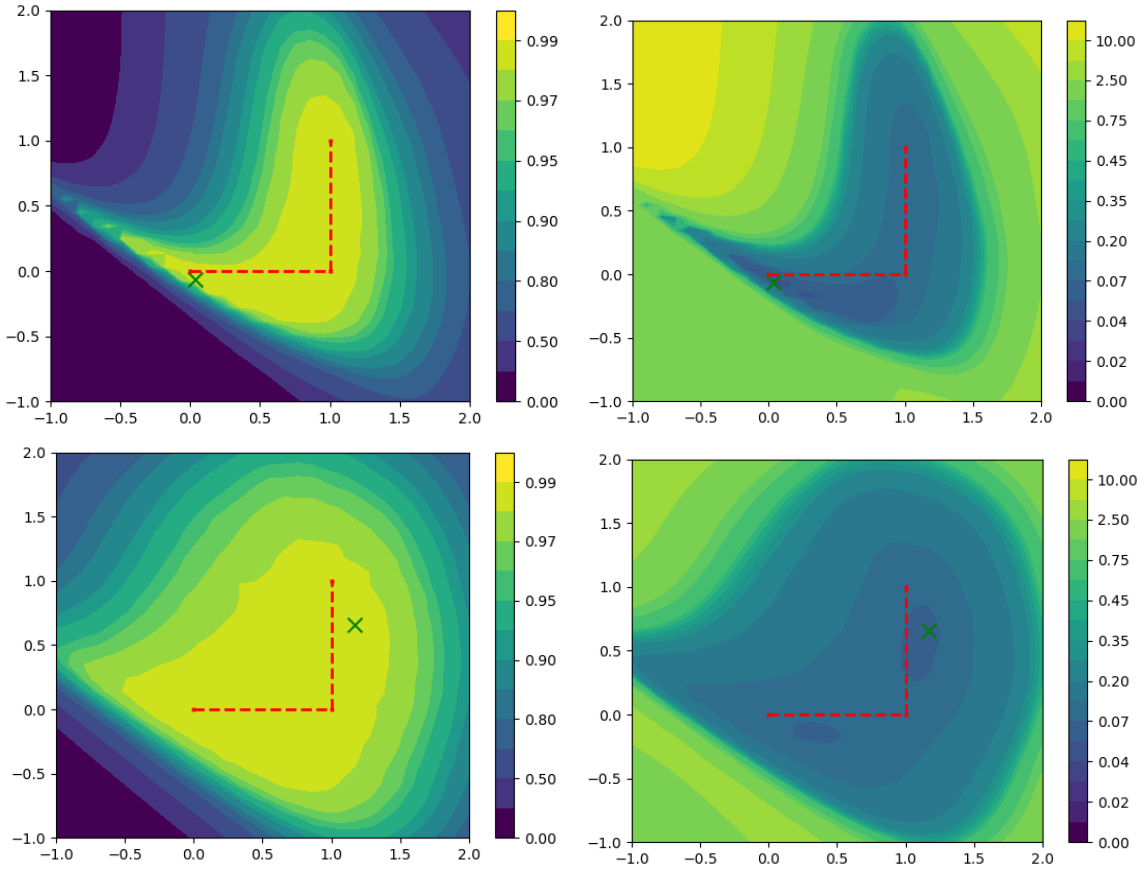


FIGURE 3.10 – Visualisation du paysage local des paramètres autour de la trajectoire d'exploration WEUQ (**en rouge**) pour le cas MNIST. L'axe horizontal représente la première direction d'exploration à partir du mode local et l'axe vertical correspond au second axe d'exploration à partir du premier point trouvé par la méthode WEUQ. La meilleure performance locale est symbolisée par une croix verte. La précision est présentée sur les figures de **gauche** et la valeur de coût à **droite**.

Les figures 3.10 et 3.11 présentent le paysage local autour de la trajectoire WEUQ pour les 4 premières itérations. Pour le cas MNIST, les deux premières itérations permettent d'"emprunter" un tunnel en "L" autour des directions trouvées lors de l'entraînement. On observe également que la zone de plus faible valeur de coût autour du mode initial est très petite au regard des pas effectués lors de l'exploration. Cependant, les pas effectués permettent de se rapprocher des extrémités de la zone dont la précision est supérieure à 0.98 et permet ainsi, en seulement deux itérations, de couvrir toute la largeur et longueur du bassin représenté autour des deux directions d'exploration. Enfin, il est intéressant de noter que l'approche proposée permet de relier deux prédicteurs non-interpolables linéairement.

Pour les deux itérations suivantes (figures du bas), on retrouve une forme en "L" du paysage local. En revanche, les deux points échantillonnés par la méthode WEUQ sont, quant à eux, interpolables linéairement. On note également que l' exploration s'arrête à proximité, mais à l'extérieur d'une zone

de plus faible coût (inférieur a 0.05). Un raffinement de l'entraînement autour du point d'arrivée, en relâchant en partie ou totalement la contrainte d'interpolation, pourrait permettre d'explorer cet autre bassin. Nous proposons ci-dessous la même visualisation pour l' exploration WEUQ sur la tâche de classification CIFAR-10 utilisant un réseau VGG16.

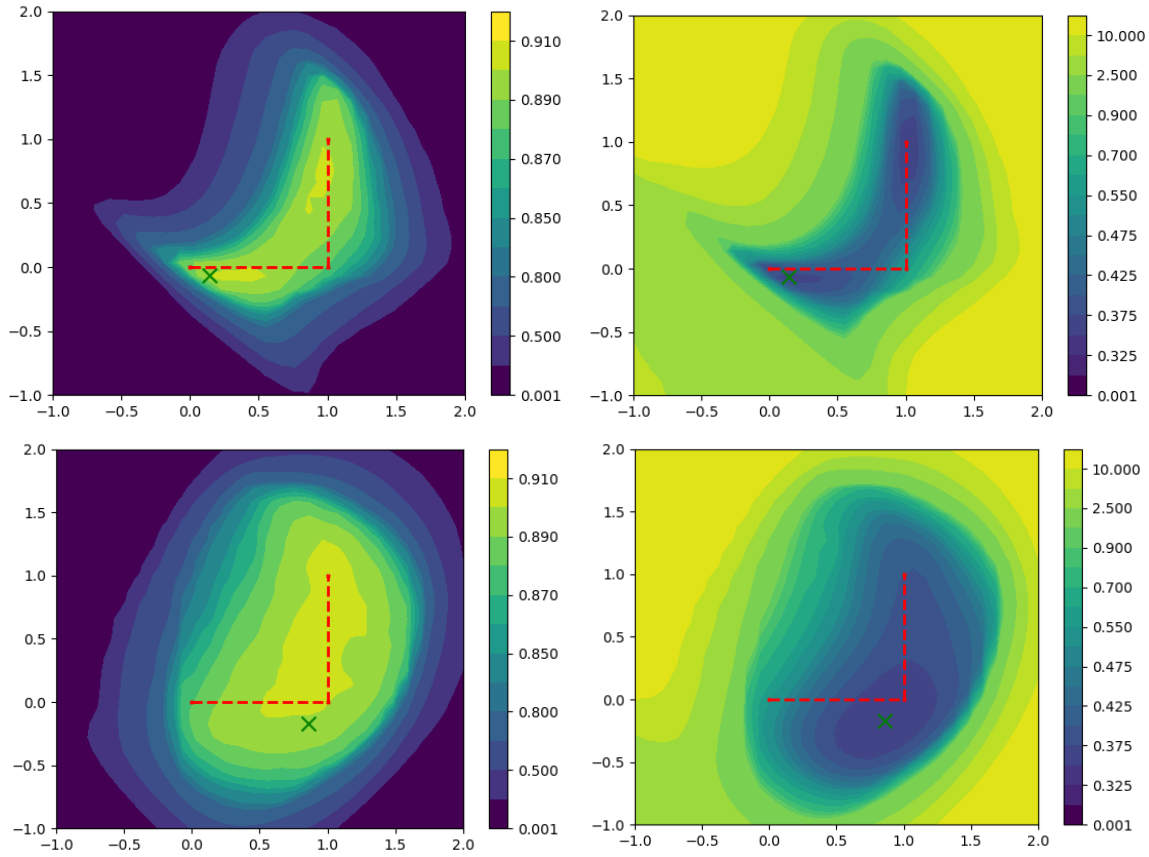


FIGURE 3.11 – Visualisation du paysage local des paramètres autour de la trajectoire d'exploration WEUQ (**en rouge**) pour le cas CIFAR-10. L'axe horizontal représente la première direction d'exploration à partir du mode local et l'axe vertical correspond au second axe d'exploration à partir du premier point trouvé par notre méthode WEUQ. La meilleure performance locale est symbolisée par une croix verte. La précision est présentée sur les figures de **gauche** et la valeur de coût à **droite**.

Pour le cas du VGG16, et pour les deux premières itérations, l' exploration conduit à "tomber" dans un second bassin d'une valeur de coût égale au bassin de départ (≈ 0.35). L'interpolation linéaire entre le point de départ et celui d'arrivée conduirait à une barrière plus élevée qu'une interpolation non-linéaire en suivant nos directions de recherche. Au regard du cas MNIST, le tunnel de faible valeur de coût semble plus étroit dans ce cas. L'échantillonnage d'un nouveau bassin équivalent en termes de performance à celui du mode initial au bout de 2 itérations est très encourageant pour la suite de nos travaux. Enfin, et comme pour le cas MNIST les itérations 3 et 4 conduisent à l'exploration d' une "zone" plus convexe du paysage des paramètres. Ces résultats préliminaires semblent valider la méthode WEUQ pour l'exploration du paysage des paramètres en suivant des

chemins présentant une faible barrière.

La figure 3.13. permet de visualiser en 2 dimensions l'intégralité du chemin parcouru durant l'exploration. Cette représentation permet d'observer les tunnels de faible coût ainsi que leurs "largeurs". Ici, l'axe des abscisses et des ordonnées ne correspondent pas à une direction unique. On peut s'imaginer cette représentation comme un dé à jouer que l'on aurait déplié sur un plan (à l'image d'un patron 2D d'une forme 3D), comme illustré au sein de la figure 3.12. Ainsi, pour une trajectoire composée d'un point de départ "A" et de dix points rencontrés (de "B" à "K"), l'image est composée de $5 \times 5 = 25$ sous-imagettes dont les axes des abscisses et ordonnées sont directement définis par les points successivement rencontrés. L'image finale produite ne présente alors pas de discontinuité.

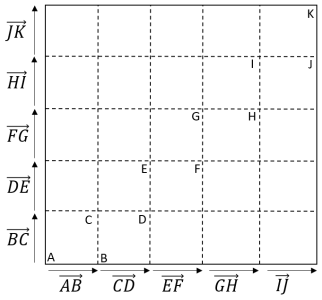


FIGURE 3.12 – Construction de la représentation 2D d'une trajectoire WEUQ.

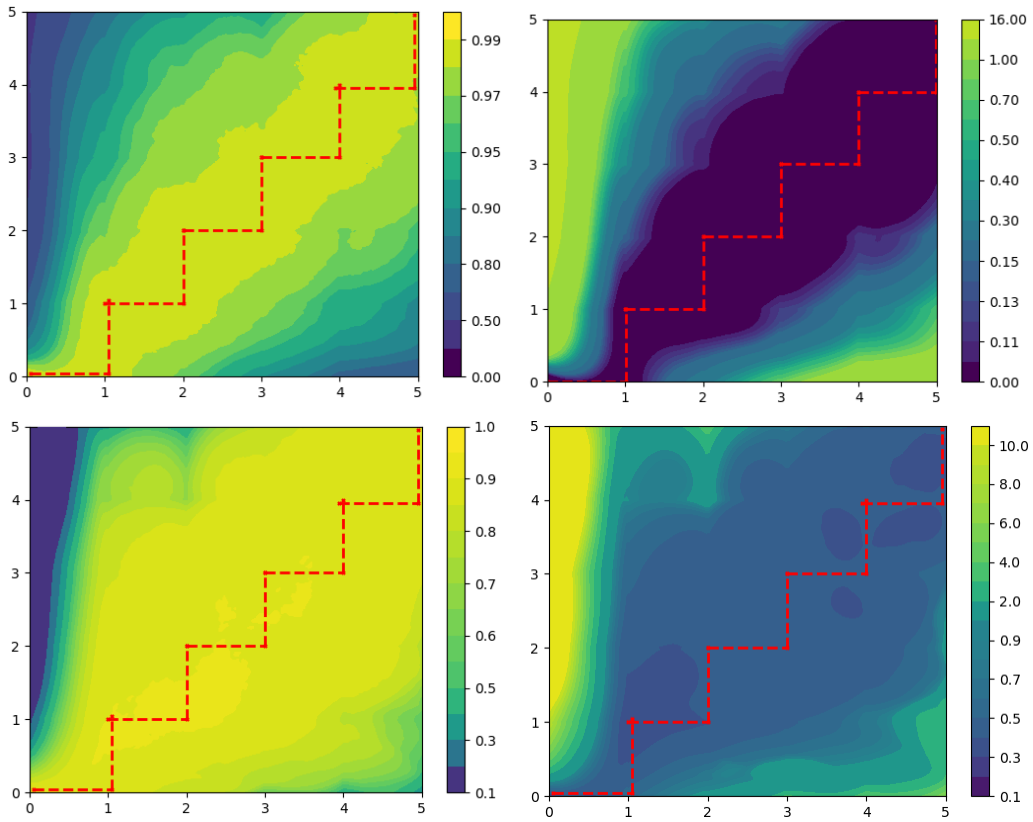


FIGURE 3.13 – Exploration WEUQ pour le cas MNIST (**Haut**) et CIFAR-10 (**Bas**). La précision à **Gauche** alors que le paysage de valeur de coût est à **Droite**.

Pour le cas MNIST, cette visualisation met en lumière les tunnels que nous empruntons grâce à notre méthodologie. Ces tunnels présentent de hautes performances avec une valeur de coût comprise entre 0.0 et 0.10 et une précision supérieur à 98%. Concernant notre cas CIFAR-10, on note également la présence d'un tunnel mais dont la "géométrie" semble plus "complexe" que pour le cas MNIST. On note, entre autres, la présence de régions "isolées" présentant de plus faibles valeurs de coût. Notre méthodologie permet ainsi d'explorer la première et la dernière de ces zones mais semblent "manquer" les autres. La méthode WEUQ explore néanmoins tout un espace présentant de hautes performances équivalentes au mode de départ. Enfin, visuellement, la largeur des tunnels autour du mode initial semble plus étroite que dans le reste du "paysage". Cette dernière observation doit cependant être prise avec précaution du fait de la quasi-impossibilité de représentation en 2 ou 3 dimensions d'un espace en très hautes dimensions.

De manière analogue, la figure 3.14 présente une visualisation de l'exploration SWEUQ.

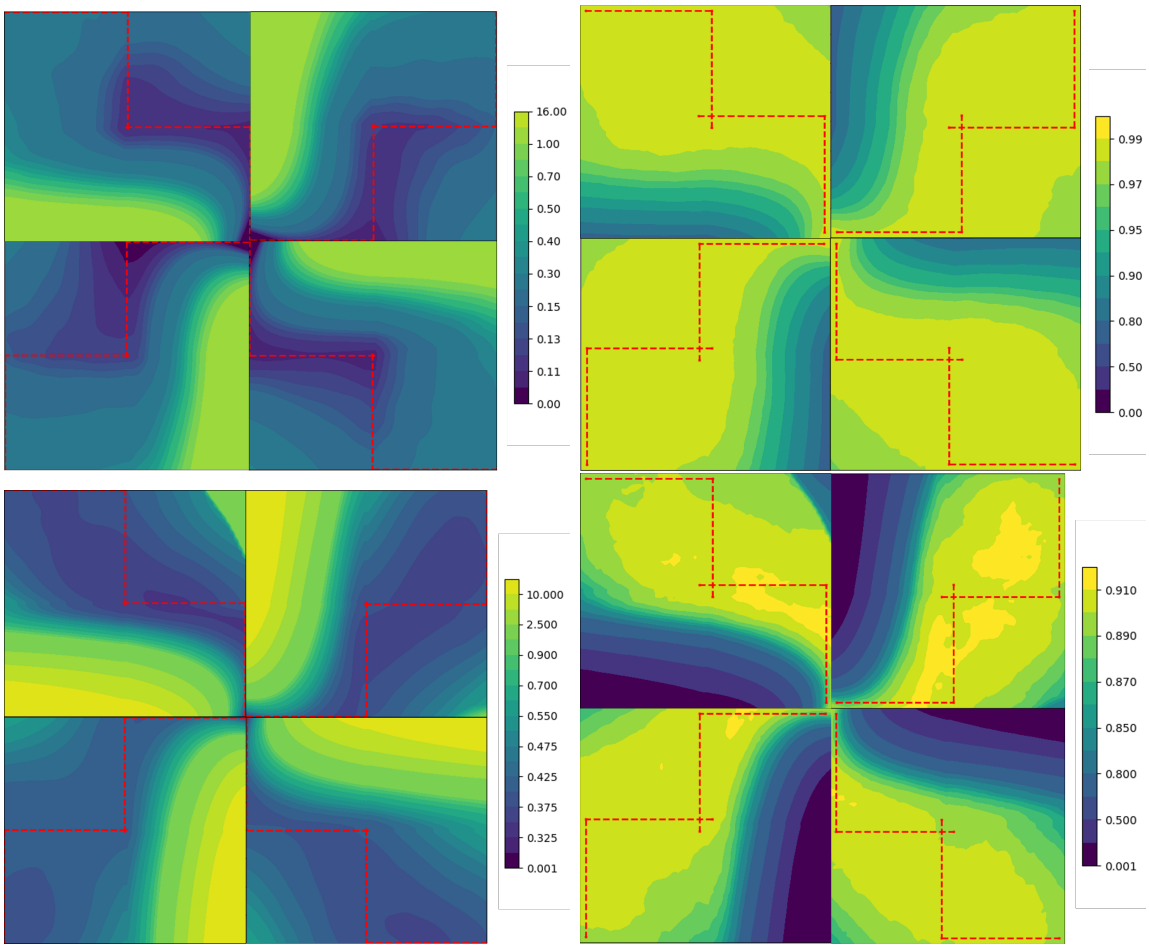


FIGURE 3.14 – Exploration SWEUQ dans 4 directions aléatoirement choisies pour le cas MNIST (**Haut**) et CIFAR-10 (**Bas**). La précision est montrée sur les figures de **Gauche** alors que le paysage de valeur de coût est montré sur les figures de **Droite**.

L'exploration SWEUQ permet d'effectuer des explorations dans plusieurs directions. Visuellement, on note que les différentes directions conduisent à des paysages locaux similaires.

Enfin, les tables 3.4 et 3.5 détaillent les résultats numériques obtenues à l'aide des deux variantes d'exploration WEUQ et SWEUQ ainsi que leur contre-partie "restartée" en reprenant l'ensemble des métriques utilisées lors de l'analyse comparative (cf. section 3.1).

TABLE 3.4 – Critères de performances pour les méthodes DE, WEUQ, SWEUQ, M-WEUQ et M-SWEUQ sur la tâche MNIST.

Méthode	NLL (↓)	Précision (↑)	ECE (↓)	MCE (↓)	Entropie$^-$ (↑)	Entropie$^+$ (↓)	AUROC (↑)	Nb épochs (↓)	
DE	0.02912	0.9893	0.001765	0.08638	0.9092	0.03593	0.9999	0.0009	6000
WEUQ	0.03872	0.9874	0.007795	0.2682	1.3357	0.08053	0.9999	0.0022	870
SWEUQ	0.04012	0.9886	0.001395	0.6397	0.8372	0.03664	0.9999	0.0033	870
M-WEUQ	0.03134	0.9883	0.003199	0.1497	1.0216	0.04509	0.9999	0.0026	1680
M-SWEUQ	0.03299	0.9883	0.001554	0.3967	0.8475	0.03050	0.9999	0.0020	1680

Pour le cas MNIST, les ensembles profonds permettent d'atteindre la meilleure performance avec une valeur de coût de 0.02912 contre 0.03134 pour la méthode M-WEUQ et une précision de 0.9893 contre 0.9886 pour la approche SWEUQ. En revanche, SWEUQ présente la meilleure valeur de diversité (0.0033) et la meilleure calibration avec une valeur ECE de 0.001395 contre 0.001765 pour les ensembles profonds. Son erreur de calibration maximale est quant à elle très élevée avec une valeur de 0.6397 contre 0.08638 pour le DE. Enfin, en termes d'entropie sur les mauvaises classifications, c'est notre méthode WEUQ qui atteint la meilleure performance avec une valeur de 1.3357. Nous notons, en particulier, que dans notre cas MNIST, les versions "simple-entraînement" (WEUQ et SWEUQ) atteignent des performances proches des versions "multi-entraînement" (DE, M-WEUQ et M-SWEUQ).

TABLE 3.5 – Critères de performances pour les méthodes DE, WEUQ, SWEUQ, M-WEUQ et M-SWEUQ sur la tâche CIFAR-10.

Méthode	NLL (↓)	Précision (↑)	ECE (↓)	MCE (↓)	Entropie$^-$ (↑)	Entropie$^+$ (↓)	AUROC (↑)	Divsersité (↑)	Nb épochs (↓)
DE	0.217	0.932	0.037	0.133	1.483	0.361	0.997	0.015	6000
WEUQ	0.287	0.904	0.006	0.833	1.373	0.297	0.995	0.011	870
SWEUQ	0.263	0.916	0.010	0.054	1.238	0.240	0.995	0.011	870
M-WEUQ	0.234	0.922	0.023	0.092	1.450	0.313	0.996	0.015	1680
M-SWEUQ	0.228	0.925	0.014	0.093	1.353	0.271	0.996	0.014	1680

Pour le cas CIFAR-10, c'est une nouvelle fois le DE qui présente les meilleures performances avec une valeur de coût de 0.217, une précision de 0.932, de diversité 0.015 et une entropie sur les mauvaises classifications de 1.483. Notre méthode SWEUQ présente la meilleure calibration avec la meilleure valeur de MCE (0.054) et une valeur de ECE de 0.010, juste derrière la méthode WEUQ présentant un ECE de 0.006 mais une valeur de MCE de de 0.833. Dans ce cas, les méthodes

"multi-entraînement" présentent des performances supérieures aux versions "simple-entraînement". Ce résultat vient confirmer les observations préalablement effectuées.

Enfin, nous présentons au sein de la figure 3.15, l'évolution de la précision et de la diversité en fonction du nombre d'éléments dans l'ensemble obtenu par un ensemble profond ainsi que nos différentes méthodes pour le cas CIFAR-10.

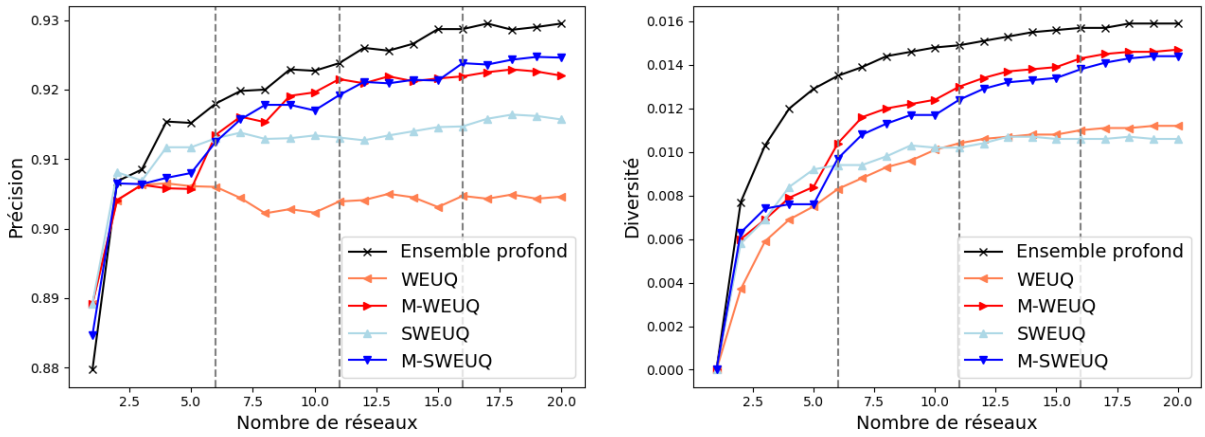


FIGURE 3.15 – Évolution de la précision (**Gauche**) et de la diversité (**Droite**) d'un ensemble en fonction du nombre de ses éléments pour les méthodes DE, WEUQ, SWEUQ, M-WEUQ et M-SWEUQ. Les droites verticales en pointillés indiquent un nouveau restart des méthodes M-WEUQ et M-SWEUQ.

Plusieurs points intéressants sont à noter :

— L'augmentation constante de la diversité de la méthode WEUQ ne permet pas une augmentation similaire de sa précision. La précision arrête sa progression après un ensemble constitué d'environ 5 éléments.

— On observe des cassures nettes de la précision et de la diversité à chaque restart des méthodes M-WEUQ et M-SWEUQ. L'exploration locale autour des différents modes permettent également une augmentation (moindre) des métriques.

— Les méthodes M-WEUQ et M-SWEUQ montrent des évolutions similaires alors que l'approche SWEUQ montre des résultats supérieurs à l'approche simple WEUQ.

— Les approches M-WEUQ et M-SWEUQ présentent des résultats proches de ceux présentés par les ensembles profonds.

Cette première approche d'exploration du paysage des paramètres est encourageante à la vue des performances précédemment exposées. Cependant, l'exploration *aléatoire* n'est pas satisfaisante à la fois en termes de performances (toujours en deçà de l'ensemble profond) et de fondements mathématiques. Ainsi, nos travaux futurs consisteront en la sélection d'une direction à privilégier en estimant la Hessienne du paysage local. La direction choisie sera alors alignée avec le(s) vecteur(s)

propre(s) présentant la (les) valeur(s) propre(s) la (les) plus faibles. Une seconde approche consisterait également à effectuer cette exploration sur un sous-espace des paramètres de faibles dimensions. Enfin, la dernière approche d'amélioration consisterait en le raffinement des paramètres en effectuant quelques époques en relâchant la condition de barrière nulle. Cette approche est motivée à la vue de la figure 3.16 où l'on peut observer que l'exploration WEUQ n'a pas permis de "rentrer" au sein de bassins contourés en blanc.

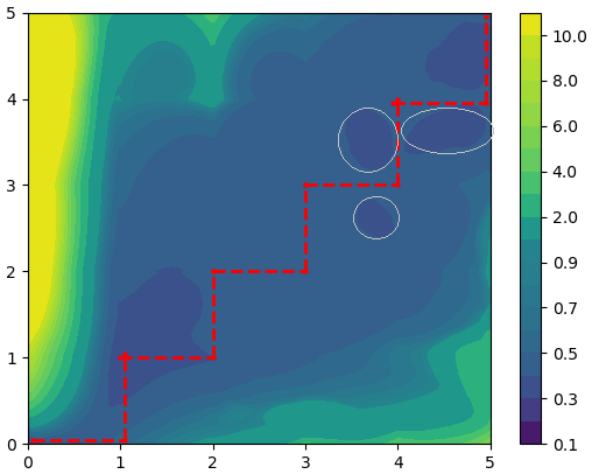


FIGURE 3.16 – Exploration WEUQ pour le cas CIFAR-10. Les zones de plus faibles valeurs de coût "manquées" par l'exploration sont entourées en **blanc**.

Dans ce chapitre, nous avons abordé la notion d'incertitude des paramètres des réseaux de neurones. Les ensembles profonds peuvent être considéré comme la méthode à l'état de l'art actuelle. Dans un cadre métrologique, elle revêt également une importance particulière car permet de quantifier l'incertitude de répétabilité, issue de la stochasticité des optimiseurs utilisés. Cependant, le nombre d'entraînement nécessaire peut constituer un frein à son utilisation. La méthode WEUQ explore la possibilité de les approcher localement en cherchant des jeux de paramètres en empruntant des tunnels de faible coût au sein du "paysage" des paramètres. L'obtention de premiers résultats prometteurs est encourageant afin d'améliorer l'algorithme présent. Enfin, concernant la quantification de l'incertitude prédictive issue de l'incertitude des paramètres, à la vue des résultats obtenus lors de notre étude comparative, les ensembles profonds constituent à l'heure actuelle le meilleur choix. Une étude de la convergence des différentes métriques est néanmoins nécessaire afin d'estimer le bon nombre de réseaux à inclure dans l'ensemble. Ce choix reposera nécessairement sur un compromis entre qualité de l'estimation et le budget temps disponible à l'inférence.

Chapitre 4

Application : Mesure automatisée de la distribution de tailles de particules de TiO_2 par imagerie MEB.

Dans de nombreux cas, l'estimation de la distribution de la taille d'une population de nanoparticules reste un défi majeur pour le développement industriel des nanomatériaux. Aujourd'hui, la microscopie électronique à balayage (MEB) est largement utilisée dans les laboratoires et les industries manufacturières et est considérée en métrologie comme une technique de référence capable de déterminer de manière fiable la taille, et la forme des nanoparticules. C'est une technique dite directe car elle repose sur des observations directes et le résultat de la mesure est directement traçable à l'unité SI de longueur, le mètre (Delvallée et al., 2015). Le principe de base de l'analyse d'image est : 1) l'identification des contours de chaque nanoparticule à l'aide d'outils automatiques ou manuels et 2) la détermination de la valeur de différents mesurandes (diamètre de surface équivalent, diamètre de Feret,...) à partir de mesures de surface ou de profil. Cependant, la fiabilité de la mesure est principalement liée à la performance de l'algorithme de segmentation utilisé pour identifier les bords des nanoparticules. La détermination de ce contour est complexe de par le phénomène naturel d'agglomération, qui tend à former des paquets de particules en 3D.

Ce chapitre présente notre méthodologie d'automatisation de la caractérisation de la taille des particules de dioxyde de titane (TiO_2) obtenues par MEB. Le dioxyde de titane sous forme nanoparticulaire est produit en très grandes quantités pour une utilisation intensive dans de nombreuses applications (alimentation, peinture, produits de construction, ...). Cependant, l'étude des propriétés dimensionnelles des particules de dioxyde de titane reste difficile en raison de leur forme non sphérique et de leur capacité d'agglomération (figure 4.1). En raison de cette complexité, la caractérisation de ce type de contenu n'est pas robuste et est souvent réalisée manuellement par des experts en nanométrologie. Cette tâche est excessivement longue et fastidieuse, d'où l'intérêt

croissant d'automatiser, même partiellement, la chaîne de traitement actuelle. La méthodologie présentée ici s'appuie sur des algorithmes d'apprentissage profond. En effet, la littérature dans ce domaine est florissante depuis quelques années, offrant de nouvelles perspectives d'amélioration dans de nombreux domaines. En particulier, ces outils récents ont prouvé leur efficacité dans de nombreuses tâches de vision par ordinateur, pour lesquelles ils ont, dans de nombreux cas, dépassé et, le cas échéant, au moins égalé les performances des algorithmes de l'état de l'art. Il convient toutefois de noter que la principale limite à la diffusion de ces algorithmes a longtemps été la taille de la base de données qu'il fallait créer pour faire fonctionner ces réseaux profonds. Cette limitation a toutefois été réduite avec le développement de l'apprentissage par transfert (Torrey and Shavlik, 2010), qui permet d'utiliser des réseaux pré-entraînés, puis de les entraîner spécifiquement sur notre tâche avec une base de données réduite.

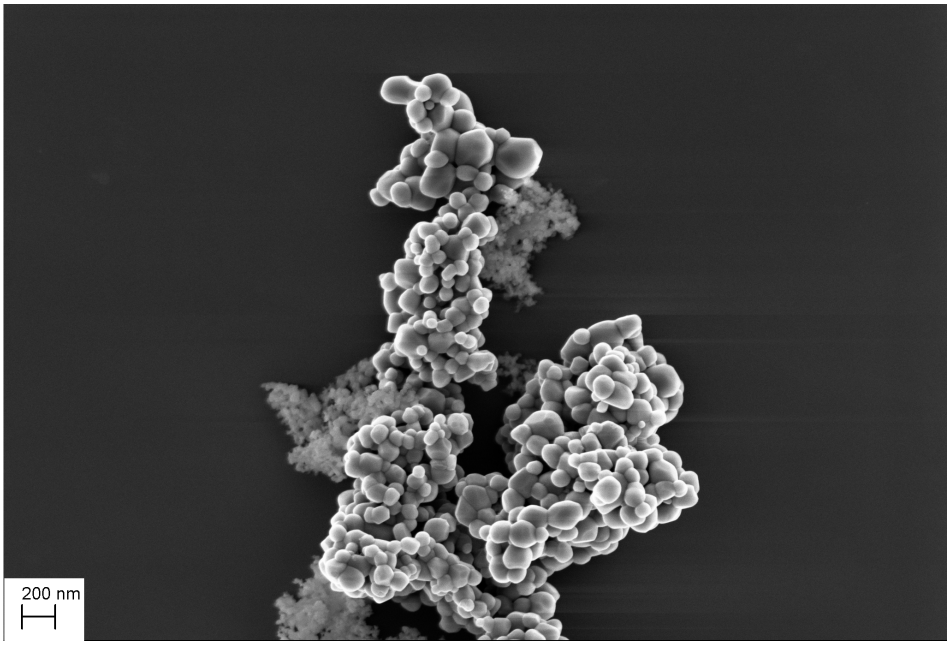


FIGURE 4.1 – Exemple d'une image MEB d'un mixture de particules de TiO_2 et de SiO_2.

La solution technologique finale doit respecter les spécifications industrielles exprimées par les experts en nanométrologie. Le pipeline de traitement doit être en mesure de :

1. Détecter les particules de TiO_2 présentes au sein des images MEB,
2. Segmenter les particules détectées (i.e. fournir un masque binaire discriminant l'intérieur et l'extérieur de la particule),
3. Classifier l'état d'agrégation de la particule (la mesure de taille ne peut s'effectuer que sur les particules intégralement imagées),
4. Calculer les mesurandes finaux de cette population de particules (diamètre équivalent, périmètre, aire, ferêt min, ferêt max, ...),
5. Quantifier les incertitudes des mesures finales.

La section 4.1 détaille les choix d'implémentation permettant de répondre à ce cahier des charges ainsi que les performances des taches de segmentation et de classification. La section 4.2 présente les choix de propagation des incertitudes au sein de ce pipeline de traitement, enfin, la section 4.3 présente la solution technologique encapsulant l'intégralité des développements théoriques et pratiques.

4.1 Pipeline de traitement

À partir d'images acquises par microscopie électronique à balayage, la chaîne de traitement menant à la caractérisation de la taille des particules de dioxyde de titane, c'est-à-dire au calcul du diamètre équivalent de chaque particule de l'image, comporte quatre tâches principales. La première est la détection des particules présentes au sein de l'image, suivie par la segmentation de chaque particule détectée (production d'un masque binaire), puis de la classification de l'état d'agglomération de chaque particule, et enfin du calcul du diamètre équivalent de chaque particule (complètement imagée) à partir du masque binaire. Même si chaque étape doit faire l'objet d'une attention particulière, la tâche principale reste évidemment la segmentation car les autres étapes du traitement dépendent directement de ces performances. L'algorithme de segmentation ainsi que les résultats associés sont détaillés dans la sous-section 4.1.1. La brique de classification est quant à elle détaillée au sein de la sous-section 4.1.2. La figure 4.2 présente le pipeline complet.

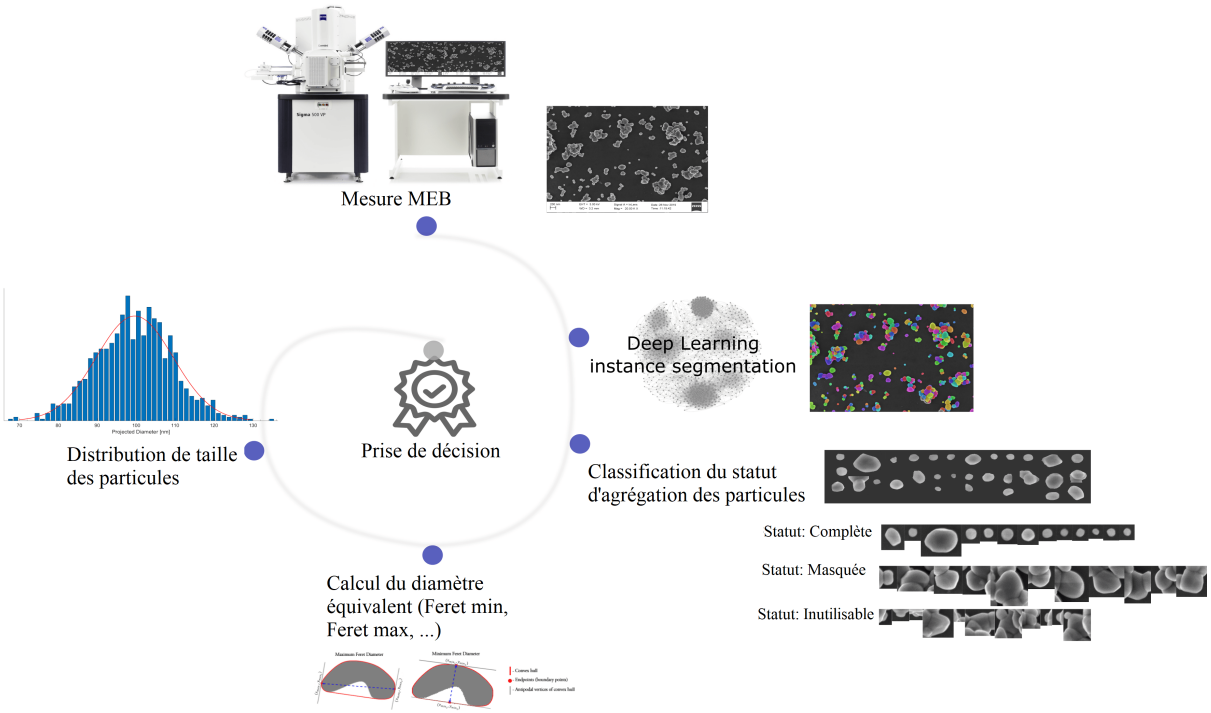


FIGURE 4.2 – Pipeline complet de traitement.

4.1.1 Détection et segmentation des particules de TiO_2

La phase de segmentation consiste en la détection de chaque particule présente dans l'image et la production d'un masque binaire spécifiant les pixels appartenant à la particule détectée. Dans un premier temps, nous présentons un état de l'art permettant la détection et la segmentation d'objets au sein d'une image.

État de l'art

La segmentation d'images est un défi de longue date pour la communauté de la vision assistée par ordinateur et de nombreuses approches différentes ont été développées. Ces problèmes sont essentiels dans le domaine de la nanométrologie où les études des particules s'effectue par imagerie microscopique telle que la microscopie électronique à balayage (MEB) ou la microscopie électronique à transmission (MET). Nous nous concentrons dans cette section sur les différentes stratégies d'automatisation de la segmentation d'images.

Une première approche de la segmentation d'images consiste en des techniques classiques de traitement d'images, parmi lesquelles nous pouvons trouver les méthodes de lignes de partage des eaux et ses variantes et adaptations (Wu et al., 2018; Tek et al., 2005; Baiyasi et al., 2020), les méthodes d'histogrammes (Wilson et al., 2020), les méthodes de "graph-cut" (Felzenszwalb and Huttenlocher, 2004), ou encore les méthodes de contours actifs (Nath et al., 2006). Ces techniques présentent l'avantage d'être efficaces, mais nécessitent un réglage rigoureux des paramètres pour qu'elles fonctionnent correctement sur des images spécifiques. En effet, avant d'appliquer l'un de ces algorithmes, il est nécessaire d'appliquer de nombreuses techniques de traitement d'image telles que le filtrage, le seuillage, les opérations morphologiques, ... Toutes ces techniques nécessitent un réglage spécifique pour atteindre un haut niveau de précision de segmentation sur une image donnée. Le principal inconvénient de ces méthodes est donc leur manque de généricité.

Une deuxième approche est basée sur le développement de flux de traitement d'images, mélangeant des méthodes générales de traitement d'images et des méthodes d'apprentissage automatique telles que l'algorithme EM (Park et al., 2012) ou l'algorithme K-means (Muneesawang and Sirisathitkul, 2015). Ces types de flux de travail montrent de fortes performances au prix d'un réglage des paramètres (manque de généricité) ou incluent des hypothèses préalables fortes, telles qu'un a priori sur les formes à détecter par exemple.

La dernière catégorie de méthodes repose sur des algorithmes d'apprentissage profond que nous pouvons séparer en deux sous-catégories distinctes : les méthodes de segmentation sémantique et les méthodes de segmentation d'instance.

Les méthodes de segmentation sémantique utilisent principalement des réseaux d'auto-encodeurs tels que UNet (Ronneberger et al., 2015). Ces algorithmes sont puissants, génériques et très précis. Ces méthodes proposent de recréer l'image d'entrée sous la forme d'une carte de segmentation où chaque pixel représente une étiquette de la classe de l'objet détecté. Alors que les méthodes sémantiques sont adaptées à de nombreux cas d'étude tels que les images médicales (Li et al., 2018; Weng et al., 2019), la caractérisation minérale (Liu et al., 2020) ou la météorologie (Jiao et al., 2020), elles ne peuvent pas être appliquées directement à nos images, où les particules de TiO_2 ont tendance à s'agglomérer. Ces techniques sont néanmoins utilisables dans notre cas lorsque l'image d'entrée ne présente qu'une unique particule à segmenter. Ce sera d'ailleurs la solution finale adoptée.

Les méthodes de segmentation par instance semblent être adaptées à notre étude de cas et à la spécificité des images d'agglomérats de TiO_2 acquises par MEB. Ces méthodes offrent une certaine généricité, une détection séparée de chaque instance et une certaine robustesse. Cependant, ces méthodes nécessitent une grande quantité de données d'entraînement, qui peuvent être difficiles à obtenir. En effet, certaines images MEB de particules de TiO_2 peuvent présenter plus de 500 instances différentes qui doivent être segmentées manuellement. Le principal algorithme utilisé pour la segmentation des instances est un algorithme développé par l'équipe de recherche de Facebook (FAIR) appelé "Mask-RCNN" (He et al., 2017). Il a été testé dans divers domaines tels que l'analyse d'images médicales (détection de tumeurs cérébrales (Zhang et al., 2019), détection de noyaux en microscopie (Johnson, 2018), détection de nodules pulmonaires (Liu et al., 2018)), l'analyse d'images satellites (Zhao et al., 2018), l'imagerie aérienne à très haute résolution spatiale (Zhang et al., 2018) ou l'astronomie (Burke et al., 2019) pour ne citer que quelques exemples. Cet algorithme a également montré des résultats prometteurs sur des images STM de nanoparticules (Okunev et al., 2020). Cependant, et comme spécifié précédemment, la qualité de la segmentation et a fortiori la quantification de l'incertitude associée doit être très élevée. Un des avantages de l'algorithme Mask-RCNN est la détection d'objets et leur segmentation directement sur l'image initiale en une seule étape. Dans notre cas, cette caractéristique rend complexe la quantification d'incertitudes de par la dimension des images MEB d'entrée (2048×1536 pixels, soit 3145728 pixels).

Afin de palier à ce problème, nous avons opté pour une phase de segmentation composée de deux étapes distinctes :

— **Phase de détection** : la première étape consiste en la détection des particules présente au sein de l'image MEB en fournissant les coordonnées de la boite englobante de la particule considérée. Cette première étape est effectuée à l'aide d'un Faster-RCNN (Ren et al., 2015) détaillée à la sous-section 4.1.1. Le Faster-RCNN est une version antérieur du Mask-RCNN, et permet "uniquement" la détection d'objets au sein d'une image.

— **Phase de segmentation** : la seconde phase consiste en la régression du masque binaire de segmentation de la particule située au centre de la boite englobante détectée lors de la première phase. Cette seconde étape est effectuée à l'aide d'un auto-encoder UNet, détaillé à la sous-section 4.1.1

Ce choix de conception permet d'effectuer une propagation d'incertitude sur des images d'entrée de taille réduite à 64×64 pixels.

Remarque 13.

Monchot et al. (2021) présentent une étude poussée de la tâche de segmentation effectuée à l'aide du Mask-RCNN.

Détection des particules par Faster-RCNN

Le Mask-RCNN est devenu un standard dans la communauté de l'apprentissage profond, étant à la fois générique et efficace. Il peut être considéré comme un algorithme en deux étapes : la première étape génère des propositions de régions d'intérets (ROI) tandis que la seconde prédit la classe de l'objet, les coordonnées de sa boîte englobante ainsi que le masque binaire de segmentation associé. Dans notre cas, nous ne désirons pas produire directement le masque binaire pour chaque ROI. Nous utilisons donc une version antérieur du Mask-RCNN, appelée Faster-RCNN et consiste en un Mask-RCNN privée de la génération du masque binaire. Ainsi, la partie amont de cet algorithme est composée d'un réseau d'apprentissage résiduel (ResNet) (He et al., 2016), couplé à un réseau de pyramide de caractéristiques (FPN) (Lin et al., 2017) pour améliorer la représentation des objets à plusieurs échelles et, dans le même temps, améliorer particulièrement la précision. Un réseau spécifique appelé réseau de proposition de région (RPN), comme son nom l'indique, propose des régions d'intérêt en utilisant une technique de fenêtre coulissante (appelée ancres) directement sur les représentations caractéristiques produites par la partie amont. Enfin, la partie aval de cette architecture, consiste en un pipeline plus courant (composé d'une succession de couches convolutives et de fonctions d'activations), et permet de prédire la classe de l'objet ainsi que d'affiner les coordonnées de la région d'intérêt englobant l'objet. La figure 4.3 (adaptée de (Zhang et al., 2020b)) présente un schéma illustratif du Faster-RCNN.

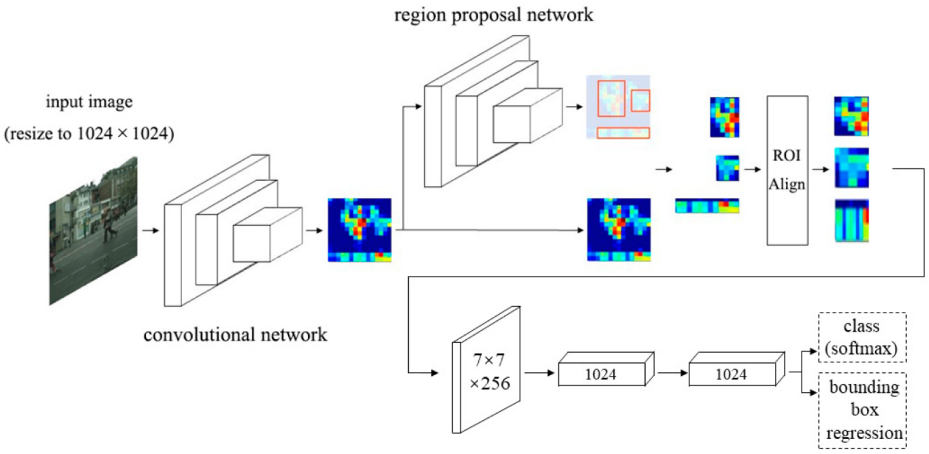


FIGURE 4.3 – Architecture du Faster-RCNN.

Segmentation des particules par UNet

L'architecture UNet est une architecture de segmentation sémantique et fut développée pour la segmentation d'images médicales. L'architecture suit le schéma d'un auto-encoder classique, à savoir une phase d'encodage à l'aide d'une succession de convolutions réduisant la résolution spatiale de l'image et augmentant le nombre de filtres. La fin de cette phase d'encodage est appelée le "code" et contient une représentation paramétrique en faible dimension de l'image d'entrée. Ce code entre alors dans la phase de décodage, consistant en une succession de convolution effectuant les étapes en miroir de la phase d'encodage, à savoir augmenter la résolution spatiale tout en diminuant le nombre de filtres. Enfin, chaque niveau d'encodage et de décodage (ayant donc la même dimension) sont concaténées permettant une nouvelle fois une représentation finale de l'image d'entrée à plusieurs échelles. La figure 4.4 présente cette architecture sous forme de schéma.

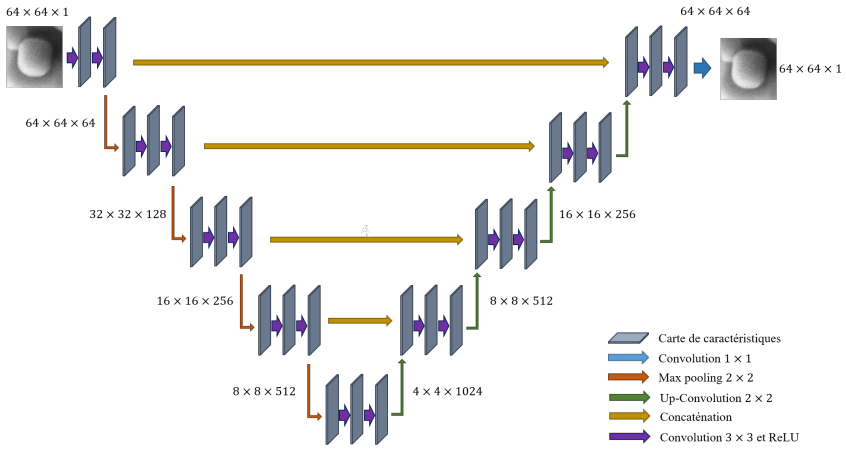


FIGURE 4.4 – Architecture UNet. Les tailles des entrées/sorties importantes sont affichées sous la forme *hauteur* × *largeur* × *profondeur*.

Création de la base de données

Faster-RCNN L'étape clé de tout développement utilisant l'apprentissage statistique est la création d'une base de données spécifique à partir de laquelle le réseau de neurones apprendra à faire des prédictions sur de nouvelles données, jamais traitées par le réseau, d'où la décomposition systématique des données d'entrées en une base de données d'apprentissage, de test et de validation. Si l'algorithme Faster-RCNN est très puissant pour les tâches de vision par ordinateur, il n'en est pas moins gourmand en ressources, c'est-à-dire que son apprentissage nécessite une très grande base de données d'images (plusieurs centaines de milliers, voire plusieurs millions d'échantillons). Dans notre cas, il est tout simplement impensable de segmenter manuellement autant d'images.

La base de données construite pour effectuer la segmentation est actuellement composée de 77 images segmentées manuellement par quatre opérateurs formés par des experts en nanométrologie représentant 11 000 particules de TiO_2. Ces mêmes experts ont ensuite validé les masques ainsi produits avant leur incorporation dans la base de données. L'entrée de l'algorithme est un tenseur de taille $W \times L \times C \times N$: W et L correspondent à la largeur et à la longueur en nombre de pixels de chaque image SEM ($W = 2048$, $L = 1536$), C correspond au nombre de canaux de l'image ($C = 1$, la mesure SEM renvoie une image en niveaux de gris) et enfin N représente le nombre d'échantillons disponibles ($N = 77$). La figure 4.5 montre une image MEB de particules agglomérées de dioxyde de titane et ses agglomérats annotés (segmentation manuelle).

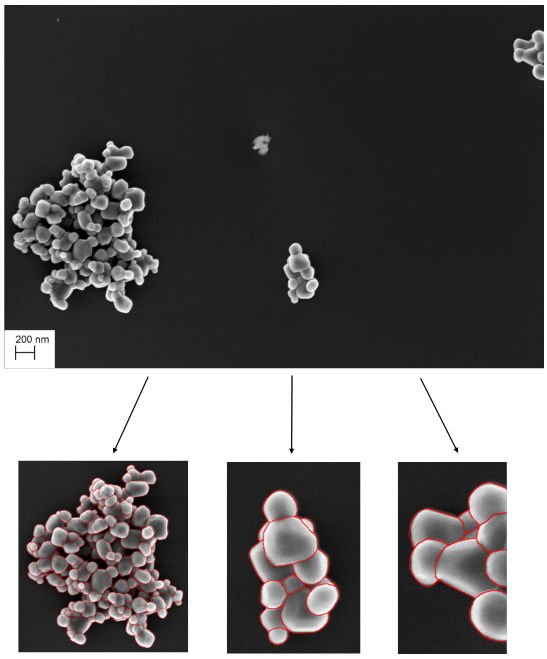


FIGURE 4.5 – Image MEB (**haut**) et ses agrégats annotés (**bas**).

Ce nombre de données étant faible, nous proposons de créer de "fausses" images à l'aide d'une technique adaptée à notre tâche et détaillé en profondeur en (Monchot et al., 2021). L'annexe K

détaille également les grandes étapes de cette méthodologie. La figure 4.6 présente un exemple d'image simulée. Notre ensemble d'entraînement final est constitué ainsi de 699 images, 77 images "réelles" et 622 "fausses" images.

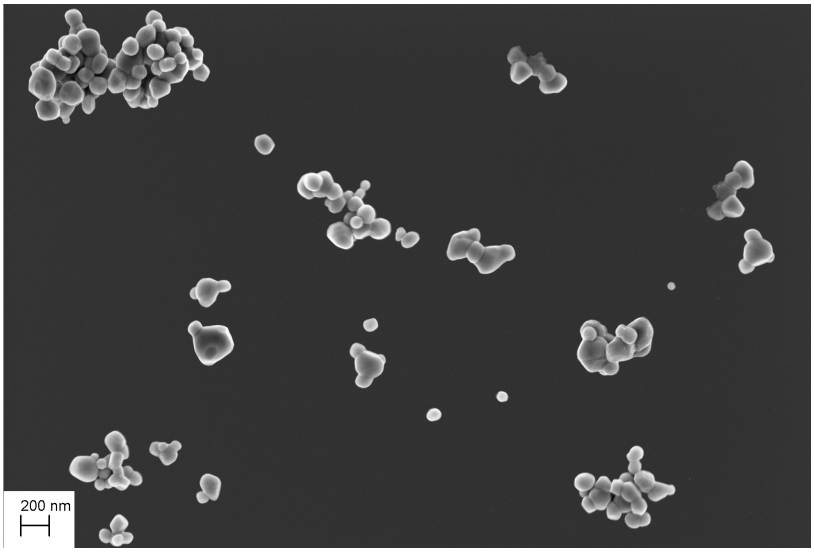


FIGURE 4.6 – Simulated SEM image of TiO_2 particles.

Cette technique d'augmentation ne permettant toujours pas d'atteindre le nombre de données suffisant pour entraîner correctement un algorithme tel que le Faster-RCNN, nous nous reposons alors sur le transfert d'apprentissage ainsi que les méthodes classiques d'augmentation de données.

UNet La base d'entrainement de UNet est donc composée de l'intégralité des particules présentes dans les 77 images originales. Elle est ainsi constituée de 11 000 imagettes contenant en son centre la particule de TiO_2 à segmenter. Pour cet algorithme, nous nous reposons uniquement sur les techniques classiques d'augmentation de données et cités précédemment.

Performances

Cette sous-section détaille les performances obtenues. Ces résultats font suites à la phase d'entraînement qui fut réalisé avec les hyperparamètres suivants :

L'ensemble de test est composé de 19 images représentant 3 741 particules de TiO_2. Ces 19 images représentent les différents types de configurations que nous pouvons rencontrer dans notre domaine d'étude. Les particules de TiO_2 peuvent être présentes sous forme d'agrégat, dispersées ou en présence d'une matrice d'autres types de particules (par exemple, avec une matrice de particules de SiO_2). La figure 4.7 présente trois images de test représentant les différents types de configurations de particules.

Hyperparamètres	Faster-RCNN	UNet
Optimiseur	SGD	SGD
Momentum	0.9	0.0
Pas initial	0.001	0.01
Pas final	0.0001	0.0001
Régularisation L_2 (Krogh and Hertz, 1992)	0,0001	0.0
Taille de batch	1	64
Taille des ancres (en pixels)	[8, 16, 32, 64, 96]	-/-
Nombre d'ancres	1024	-/-
Distance entre les ancres (en pixels)	1	-/-

TABLE 4.1 – Valeur des hyperparamètres pendant l'entraînement.

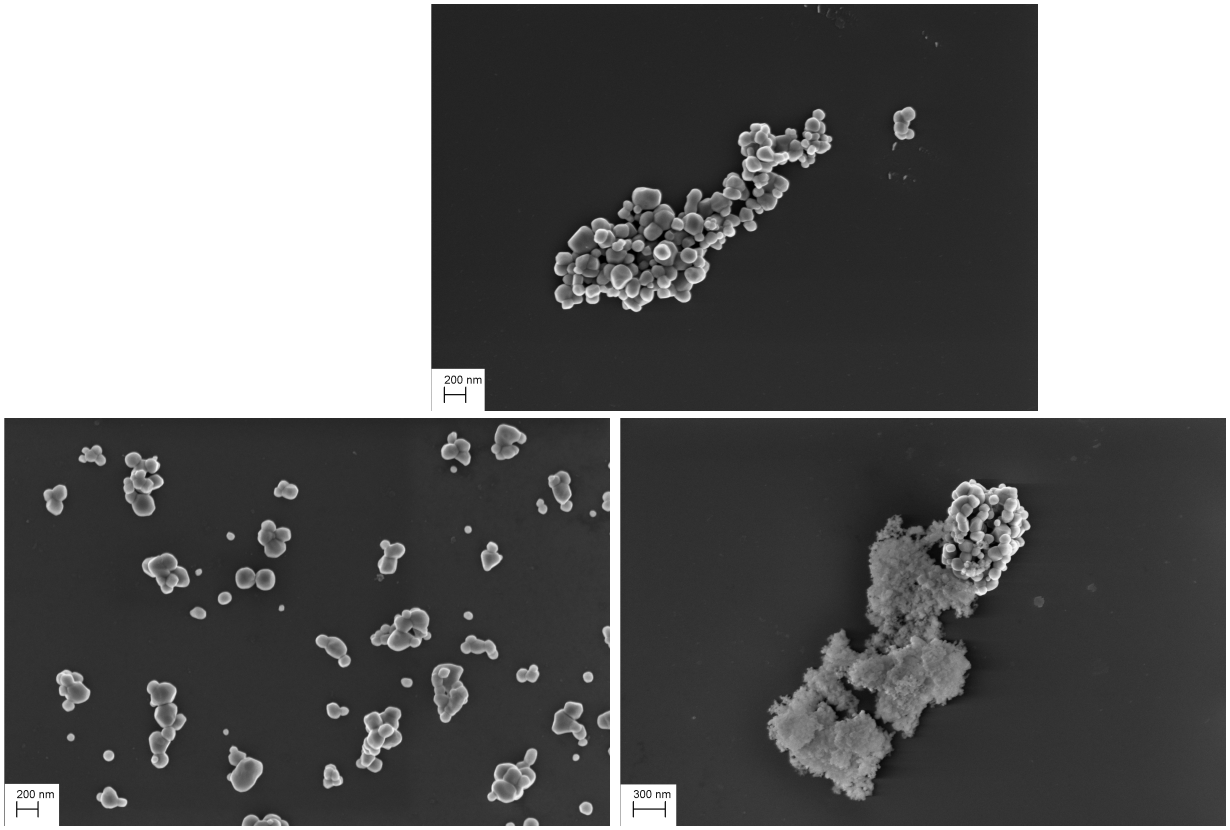


FIGURE 4.7 – Test SEM images, agglomerate TiO_2 particles (**top figure**), scattered TiO_2 particles (**bottom left figure**) and with a mixture of SiO_2 and TiO_2 particles (**bottom right figure**).

Les différentes images montrent également une grande diversité dans la disposition des particules (figure 4.8) :

— **Isolée** : la particule est complètement imagée et située en dehors d'un agrégat,

— **Complète** : la particule est complètement imagée et située près ou dans un agrégat,

— **Touch-complète** : la particule est complètement imagée mais imbriquée dans une autre particule (entre l'état "complète" et l'état "masquée"),

— **Masquée** : la particule est en partie cachée par une autre particule,
— **Inutilisable** : la particule est masquée par une autre particule dont la surface visible est très réduite (moins de 40% de sa surface est imagée et, par conséquent, ne constitue pas un intérêt pour notre objectif de mesure de la distribution de la taille).

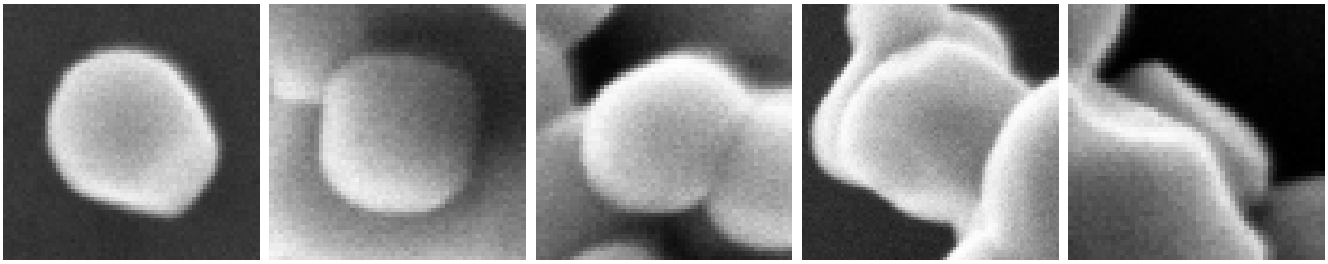


FIGURE 4.8 – Différentes configurations de particules de TiO_2 rencontrées : isolée, complète, touch-complète, masquée et inutilisable (de **gauche** à **droite**).

Les résultats suivants seront affichés en fonction de ces configurations.

Performances de détection du Faster-RCNN Le tableau 4.2 résume le nombre de particules détectées dans l'ensemble de test par rapport au nombre de particules dans la segmentation de référence (créée par segmentation manuelle et validée par des experts).

TABLE 4.2 – Nombre de particules détectées sur le jeu de test en fonction de leur état d'agrégation.

Particules détectées	**Toutes**	**Complète**	**Touch-Complète**	**Masquée**	**Inutilisable**
Référence	3741	341	515	1302	1583
Détectées	3135	339	495	1253	1048
Pourcentage	83.80	99.41	96.11	96.23	66.2

Dans l'ensemble, l'algorithme Faster-RCNN a pu détecter environ 84% des particules. Cette valeur atteint 97% lorsque nous ne conservons que les particules utiles (c'est-à-dire non classées comme "inutilisables"). La performance de détection est pilotée par l'hyperparamètre contrôlant le seuil de détection et peut donc être modifiée. La valeur du seuil est fixée de manière à détecter presque toutes les instances de particules utiles sans générer de faux positifs (en particulier dans les images présentant une matrice).

Pour compléter l'évaluation des performances de détection de notre réseau, nous calculons la précision moyenne (Everingham et al., 2010) (mAP). La mAP est une mesure populaire qui permet d'évaluer les performances de détection d'un algorithme en calculant la courbe de précision/rappel pour différents seuils d'IoU. Le tableau 4.3 présente les résultats de cette évaluation.

TABLE 4.3 – Score AP pour l'évaluation du Faster-RCNN sur les images MEB de particules de TiO_2.

Dice	**mAP**	**AP_{50}**	**AP_{75}**	**AP_{90}**
Score	60.6	84.6	71.0	21.5

La tâche de détection sur l'ensemble du test (19 images) a été réalisée en 110 s, ce qui équivaut à $5,79$ s par image ou $0,035$ s par particule détectée. À titre de comparaison, une segmentation effectuée manuellement prend environ 15 à 30 secondes par particule. Si on l'étend à l'ensemble du jeu d'essai, elle prend au moins $15 \times 3741 = 56115$ s (∼15,6 h).

Résultats de segmentation de UNet Nous souhaitons maintenant détailler les performances de segmentation de l'algorithme UNet en mesurant le coefficient de Sørensen–Dice sur les particules détectées. Ces résultats sont présentés ci-dessous (table 4.4) en détaillant les statistiques du coefficient de Sørensen–Dice sur les particules détectées en fonction des catégories "toutes", "complète" et "utilisables". Les particules "utilisables" sont toutes les particules à l'expection des particules "inutilisables".

TABLE 4.4 – Coefficient de Sørensen–Dice sur les particules de TiO_2.

Dice	**Moyenne**	**STD**
Toutes	0.905	0.040
Complète	0.920	0.024
Utilisable	0.912	0.026
Inutilisable	0.903	0.046

La performance globale du réseau est satisfaisante avec un coefficient de Sørensen–Dice global de $0,905$ calculée sur l'ensemble des particules détectées. Le coefficient atteint 0.912 lorsque l'on observe uniquement les particules utilisables et $0,920$ pour les particules "complètes". Cela peut s'expliquer en examinant la définition des particules classées comme "inutilisables". Nous rappelons qu'il s'agit de particules masquées, dont la zone visible est très petite (moins de 40% de la surface de la particule), ce qui les rend plus difficiles à segmenter.

Enfin, on peut se demander comment le coefficient de Sørensen–Dice est traduit en termes de mesurandes finaux que sont l'aire, le diamètre de surface équivalent, le diamètre de Feret min (Feret, 1931), le diamètre de Feret max, et le périmètre. L'histogramme de la figure 4.9 montre la distribution résiduelle du diamètre de surface équivalente des particules détectées classées comme utiles en pourcentage.

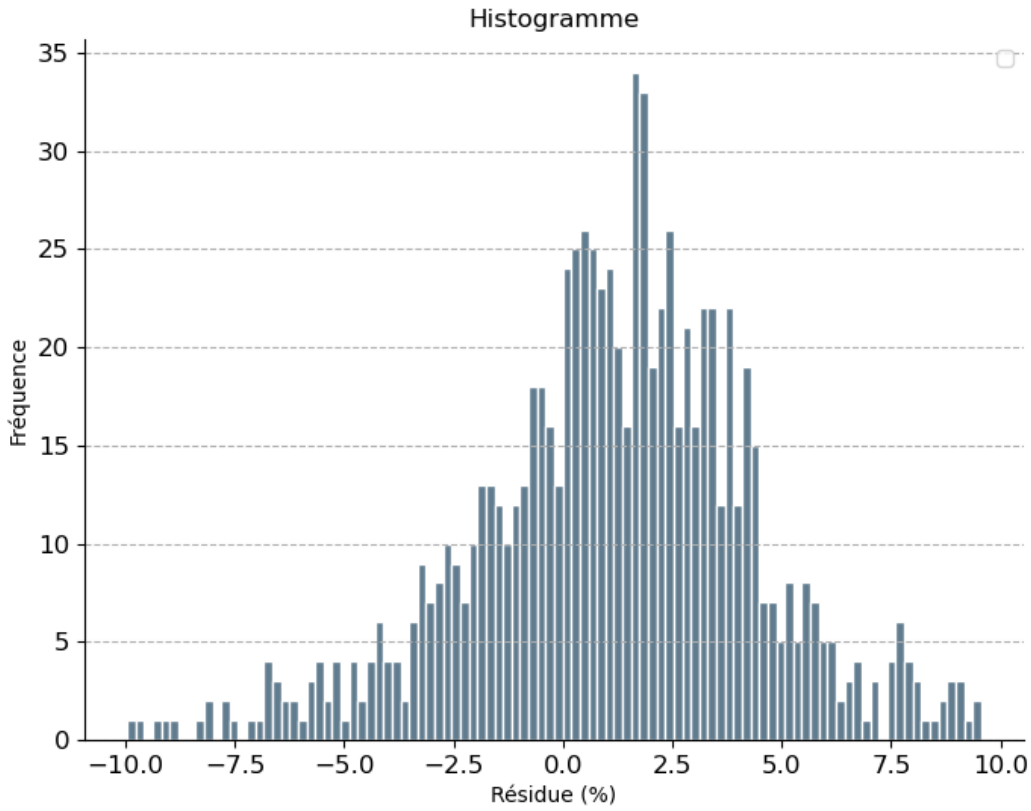


FIGURE 4.9 – Distribution résiduelle du diamètre de surface équivalente des particules détectées classées comme utiles en pourcentage.

Concernant le mesurande du diamètre équivalent projeté, la distribution montre une moyenne de 1.1 et un écart-type de 3.2. En outre, plus de 95% des mesures présentent une erreur inférieure à 7% et 50% des mesures présentent une erreur inférieure à 2.5% (par rapport à la référence manuelle produite par les opérateurs). La distribution résiduelle présente un biais faible mais non négligeable. Ce biais peut provenir du réseau lui-même, ou d'un biais introduit dans les annotations de référence.

4.1.2 Classification de l'état d'agrégation des particules de TiO_2

La brique de classification permet d'estimer la classe d'appartenance de l'ensemble des particules détectées au sein des images MEB. Seules les particules entièrement imagées (isolées, complètes et touch-complètes) doivent être mesurées, tandis que les particules masquées pourraient être soumises à un post-traitement afin d'estimer leur taille réelle (Coquelin et al., 2019).

L'objectif de cette brique est alors de classifier l'ensemble des particules détectées au sein de ces 5 classes. Pour cela, nous avons mis en place un réseau de neurone classique reposant sur l'architecture VGG (Simonyan and Zisserman, 2014).

L'architecture VGG

Les VGG (Simonyan and Zisserman, 2014) sont des réseaux de neurones convolutifs classiques, très utilisés en pratique pour leur simplicité d'implémentation et leurs performances. La figure 4.10 présente le schéma du réseau VGG utilisé pour notre tâche de classification. La philosophie de cette architecture consiste en l'enchaînement de blocs convolutifs munis de ReLU comme fonction d'activation, chacun séparé par une couche de "max pooling" permettant de réduire la résolution spatiale. Dans une architecture VGG, lorsque la résolution spatiale est réduite par 2 (en hauteur et en largeur, soit 4 fois moins de "pixels"), on augmente par 2 la profondeur de l'image (2 fois plus de filtres).

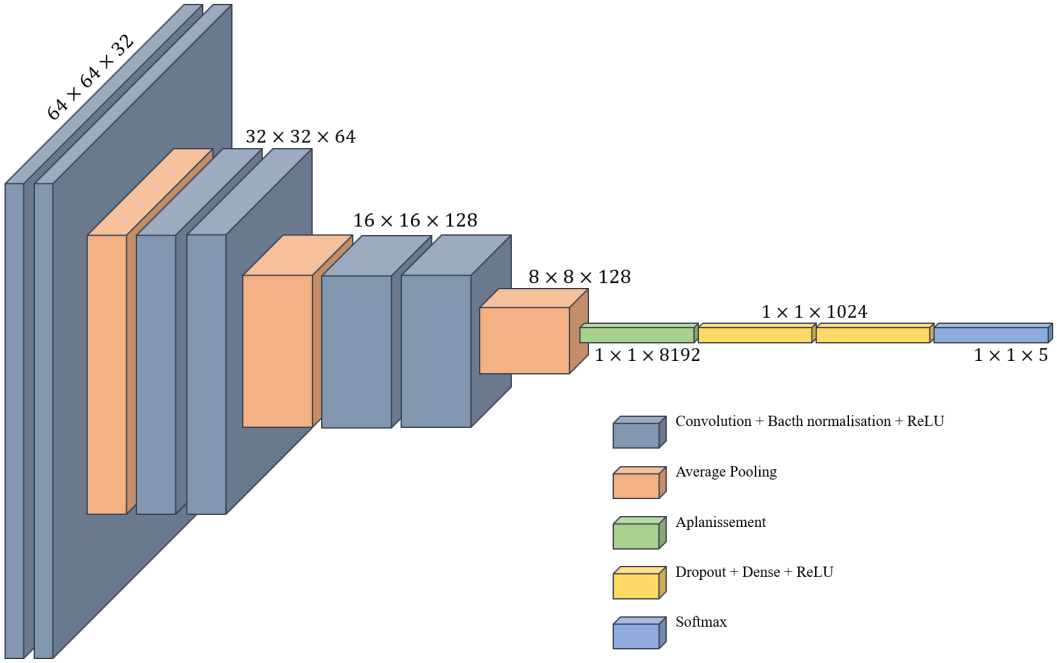


FIGURE 4.10 – Architecture du VGG utilisée.

Une architecture VGG présente en général deux couches entièrement connectées faisant office de classifieur. Le principal inconvénient des architectures VGG est leur nombre de paramètres qui peut être important.

Base de données et entraînement

Les images d'entrée sont composées de deux canaux : le premier contient l'image de la particule en nuance de gris et le deuxième contient le masque de segmentation produit lors de l'étape précédente. Nous avons alors ajouté une nouvelle classe de classification, nommée "À jeter", correspondant aux particules mal segmentées.

La table 4.5 présente le détail des bases de données d'entraînement et de test.

Classe	**Train**	**Test**
Isolée	567	54
Complète	1129	98
Touch-complete	2329	250
Masquée	4611	478
Inutilisable	1747	183
À jeter	2789	320
Total	13172	1383

TABLE 4.5 – Détail des bases de données d'entraînement et de test par classe.

La phase d'entraînement a été effectuée sur 300 époques, utilisant l'optimiseur classique SGD, afin de nous placer dans un cadre correspondant à notre cadre expérimental d'estimation des incertitudes des paramètres présenté précédemment. Nous utilisons les méthodes classiques d'augmentation de données que sont la rotation, les retournements horizontaux et verticaux, ...

Performances

Sur le jeu de test détaillé précédemment, notre algorithme atteint un score de bonne classification de 76.2% (soit 23.8% de mauvaises classifications). C'est un score satisfaisant de classification, étant du même ordre de grandeur que la performance estimée sur un opérateur humain. En effet, et dans certains cas, il peut être très complexe de différentier les particules dites "masquées" des particules dites "inutilisables", ainsi que les particules "touch-complètes" qui sont à la fois proches des particules "complètes" et des particules "masquées". Le tableau 4.6 présente le nombre de bonnes et mauvaises classifications en fonction de la valeur de la prédiction.

TABLE 4.6 – Nombre de bonnes classifications en fonction de la classe de la particule et du score de prédiction (les pourcentages sont en ()). Nb particules précise le nombre de particules restantes après seuillage.

Classe	**Seuil**						
	0.0	0.5	0.6	0.7	0.8	0.9	0.95
Complète	77 (78.6)	71 (78.0)	62 (84.9)	56 (84.5)	50 (89.3)	41 (95.3)	33 (97.1)
A jeter	251 (78.4)	242 (81.8)	234 (83.9)	221 (86.0)	203 (91.0)	181 (95.8)	160 (98.8)
Isolée	52 (96.3)	52 (96.3)	52 (96.3)	52 (96.3)	52 (96.3)	52 (96.3)	52 (96.3)
Masquée	376 (78.7)	359 (75.1)	324 (83.7)	273 (85.0)	224 (89.6)	163 (95.9)	97 (0.97)
Touch Complete	177 (70.8)	165 (71.7)	147 (76.2)	118 (80.3)	86 (80.4)	37 (80.4)	19 (86.4)
Inutilisable	122 (66.7)	117 (70.5)	98 (76.0)	85 (77.3)	71 (84.5)	44 (91.7)	22 (95.7)
Total	1055 (76.3)	1006 (78.3)	917 (82.3)	805 (84.6)	686 (88.7)	518 (94.4)	383 (97.2)
Nb particules	1383 (100.0)	1284 (92.8)	1114 (80.5)	952 (68.8)	773 (55.9)	549 (39.7)	394 (28.5)

On observe empiriquement qu'il est possible d'augmenter la "confiance" en la prédiction en seuillant les scores de prédictions. Ainsi, lorsque l'on garde uniquement les prédictions ayant un score supérieur à 0.80, on atteint un pourcentage de bonne classification de 88.7%. Ce seuillage nous prive potentiellement de plus de 40% des particules détectées et segmentées à la tâche précédente.

La table 4.7 présente la matrice de confusion correspondante :

TABLE 4.7 – Matrice de confusion de la tâche de classification par VGG

Vraie classe	**Classe prédite**					
	Complète	A jeter	Isolée	Masquée	Touch_complete	Inutilisable
Complète	77	1	0	1	19	0
A jeter	2	251	1	34	9	23
Isolée	0	2	52	0	0	0
Masquée	4	15	0	376	59	24
Touch_complete	30	8	0	32	177	3
Inutilisable	0	14	0	42	5	122

La classe touch-complète est la classe présentant le plus fort risque de confusion avec d'autres classes (en l'occurrence, les complètes et les masquées, pour les raisons précédemment présentées). La quantification d'incertitudes dans ce cas prend alors tout son sens. En effet, le post-traitement des particules masquées et des particules touch-complètes étant fondamentalement différents (du fait qu'une particule soit masquée ou totalement imagée), il est primordial de s'assurer de la bonne classification des particules avec forte probabilité (95%).

On peut néanmoins calculer la nouvelle performance en réduisant la tâche en deux classes : les particules totalement imagées ("isolée", "complète", et "touch-complète") et les autres. Ce nouveau score atteint 90.8% de bonnes classifications.

Remarque 14.

L'espace défini par les images dites "touch-complete" est une zone de l'espace d'apprentissage où l'incertitude aléatoire est élevée. En effet, les frontières avec les classes "complète" et "masquée" semblent très flou.

Au contraire, les particules "isolées" semblent avoir une incertitude aléatoire très faible.

4.2 Propagation des incertitudes

Dans cette section, nous mettons en oeuvre sur notre cas d'application, nos choix méthodologiques afin de quantifier l'incertitude prédictive d'un réseau de neurones. Pour cela, nous nous munissons

d'un ensemble d'images de test. Ces images de test appartiennent à la même distribution que la base de donnée d'entraînement. En effet, ces images ont été acquises durant la même acquisition que les images constituant la base de données d'entraînement. La section 4.2.1 présente alors un exemple de propagation des incertitudes d'entrée ainsi que leur modélisation au sein du réseau de segmentation sémantique UNet. Puis, la section 4.2.2 présente la quantification de l'incertitude des paramètres à l'aide des ensembles profonds.

4.2.1 Incertitude des entrées : propagation d'un bruit de flou par WGMprop

L'incertitude des entrées de notre pipeline provient de l'acquisition des données par microscopie MEB. La microscopie MEB consiste à envoyer un faisceau d'électrons avec un certain angle d'incidence sur l'échantillon à caractériser. L'angle d'incidence du faisceau engendre alors des frontières des particules diffuses, comme présenté au sein de la figure 4.11.

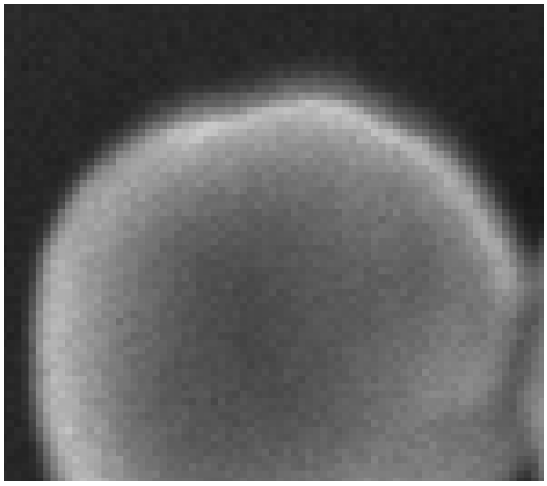


FIGURE 4.11 – Particule de TiO_2 acquise par microscopie MEB.

Afin de modéliser ce phénomène, nous appliquons notre méthodologie WGMprop présentée au sein du chapitre 2, muni d'un bruit de flou de faible intensité ainsi que d'un seuil $T_{split} = 0.0001$, d'un niveau de division $n = 7$ et d'un sous-espace actif de dimension maximale $r = 30$, nous propageons les distributions d'entrées au sein de notre brique de classification afin d'obtenir la densité de probabilité du mesurande de sortie, ici le diamètre équivalent projeté. La figure 4.12 représente cette propagation.

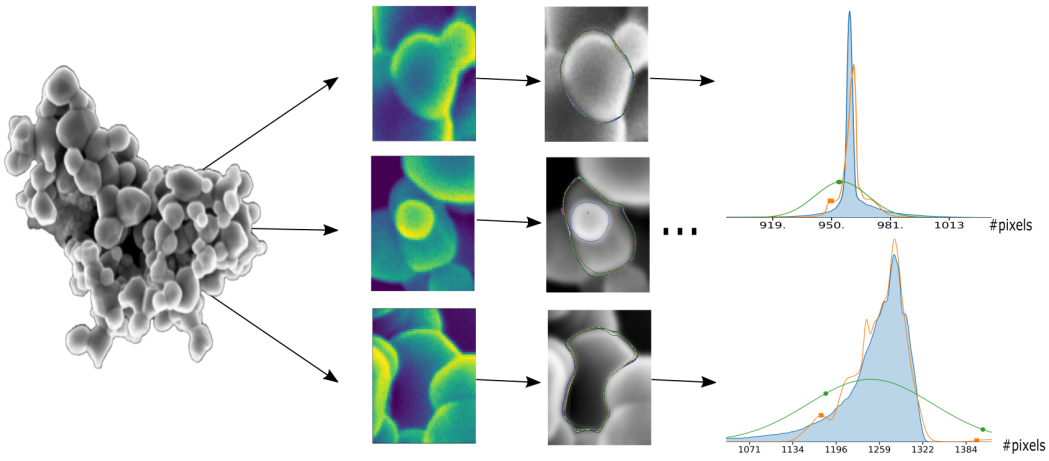


FIGURE 4.12 – Propagation par WGMprop d'un bruit de flou d'intensité faible au sein de l'auto-encoder de segmentation.

La généricité de notre approche WGMprop, et son implémentation "boite noire", nous permet de propager facilement les incertitudes des entrées de bout en bout. Le faible nombre d'hyperparamètres permet également une utilisation pratique facile à prendre en main pour les experts en nanométrologie.

4.2.2 Incertitude des paramètres : sélection du nombre de prédicteurs dans l'ensemble profond

Conformément aux recommandations effectuées au sein du chapitre 3, nous quantifions l'incertitude des paramètres de nos réseaux à l'aide de la méthode d'ensemble profond, contenant jusqu'à 50 réseaux indépendants. Nous présentons, ici, les (premiers) résultats obtenus sur notre brique de classification. La figure 4.13 présente l'évolution de la valeur de coût ainsi que la précision des ensembles en fonction du nombre d'éléments constitutifs.

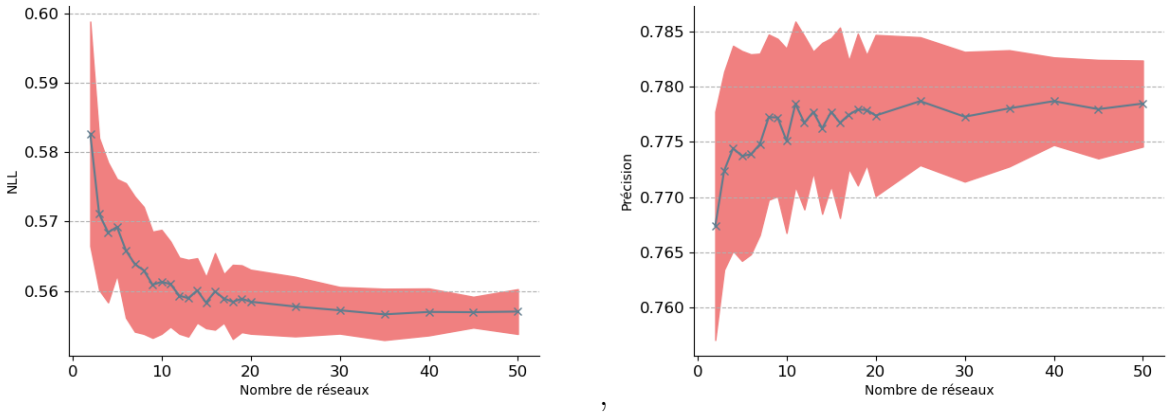


FIGURE 4.13 – Évolution de la valeur de coût (**à gauche**) et de la précision (**à droite**) des ensembles en fonction du nombre d'éléments constitutifs

La valeur de coût ainsi que la précision atteignent un régime stationnaire pour un ensemble constitué entre 20 et 30 éléments. On note notamment que la précision moyenne augmente d'environ 1 point de pourcentage passant d'un score de 0.767 à une valeur de 0.777. Une nouvelle fois, il sera important dans les recherches futures, d'intégrer la quantification des incertitudes de la base de données d'entraînement afin de fournir un budget d'incertitude plus complet sur ces problématiques de classification de l'état d'agglomération.

La figure 4.14 présente la calibration (score ECE) ainsi que l'entropie sur les mauvaises classifications des ensembles en fonction du nombre d'éléments constitutifs.

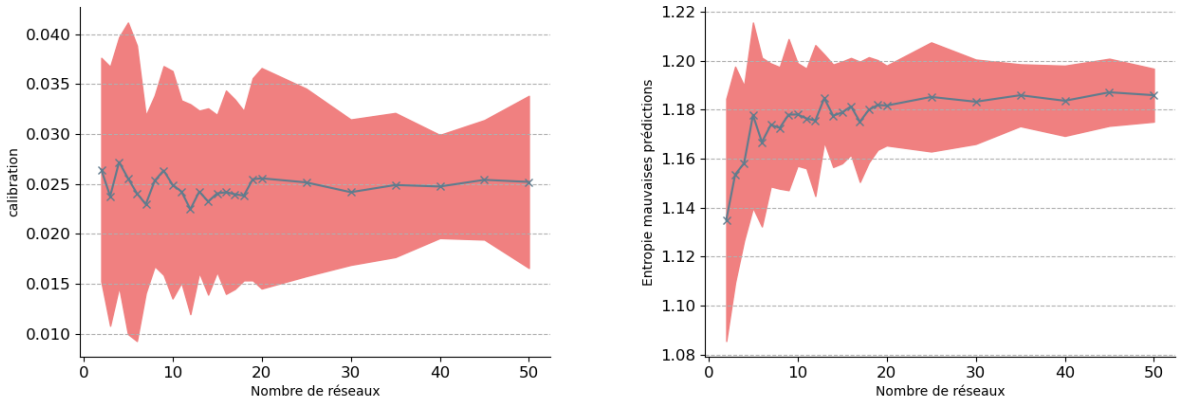


FIGURE 4.14 – Évolution de la calibration (**à gauche**) ainsi que de l'entropie sur les mauvaises prédictions (**à droite**) des ensembles en fonction du nombre d'éléments constitutifs.

Sur notre cas d'application, la calibration moyenne atteint un bon score avec une valeur ECE évoluant autour de 0.025. La performance globale ne semble pas réellement impactée par l'augmentation du nombre de prédicteurs dans l'ensemble. En revanche, on note que la valeur se stabilise pour un ensemble constitué de 20 éléments. On note également une légère diminution de son écart-type. Au contraire du score de calibration, la constitution d'ensembles permet d'améliorer l'entropie prédictive sur les particules mal classifiées. La courbe atteint un régime "quasi"-stationnaire pour des ensembles constitués entre 20 et 30 éléments. Expérimentalement, nous avons observé que cette quantification de l'entropie permet entre autre une meilleure "détection" des particules dites "touch-completes".

Au regards de ces résultats précédent, notre solution finale de notre brique de classification sous incertitudes est compris d'un ensemble de prédicteurs contenant 25 éléments. Ce nombre assure de bonnes performances tout en maintenant un coût calculatoire raisonnable dans notre cadre laboratoire.

Afin de quantifier l'incertitude finale de notre distribution de tailles estimées, il suffit de combiner les deux approches précédentes en propageant les distributions d'entrées au sein des différents

ensembles de prédicteurs. Cette phase est alors nécessairement plus longue que la mesure effectuée sans quantification d'incertitudes et devra faire l'objet d'un travail rigoureux d'implémentation au sein de notre pipeline complet de traitement. Il est intéressant de noter qu'une fois la distribution d'entrée propagée au sein d'un premier réseau à l'aide de WGMprop, la mixture de gaussienne d'entrée résultante est conservée afin de la propager au sein des autres réseaux de l'ensemble. Cela permet de réduire drastiquement le temps de quantification, tout en maintenant une performance élevée. Enfin, lors de l'étape de mesure finale du mesurande, nous gardons uniquement les particules dont la probabilité d'appartenir à la classe *isolée*, *complète* ou *touch-complète* est d'au moins 95%.

4.3 NanometrologIA : une plateforme web sur mesure

Nous présentons dans cette section la plateforme web encapsulant l'intégralité de notre pipeline de traitement. Ainsi, afin de répondre aux besoins industriels, nous avons développé une application web contenant plusieurs pages :

- une page d'accueil présentant un schéma bloc ainsi que les sources scientifiques,
- une page de chargement au format ".tiff" permettant de lire les métadonnées des images MEB,
- une page permettant de visualiser les images segmentées,
- une page permettant de modifier l'intégralité des prédictions (segmentation, suppression des faux positifs, modifications de score de prédictions, modifications des classes prédites, ...),
- une page récapitulant l'intégralité des résultats sous forme de tableaux et proposant de télécharger les résultats de la mesure,
- une page présentant des graphiques d'analyse des résultats de la mesure.

La figure 4.15 présente des visuels de ces différentes pages :

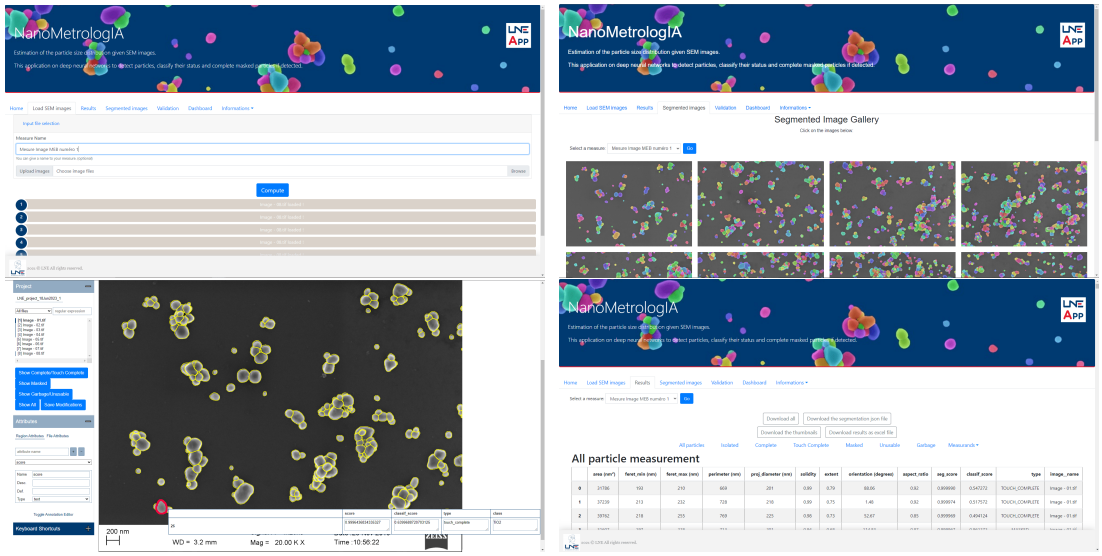


FIGURE 4.15 – Pages principales de NanoMetrologIA : la page de garde en **haut à gauche**, la page de visualisation des segmentations en **haut à droite**, la page de modification des segmentations en **bas à gauche** et enfin la page des tableaux de résultats en **bas à droite**.

De plus, l'application WEB contient une page de résultats présentant les graphiques utiles pour les experts en nanométrologie (Fig. 4.16) :

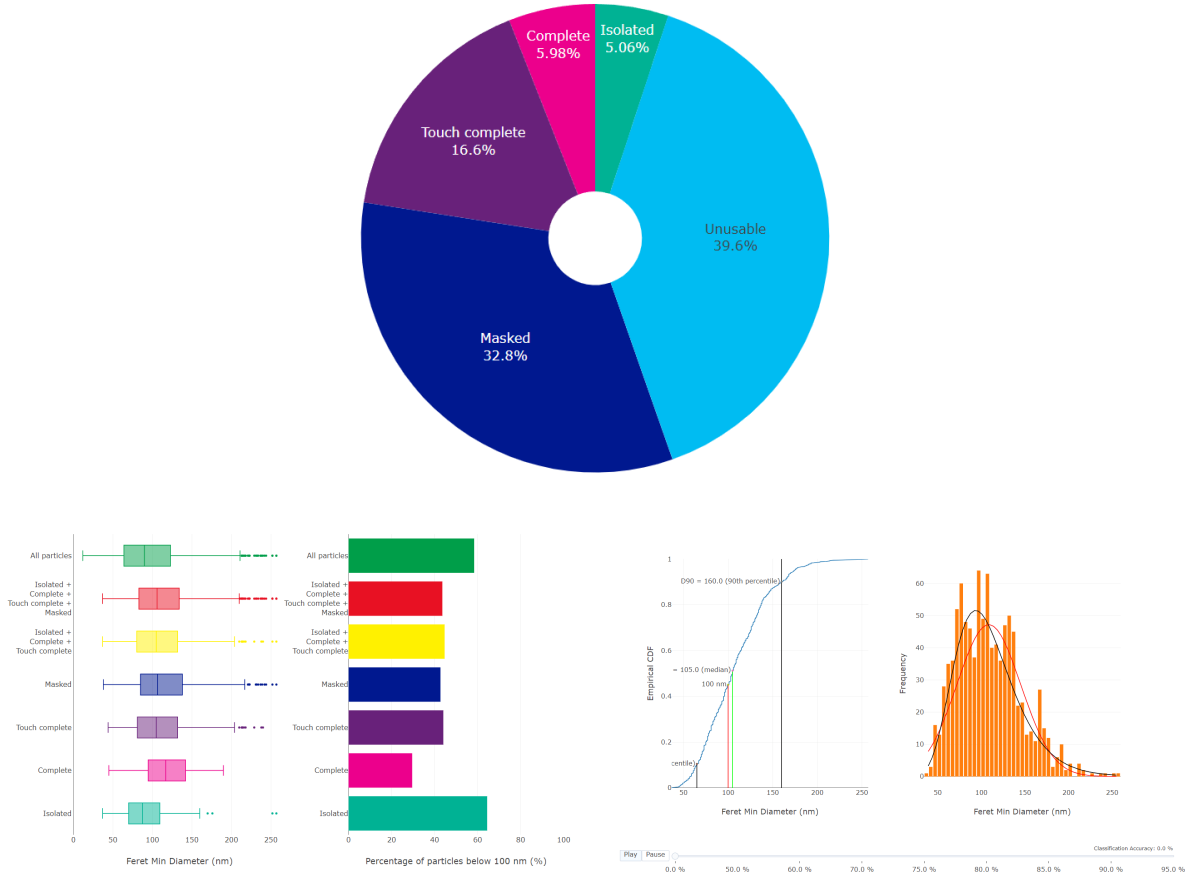


FIGURE 4.16 – Présentation des différents graphiques présents dans la page de résultats de l'application WEB NanoMetrologIA. Un camembert présentant la proportion par classe des particules détectées (**Haut**), un graphique en boîtes présentant la répartition des valeurs du mesurande en fonction de la classe de la particule (**Bas gauche**) et la PDF du mesurande considéré (**Bas droite**).

Cette application web, développé en HTML, CSS, Javascript pour le front end et en Python pour le backend, est hébergé directement sur le cluster de calcul du LNE, permettant un déploiement sur l'intranet de l'entreprise, la rendant accessible depuis une simple URL.

Cette plateforme web permet aux experts des nanoparticules d'effectuer de nombreuses mesures en très peu de temps et permet une allocation du temps de travail plus orienté sur des aspects de recherche.

La continuité des développements consiste en l'intégration des méthodes de quantification de l'incertitude prédictive au sein de cette plateforme WEB.

Chapitre 5

Conclusion et Ouverture

L'utilisation croissante des solutions technologiques basées sur les réseaux de neurones dans l'industrie et la vie courante requiert un meilleur contrôle et une meilleure quantification de l'incertitude associée aux prédictions. D'un point de vue métrologique, l'expression de l'incertitude en apprentissage profond, fortement axée autour de la dichotomie entre sources aléatoires et sources épistémiques, ne semble pas satisfaisante afin d'établir un bilan d'incertitude compréhensible et complet. La taxonomie proposée permet une vision plus claire de l'incertitude prédictive des algorithmes d'apprentissage profond en mettant en lumière l'ensemble des sources d'incertitude et leurs potentielles interactions. Cette nomenclature, nous l'espérons, posera un socle solide pour l'expression et la quantification de l'incertitude au sein de modèles d'apprentissage statistique.

Nous avons ensuite proposé une méthode générique afin de propager les incertitudes des entrées, issues de capteurs physiques, à l'inférence. La méthode développée, WGMprop, est un excellent compromis entre une approche à l'"aveugle" de Monte-Carlo et des méthodes reposant sur des hypothèses fortes sur la forme de la distribution de sortie. WGMprop propose ainsi de propager l'incertitude d'entrée en modélisant la distribution de sortie par un mélange de distributions gaussiennes. Cette propagation, basée sur le paradigme Split&Merge, repose sur la distance de 2-Wasserstein afin de quantifier l'impact de la non-linéarité du modèle sur la distribution d'entrée et ainsi contrôler le nombre de composantes nécessaire dans le mélange. La méthode présente de nombreuses propriétés intéressantes telles que des garanties théoriques de convergence, ainsi qu'une estimation de l'erreur effectuée entre notre approximation et la "vraie" distribution de sortie. La méthode a montré de hautes performances sur une variété de tâches, et d'architectures. Ces travaux préalablement présentés peuvent être poursuivis, améliorés et complétés. En particulier, l'intégration d'un seuillage adaptatif ainsi que l'étude de la meilleure direction de division permettraient d'améliorer les performances aussi bien en temps qu'en précision. Enfin, il serait intéressant de dériver les différentes formules de division pour des distributions uniformes et triangulaires afin d'étendre le champs d'application de notre méthode.

Par la suite, nous avons étudié l'incertitude prédictive liée à l'incertitude inhérente dans les paramètres des réseaux de neurones. Notre vision repose sur une incertitude couplée entre une incertitude d'échantillonage, venant du manque de connaissance liée à la taille limitée de la base de données d'entraînement, mais également de l'incertitude de répétabilité, issue de la stochasticité de l'optimisation des paramètres. À cette occasion, nous avons détaillé notre étude comparant les méthodes de l'état de l'art sur des tâches de classification, laquelle nous a permit de conclure de la "supériorité" des approches ensemblistes sur un panel de métriques sélectionnées. Ces conclusions ont notamment été soutenus par courte une méta-analyse regroupant quatre des plus récentes analyses comparatives de l'état de l'art. Par la suite, nous avons proposé une approche d'exploration du "paysage" des paramètres, tentant d'approcher localement les ensembles profonds. L'exploration WEUQ permet alors, à partir d'un mode local trouvé par descente de gradient stochastique, d'emprunter des tunnels de faible coût tout en favorisant l'exploration de zones augmentant la diversité de l'ensemble par l'intermédiaire d'un terme de régularisation. La méthode proposée permet alors de construire des ensembles de prédicteurs présentant un temps réduit d'entraînement comparé aux ensembles profonds. Enfin, la méthode a montré des premiers résultats encourageants et offre de nombreuses perspectives en vue de son amélioration. La sélection de la direction d'exploration basée sur la courbure locale de l'espace serait une amélioration sensible de l'approche et permettrait d'augmenter son ancrage théorique.

Enfin, nous avons présenté le cas d'application ainsi que les solutions technologiques mises en place afin de répondre aux besoins industriels exprimés. La plateforme WEB développée à cette occasion, nommée NanoMetrologIA, permet un traitement automatisé des images de particules de TiO_2 acquises par microscopie électronique à balayage. La mesure automatisée de la distribution de tailles de ces particules permet notamment un gain de temps significatif pour les chercheurs en nano-particules. La quantification de l'incertitude associée revêt une importance primordiale afin de pouvoir utiliser cette technologie dans un cadre réglementaire. L'intégration des développements théoriques présentés tout au long de ce rapport au sein de la plateforme de traitement permettra alors de fournir une mesure de distribution de taille sous incertitude et pourra donc être utilisée dans un cadre industriel.

Ces travaux de thèse n'incluent malheureusement pas l'étude de l'incertitude des données, quantifiant l'incertitude prédictive issue de l'incertitude de la base de données d'entraînement. Cette dernière peut être induite par des bruits, aussi bien dans l'espace des entrées acquises que des sorties correspondantes. La suite logique de ces travaux de thèse s'attardera donc sur ces notions. Nous avons intuité que cette notion peut être importante sur notre cas d'application de classification où plusieurs classes de particules sont proches au sein de leur contenu sémantique.

Bibliographie

Moloud Abdar, Farhad Pourpanah, Sadiq Hussain, Dana Rezazadegan, Li Liu, Mohammad Ghavamzadeh, Paul Fieguth, Xiaochun Cao, Abbas Khosravi, U Rajendra Acharya, et al. A review of uncertainty quantification in deep learning : Techniques, applications and challenges. *Information Fusion*, 76 :243–297, 2021.

Ahmed Hussen Abdelaziz, Shinji Watanabe, John R Hershey, Emmanuel Vincent, and Dorothea Kolossa. Uncertainty propagation through deep neural networks. In *Interspeech 2015*, 2015.

Samuel K Ainsworth, Jonathan Hayase, and Siddhartha Srinivasa. Git re-basin : Merging models modulo permutation symmetries. *arXiv preprint arXiv :2209.04836*, 2022.

Haleh Akrami, Anand Joshi, Sergul Aydore, and Richard Leahy. Deep quantile regression for uncertainty estimation in unsupervised and supervised lesion detection. *arXiv preprint arXiv :2109.09374*, 2021.

Daniel Alspach and Harold Sorenson. Nonlinear bayesian estimation using gaussian sum approximations. *IEEE transactions on automatic control*, 17(4) :439–448, 1972.

Ramón Fernandez Astudillo and João Paulo da Silva Neto. Propagation of uncertainty through multilayer perceptrons for robust automatic speech recognition. In *Twelfth Annual Conference of the International Speech Communication Association*, 2011.

Rashad Baiyasi, Miranda J Gallagher, Lauren A McCarthy, Emily K Searles, Qingfeng Zhang, Stephan Link, and Christy F Landes. Quantitative analysis of nanorod aggregation and morphology from scanning electron micrographs using semseg. *The Journal of Physical Chemistry A*, 2020.

Luis Basora, Arthur Viens, Manuel Arias Chao, and Xavier Olive. A benchmark on uncertainty quantification for deep learning prognostics. *arXiv preprint arXiv :2302.04730*, 2023.

Gregory Benton, Wesley Maddox, Sanae Lotfi, and Andrew Gordon Gordon Wilson. Loss surface simplexes for mode connecting volumes and fast ensembling. In *International Conference on Machine Learning*, pages 769–779. PMLR, 2021.

IEC BIPM, ILAC IFCC, IUPAC ISO, and OIML IUPAP. Supplement 1 to the 'guide to the expression of uncertainty in measurement'—-propagation of distributions using a monte carlo method, jcgm 101 : 2008. *International Organization for Standardization Geneva ISBN*, 2008.

Christopher M Bishop and Nasser M Nasrabadi. *Pattern recognition and machine learning*, volume 4. Springer, 2006.

David M Blei, Alp Kucukelbir, and Jon D McAuliffe. Variational inference : A review for statisticians. *Journal of the American statistical Association*, 112(518) :859–877, 2017.

Charles Blundell, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra. Weight uncertainty in neural network. In *International conference on machine learning*, pages 1613–1622. PMLR, 2015.

Leo Breiman. Bagging predictors. *Machine learning*, 24 :123–140, 1996.

Glenn W Brier. Verification of forecasts expressed in terms of probability. *Monthly weather review*, 78(1) :1–3, 1950.

Gavin Brown. *Diversity in neural network ensembles*. PhD thesis, Citeseer, 2004.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33 :1877–1901, 2020.

Colin J Burke, Patrick D Aleo, Yu-Ching Chen, Xin Liu, John R Peterson, Glenn H Sembroski, and Joshua Yao-Yu Lin. Deblending and classifying astronomical sources with mask r-cnn deep learning. *Monthly Notices of the Royal Astronomical Society*, 490(3) :3952–3965, 2019.

Sebastian Buschjäger, Lukas Pfahler, and Katharina Morik. Generalized negative correlation learning for deep ensembling. *arXiv preprint arXiv :2011.02952*, 2020.

Daniela Calvetti, Lothar Reichel, and Danny Chris Sorensen. An implicitly restarted lanczos method for large symmetric eigenvalue problems. *Electronic Transactions on Numerical Analysis*, 2(1) :21, 1994.

Ricardo JGB Campello, Davoud Moulavi, and Jörg Sander. Density-based clustering based on hierarchical density estimates. In *Pacific-Asia conference on knowledge discovery and data mining*, pages 160–172. Springer, 2013.

George DC Cavalcanti, Luiz S Oliveira, Thiago JM Moura, and Guilherme V Carvalho. Combining diversity measures for ensemble pruning. *Pattern Recognition Letters*, 74 :38–45, 2016.

Heang-Ping Chan, Ravi K Samala, Lubomir M Hadjiiski, and Chuan Zhou. Deep learning in medical image analysis. *Deep Learning in Medical Image Analysis : Challenges and Applications*, pages 3–21, 2020.

Tianqi Chen, Emily Fox, and Carlos Guestrin. Stochastic gradient hamiltonian monte carlo. In *International conference on machine learning*, pages 1683–1691. PMLR, 2014.

Djork-Arné Clevert, Thomas Unterthiner, and Sepp Hochreiter. Fast and accurate deep network learning by exponential linear units (elus). *arXiv preprint arXiv :1511.07289*, 2015.

Paul G Constantine, Eric Dow, and Qiqi Wang. Active subspace methods in theory and practice : applications to kriging surfaces. *SIAM Journal on Scientific Computing*, 36(4) :A1500–A1524, 2014.

L Coquelin, N Fischer, N Feltin, L Devoille, and G Felhi. Towards the use of deep generative models for the characterization in size of aggregated tio2 nanoparticles measured by scanning electron microscopy (sem). *Materials Research Express*, 6(8) :085001, 2019.

Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20(3) :273–297, 1995.

Maurice Cox and Anthony O'Hagan. Meaningful expression of uncertainty in measurement. *Accreditation and Quality Assurance*, 27(1) :19–37, 2022.

Subrata Das, Lalit Jain, and Arup Das. Deep learning for military image captioning. In *2018 21st International Conference on Information Fusion (FUSION)*, pages 2165–2171. IEEE, 2018.

Paul De Bièvre. The 2012 international vocabulary of metrology :"vim". *Chemistry International–Newsmagazine for IUPAC*, 34(3) :26–27, 2012.

A Delvallée, N Feltin, S Ducourtieux, M Trabelsi, and JF Hochepied. Direct comparison of afm and sem measurements on the same set of nanoparticles. *Measurement Science and Technology*, 26(8) : 085601, 2015.

Kyle J DeMars, Robert H Bishop, and Moriba K Jah. Entropy-based approach for uncertainty propagation of nonlinear dynamical systems. *Journal of Guidance, Control, and Dynamics*, 36(4) : 1047–1057, 2013.

Stefan Depeweg, José Miguel Hernández-Lobato, Finale Doshi-Velez, and Steffen Udluft. Uncertainty decomposition in bayesian neural networks with latent variables. *arXiv preprint arXiv :1706.08495*, 2017.

Armen Der Kiureghian and Ove Ditlevsen. Aleatory or epistemic ? does it matter ? *Structural safety*, 31(2) :105–112, 2009.

John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of machine learning research*, 12(7), 2011.

N ENV13005. Guide pour l'expression de l'incertitude de mesure. *Paris, France : AFNOR*, 1999.

Mark Everingham, Luc Van Gool, Christopher K. I. Williams andJohn Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *International Journal of Computer Vision*, 88 : 303–338, 2010. doi : https://doi.org/10.1007/s11263-009-0275-4. URL https://doi.org/10.1007/s11263-009-0275-4.

Sebastian Farquhar, Michael A Osborne, and Yarin Gal. Radial bayesian neural networks : Beyond discrete support in large-scale bayesian deep learning. In *International Conference on Artificial Intelligence and Statistics*, pages 1352–1362. PMLR, 2020.

Friedrich Faubel and Dietrich Klakow. Further improvement of the adaptive level of detail transform : Splitting in direction of the nonlinearity. In *2010 18th European Signal Processing Conference*, pages 850–854. IEEE, 2010.

Friedrich Faubel, John McDonough, and Dietrich Klakow. The split and merge unscented gaussian mixture filter. *IEEE Signal Processing Letters*, 16(9) :786–789, 2009.

Pedro F Felzenszwalb and Daniel P Huttenlocher. Efficient graph-based image segmentation. *International journal of computer vision*, 59(2) :167–181, 2004.

LR Feret. Assoc. internat. pour íessai des mat, 1931.

Felix Fiedler and Sergio Lucia. Improved uncertainty quantification for neural networks with bayesian last layer. *arXiv preprint arXiv :2302.10975*, 2023.

David Freedman, Robert Pisani, and Roger Purves. Statistics (international student edition). *Pisani, R. Purves, 4th edn. WW Norton & Company, New York*, 2007.

Brendan J. Frey and Delbert Dueck. Clustering by passing messages between data points. *Science*, 315(5814) :972–976, 2007. doi : 10.1126/science.1136800. URL https://www.science.org/doi/abs/10.1126/science.1136800.

Jerome H Friedman. Greedy function approximation : a gradient boosting machine. *Annals of statistics*, pages 1189–1232, 2001.

Keinosuke Fukunaga and Larry Hostetler. The estimation of the gradient of a density function, with applications in pattern recognition. *IEEE Transactions on information theory*, 21(1) :32–40, 1975.

Wayne A Fuller. *Measurement error models*. John Wiley & Sons, 2009.

Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation : Representing model uncertainty in deep learning. In *international conference on machine learning*, pages 1050–1059. PMLR, 2016.

Timur Garipov, Pavel Izmailov, Dmitrii Podoprikhin, Dmitry P Vetrov, and Andrew G Wilson. Loss surfaces, mode connectivity, and fast ensembling of dnns. *Advances in neural information processing systems*, 31, 2018.

Jochen Gast and Stefan Roth. Lightweight probabilistic deep networks. pages 3369–3378, 2018.

Jakob Gawlikowski, Cedrique Rovile Njieutcheu Tassi, Mohsin Ali, Jongseok Lee, Matthias Humt, Jianxiang Feng, Anna Kruspe, Rudolph Triebel, Peter Jung, Ribana Roscher, et al. A survey of uncertainty in deep neural networks. *arXiv preprint arXiv :2107.03342*, 2021.

Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 249–256. JMLR Workshop and Conference Proceedings, 2010.

Jianping Gou, Baosheng Yu, Stephen J Maybank, and Dacheng Tao. Knowledge distillation : A survey. *International Journal of Computer Vision*, 129 :1789–1819, 2021.

K Chidananda Gowda and GJPR Krishna. Agglomerative clustering using the concept of mutual nearest neighbourhood. *Pattern recognition*, 10(2) :105–112, 1978.

Sorin Grigorescu, Bogdan Trasnea, Tiberiu Cocias, and Gigel Macesanu. A survey of deep learning techniques for autonomous driving. *Journal of Field Robotics*, 37(3) :362–386, 2020.

Cornelia Gruber, Patrick Oliver Schenk, Malte Schierholz, Frauke Kreuter, and Göran Kauermann. Sources of uncertainty in machine learning – a statisticians' view, 2023.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q Weinberger. On calibration of modern neural networks. In *International conference on machine learning*, pages 1321–1330. PMLR, 2017.

Fredrik K Gustafsson, Martin Danelljan, and Thomas B Schon. Evaluating scalable bayesian deep learning methods for robust computer vision. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops*, pages 318–319, 2020.

Simon Haykin. *Neural networks : a comprehensive foundation*. Prentice Hall PTR, 1998.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers : Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, pages 1026–1034, 2015.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017.

JB Heaton, Nicholas G Polson, and Jan Hendrik Witte. Deep learning in finance. *arXiv preprint arXiv :1602.06561*, 2016.

José Miguel Hernández-Lobato and Ryan Adams. Probabilistic backpropagation for scalable learning of bayesian neural networks. In *International conference on machine learning*, pages 1861–1869. PMLR, 2015.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv :1503.02531*, 2015.

Jennifer A Hoeting, David Madigan, Adrian E Raftery, and Chris T Volinsky. Bayesian model averaging : a tutorial (with comments by m. clyde, david draper and ei george, and a rejoinder by the authors. *Statistical science*, 14(4) :382–417, 1999.

Jian Huang, Junyi Chai, and Stella Cho. Deep learning in finance and banking : A literature review and classification. *Frontiers of Business Research in China*, 14(1) :1–24, 2020.

Eyke Hüllermeier and Willem Waegeman. Aleatoric and epistemic uncertainty in machine learning : An introduction to concepts and methods. *Machine Learning*, 110 :457–506, 2021.

Sergey Ioffe and Christian Szegedy. Batch normalization : Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pages 448–456. pmlr, 2015.

Pavel Izmailov, Sharad Vikram, Matthew D Hoffman, and Andrew Gordon Gordon Wilson. What are bayesian neural network posteriors really like ? In *International conference on machine learning*, pages 4629–4640. PMLR, 2021.

Libin Jiao, Lianzhi Huo, Changmiao Hu, and Ping Tang. Refined unet : Unet-based refinement network for cloud and shadow precise segmentation. *Remote Sensing*, 12(12) :2001, 2020.

Jeremiah W Johnson. Adapting mask-rcnn for automatic nucleus segmentation. *arXiv preprint arXiv :1805.00500*, 2018.

Laurent Valentin Jospin, Hamid Laga, Farid Boussaid, Wray Buntine, and Mohammed Bennamoun. Hands-on bayesian neural networks—a tutorial for deep learning users. *IEEE Computational Intelligence Magazine*, 17(2) :29–48, 2022.

James Joyce. Bayes' theorem. 2003.

Simon J Julier and Jeffrey K Uhlmann. Unscented filtering and nonlinear estimation. *Proceedings of the IEEE*, 92(3) :401–422, 2004.

Alex Kendall and Yarin Gal. What uncertainties do we need in bayesian deep learning for computer vision ? *Advances in neural information processing systems*, 30, 2017.

Mohammad Emtiyaz Khan and Håvard Rue. The bayesian learning rule. *arXiv preprint arXiv :2107.04562*, 2021.

Diederik P Kingma and Jimmy Ba. Adam : A method for stochastic optimization. *arXiv preprint arXiv :1412.6980*, 2014.

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

Anders Krogh and John A Hertz. A simple weight decay can improve generalization. In *Advances in neural information processing systems*, pages 950–957, 1992.

Solomon Kullback and Richard A Leibler. On information and sufficiency. *The annals of mathematical statistics*, 22(1) :79–86, 1951.

Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. *Advances in neural information processing systems*, 30, 2017.

Yann LeCun. The mnist database of handwritten digits. *http ://yann. lecun. com/exdb/mnist/*, 1998.

Jinsol Lee and Ghassan AlRegib. Gradients as a measure of uncertainty in neural networks. In *2020 IEEE International Conference on Image Processing (ICIP)*, pages 2416–2420. IEEE, 2020.

Chunyuan Li, Changyou Chen, David Carlson, and Lawrence Carin. Preconditioned stochastic gradient langevin dynamics for deep neural networks. In *Proceedings of the AAAI conference on artificial intelligence*, volume 30, 2016.

Xiaomeng Li, Hao Chen, Xiaojuan Qi, Qi Dou, Chi-Wing Fu, and Pheng-Ann Heng. H-denseunet : hybrid densely connected unet for liver and tumor segmentation from ct volumes. *IEEE transactions on medical imaging*, 37(12) :2663–2674, 2018.

Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2117–2125, 2017.

Geert Litjens, Peter Bandi, Babak Ehteshami Bejnordi, Oscar Geessink, Maschenka Balkenhol, Peter Bult, Altuna Halilovic, Meyke Hermsen, Rob van de Loo, Rob Vogels, et al. 1399 h&e-stained sentinel lymph node sections of breast cancer patients : the camelyon dataset. *GigaScience*, 7(6) : giy065, 2018.

Menglu Liu, Junyu Dong, Xinghui Dong, Hui Yu, and Lin Qi. Segmentation of lung nodule in ct images based on mask r-cnn. In *2018 9th International Conference on Awareness Science and Technology (iCAST)*, pages 1–6. IEEE, 2018.

Xiaobo Liu, Yuwei Zhang, Hongdi Jing, Liancheng Wang, and Sheng Zhao. Ore image segmentation method using u-net and res_unet convolutional networks. *RSC Advances*, 10(16) :9396–9406, 2020.

Stuart Lloyd. Least squares quantization in pcm. *IEEE transactions on information theory*, 28(2) : 129–137, 1982.

Ilya Loshchilov and Frank Hutter. Sgdr : Stochastic gradient descent with warm restarts. *arXiv preprint arXiv :1608.03983*, 2016.

Andrew L Maas, Awni Y Hannun, Andrew Y Ng, et al. Rectifier nonlinearities improve neural network acoustic models. In *Proc. ICML*, volume 30, page 3, 2013.

David JC MacKay. A practical bayesian framework for backpropagation networks. *Neural computation*, 4(3) :448–472, 1992.

Wesley J Maddox, Pavel Izmailov, Timur Garipov, Dmitry P Vetrov, and Andrew Gordon Wilson. A simple baseline for bayesian uncertainty in deep learning. *Advances in neural information processing systems*, 32, 2019.

Andrey Malinin and Mark Gales. Predictive uncertainty estimation via prior networks. *Advances in neural information processing systems*, 31, 2018.

J Martin and C Elster. Aleatoric uncertainty for errors-in-variables models in deep regression. *Neural Processing Letters*, pages 1–20, 2022.

Ryan Martin, Raymond Mess, and Stephen G Walker. Empirical bayes posterior concentration in sparse high-dimensional linear models. 2017.

Waldyn G Martinez. Ensemble pruning via quadratic margin maximization. *IEEE Access*, 9 : 48931–48951, 2021.

Leland McInnes, John Healy, and James Melville. Umap : Uniform manifold approximation and projection for dimension reduction. arxiv 2018. *arXiv preprint arXiv :1802.03426*, 1802.

Daily Milanés-Hermosilla, Rafael Trujillo Codorniú, René López-Baracaldo, Roberto Sagaró-Zamora, Denis Delisle-Rodriguez, John Jairo Villarejo-Mayor, and José Ricardo Núñez-Álvarez. Monte carlo dropout for uncertainty estimation and motor imagery classification. *Sensors*, 21(21) :7241, 2021.

Paul Monchot, Loïc Coquelin, Khaled Guerroudj, Nicolas Feltin, Alexandra Delvallée, Loïc Crouzier, and Nicolas Fischer. Deep learning based instance segmentation of titanium dioxide particles in the form of agglomerates in scanning electron microscopy. *Nanomaterials*, 11(4) :968, 2021.

Paul Monchot, Loïc Coquelin, Sébastien J. Petit, Sébastien Marmin, Erwan Le Pennec, and Nicolas Fischer. Input uncertainty propagation through trained neural networks. *Proceedings of the 40th international conference on machine learning (ICML-23)*, 2023.

Christopher Z Mooney. *Monte carlo simulation*. Number 116. Sage, 1997.

Khan Muhammad, Amin Ullah, Jaime Lloret, Javier Del Ser, and Victor Hugo C de Albuquerque. Deep learning for safe autonomous driving : Current challenges and future directions. *IEEE Transactions on Intelligent Transportation Systems*, 22(7) :4316–4336, 2020.

Paisarn Muneesawang and Chitnarong Sirisathitkul. Size measurement of nanoparticle assembly using multilevel segmented tem images. *Journal of Nanomaterials*, 2015, 2015.

Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th international conference on machine learning (ICML-10)*, pages 807–814, 2010.

Jay Nandy, Wynne Hsu, and Mong Li Lee. Towards maximizing the representation gap between in-domain & out-of-distribution examples. *Advances in Neural Information Processing Systems*, 33 :9239–9250, 2020.

Sumit K Nath, Kannappan Palaniappan, and Filiz Bunyak. Cell segmentation using coupled level sets and graph-vertex coloring. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 101–108. Springer, 2006.

Radford M Neal. *Bayesian learning for neural networks*, volume 118. Springer Science & Business Media, 2012.

Radford M Neal et al. Mcmc using hamiltonian dynamics. *Handbook of markov chain monte carlo*, 2(11) :2, 2011.

Philipp Oberdiek, Matthias Rottmann, and Hanno Gottschalk. Classification uncertainty of deep neural networks based on gradient information. In *Artificial Neural Networks in Pattern Recogni-*

tion : 8th IAPR TC3 Workshop, ANNPR 2018, Siena, Italy, September 19–21, 2018, Proceedings 8, pages 113–125. Springer, 2018.

A.G. Okunev, A.V. Mashukov, M.Y. andNartova, and A.V Matveev. Nanoparticle recognition on scanning probe microscopy images using computer vision and deep learning. *Nanomaterials*, 10 : 1285, 2020.

Luis A Ortega, Rafael Cabañas, and Andres Masegosa. Diversity and generalization in neural network ensembles. In *International Conference on Artificial Intelligence and Statistics*, pages 11720–11743. PMLR, 2022.

Yaniv Ovadia, Emily Fertig, Jie Ren, Zachary Nado, David Sculley, Sebastian Nowozin, Joshua Dillon, Balaji Lakshminarayanan, and Jasper Snoek. Can you trust your model's uncertainty ? evaluating predictive uncertainty under dataset shift. *Advances in neural information processing systems*, 32, 2019.

Donald Bruce Owen. A table of normal integrals : A table. *Communications in Statistics-Simulation and Computation*, 9(4) :389–419, 1980.

J Padarian, B Minasny, and AB McBratney. Assessing the uncertainty of deep learning soil spectral models using monte carlo dropout. *Geoderma*, 425 :116063, 2022.

Chiwoo Park, Jianhua Z Huang, Jim X Ji, and Yu Ding. Segmentation, inference and classification of partially overlapping nanoparticles. *IEEE transactions on pattern analysis and machine intelligence*, 35(3) :1–1, 2012.

Muhammad Imran Razzak, Saeeda Naz, and Ahmad Zaib. Deep learning for medical image processing : Overview, challenges and the future. *Classification in BioApps : Automation of Decision Making*, pages 323–350, 2018.

Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn : Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 28, 2015.

Hippolyt Ritter, Aleksandar Botev, and David Barber. A scalable laplace approximation for neural networks. In *6th International Conference on Learning Representations, ICLR 2018-Conference Track Proceedings*, volume 6. International Conference on Representation Learning, 2018.

Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net : Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.

Sebastian Ruder. An overview of gradient descent optimization algorithms. *arXiv preprint arXiv :1609.04747*, 2016.

David W Scott. *Multivariate density estimation : theory, practice, and visualization*. John Wiley & Sons, 2015.

Nabeel Seedat and Christopher Kanan. Towards calibrated and scalable uncertainty representations for neural networks. *arXiv preprint arXiv :1911.00104*, 2019.

Murat Sensoy, Lance Kaplan, and Melih Kandemir. Evidential deep learning to quantify classification uncertainty. *Advances in neural information processing systems*, 31, 2018.

Dinggang Shen, Guorong Wu, and Heung-Il Suk. Deep learning in medical image analysis. *Annual review of biomedical engineering*, 19 :221–248, 2017.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv :1409.1556*, 2014.

Harold W Sorenson and Daniel L Alspach. Recursive bayesian estimation using gaussian sums. *Automatica*, 7(4) :465–479, 1971.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout : a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1) :1929–1958, 2014.

Brian Staber and Sébastien Da Veiga. Benchmarking bayesian neural networks and evaluation metrics for regression tasks. *arXiv preprint arXiv :2206.06779*, 2022.

Ondřej Straka, Jindřich Duník, and Ivo Punčochář. Directional splitting for structure adaptation of bayesian filters. In *2016 American Control Conference (ACC)*, pages 2705–2710. IEEE, 2016.

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826, 2016.

Natasa Tagasovska and David Lopez-Paz. Single-model uncertainties for deep learning. *Advances in Neural Information Processing Systems*, 32, 2019.

F Boray Tek, Andrew G Dempster, and Izzet Kale. Blood cell segmentation using minimum area watershed and circle radon transformations. In *Mathematical morphology : 40 years on*, pages 441–454. Springer, 2005.

Gabriel Terefjanu. An adaptive split-merge scheme for uncertainty propagation using gaussian mixture models. In *49th AIAA Aerospace Sciences Meeting including the New Horizons Forum and Aerospace Exposition*, page 890, 2011.

Tijmen Tieleman, Geoffrey Hinton, et al. Lecture 6.5-rmsprop : Divide the gradient by a running average of its recent magnitude. *COURSERA : Neural networks for machine learning*, 4(2) :26–31, 2012.

Jessica S Titensky, Hayden Jananthan, and Jeremy Kepner. Uncertainty propagation in deep neural networks using extended kalman filtering. In *2018 IEEE MIT Undergraduate Research Technology Conference (URTC)*, pages 1–4. IEEE, 2018.

Lisa Torrey and Jude Shavlik. Transfer learning. pages 242–264, 2010.

Meet P Vadera, Adam D Cobb, Brian Jalaian, and Benjamin M Marlin. Ursabench : Comprehensive benchmarking of approximate bayesian inference methods for deep neural networks. *arXiv preprint arXiv :2007.04466*, 2020.

Joost Van Amersfoort, Lewis Smith, Yee Whye Teh, and Yarin Gal. Uncertainty estimation using a single deep deterministic neural network. In *International conference on machine learning*, pages 9690–9700. PMLR, 2020.

Laurens van der Maaten and Geoffrey E. Hinton. Visualizing high-dimensional data using t-sne. *Journal of Machine Learning Research*, 9 :2579–2605, 2008.

Cédric Villani. *Optimal transport : old and new*, volume 338. Springer, 2009.

Larry Wasserman. *All of statistics : a concise course in statistical inference*, volume 26. Springer, 2004.

Meng Wei, Hairong Gu, Min Ye, Qiao Wang, Xinxin Xu, and Chenguang Wu. Remaining useful life prediction of lithium-ion batteries based on monte carlo dropout and gated recurrent unit. *Energy Reports*, 7 :2862–2871, 2021.

Max Welling and Yee W Teh. Bayesian learning via stochastic gradient langevin dynamics. In *Proceedings of the 28th international conference on machine learning (ICML-11)*, pages 681–688, 2011.

Yeming Wen, Paul Vicol, Jimmy Ba, Dustin Tran, and Roger Grosse. Flipout : Efficient pseudo-independent weight perturbations on mini-batches. *arXiv preprint arXiv :1803.04386*, 2018.

Yu Weng, Tianbao Zhou, Yujie Li, and Xiaoyu Qiu. Nas-unet : Neural architecture search for medical image segmentation. *IEEE Access*, 7 :44247–44257, 2019.

Florian Wenzel, Kevin Roth, Bastiaan S Veeling, Jakub Światkowski, Linh Tran, Stephan Mandt, Jasper Snoek, Tim Salimans, Rodolphe Jenatton, and Sebastian Nowozin. How good is the bayes posterior in deep neural networks really ? *arXiv preprint arXiv :2002.02405*, 2020.

Andrew G Wilson and Pavel Izmailov. Bayesian deep learning and a probabilistic perspective of generalization. *Advances in neural information processing systems*, 33 :4697–4708, 2020.

Andrew Gordon Wilson. The case for bayesian deep learning. *arXiv preprint arXiv :2001.10995*, 2020.

Ronald Wilson, Navid Asadizanjani, Domenic Forte, and Damon L Woodard. Histogram-based auto segmentation : A novel approach to segmenting integrated circuit structures from sem images. *arXiv preprint arXiv :2004.13874*, 2020.

Danny Wood, Tingting Mu, Andrew Webb, Henry Reeve, Mikel Lujan, and Gavin Brown. A unified theory of diversity in ensemble learning. *arXiv preprint arXiv :2301.03962*, 2023.

Mitchell Wortsman, Maxwell C Horton, Carlos Guestrin, Ali Farhadi, and Mohammad Rastegari. Learning neural network subspaces. In *International Conference on Machine Learning*, pages 11217–11227. PMLR, 2021.

Yue Wu, Wen Wang, Fang Zhang, Zhitao Xiao, Jun Wu, and Lei Geng. Nanoparticle size measurement method based on improved watershed segmentation. In *Proceedings of the 2018 International Conference on Electronics and Electrical Engineering Technology*, pages 232–237, 2018.

Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist : a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv :1708.07747*, 2017.

Bin Zhang and Yung C Shin. An adaptive gaussian mixture method for nonlinear uncertainty propagation in neural networks. *Neurocomputing*, 458 :170–183, 2021.

Rui Zhang, Chao Cheng, Xuehua Zhao, and Xuechen Li. Multiscale mask r-cnn–based lung tumor detection using pet imaging. *Molecular imaging*, 18 :1536012119863531, 2019.

Shaofeng Zhang, Meng Liu, and Junchi Yan. The diversified ensemble neural network. *Advances in Neural Information Processing Systems*, 33 :16001–16011, 2020a.

Weixing Zhang, Chandi Witharana, Anna K Liljedahl, and Mikhail Kanevskiy. Deep convolutional neural networks for automated characterization of arctic ice-wedge polygons in very high spatial resolution aerial imagery. *Remote Sensing*, 10(9) :1487, 2018.

Yiqing Zhang, Jun Chu, Lu Leng, and Jun Miao. Mask-refined r-cnn : A network for refining object details in instance segmentation. *Sensors*, 20(4) :1010, 2020b.

Kang Zhao, Jungwon Kang, Jaewook Jung, and Gunho Sohn. Building extraction from satellite images using mask r-cnn with building boundary regularization. In *CVPR Workshops*, pages 247–251, 2018.

Michael Zhu and Suyog Gupta. To prune, or not to prune : exploring the efficacy of pruning for model compression. *arXiv preprint arXiv :1710.01878*, 2017.

Appendices

Annexe A

Ensembles profonds : répétabilité et échantillonnage ?

Nous présentons ici une réflexion sur les notions d'incertitude de répétabilité et d'échantillonnage.

A.1 Bayesian or not Bayesian ?

La quantification de l'incertitude des paramètres d'un réseau de neurones est en grande partie étudiée sous le prisme du manque de données dans la base de données d'apprentissage créant des "zones" dans l'espace des entrées où l'algorithme a une plus forte variance de prédiction. Cette incertitude, que nous avons appelé incertitude d'échantillonnage, est par définition fortement liée au framework bayésien, permettant de répondre à la question :

*Quelle est la **probabilité** de l'**hypothèse** (ici, des paramètres du réseau de neurones) compte tenu des **données observées** ?*

Remarque 15.

La question répondue par la quantification de l'incertitude "épistémique" est donc différente de cette dernière.

Remarque 16.

Le principe du cadre bayésien repose sur une connaissance a priori, mise à jour lors de l'observation de données. Cette méthodologie repose sur la règle de Bayes suivante :

$$p(A \mid B) = \frac{p(B \mid A) \cdot p(A)}{p(B)} \tag{A.1}$$

où A et B sont deux variables aléatoires.

Ainsi, injecter une connaissance à priori à travers l'établissement de la distribution à priori $p(A)$ et évaluer la fonction de vraisemblance $p(B \mid A)$ à l'aide de l'observation de données permet d'estimer le postérieur de A conditionnellement à B : $p(A \mid B)$.

En apprentissage profond, l'estimation du postérieur des paramètres conditionnellement à la base de données d'apprentissage permet, entre autres, de marginaliser sur les paramètres du réseau afin de tenir compte de leur éventuelle variabilité dans la prédiction finale. On rappelle que pour une nouvelle entrée $\mathbf{x}^*$, on obtient la distribution de $\mathbf{y}^* = f(\mathbf{x}^*)$, où f désigne le prédicteur, par :

$$p(\mathbf{y}^* \mid \mathbf{x}^*, \mathcal{D}_n) = \int p(\mathbf{y}^* \mid \mathbf{x}^*, \theta) p(\theta \mid \mathcal{D}_N) d\theta \approx \frac{1}{N} \sum_{i=1}^{N} p(\mathbf{y}^* \mid \mathbf{x}^*, \theta_i) \quad \theta_i \sim p(\theta \mid \mathcal{D}_N) \tag{A.2}$$

Ainsi, le framework bayésien fournit un cadre méthodologique et mathématique solide afin d'estimer la densité de probabilité de la sortie d'un prédicteur conditionnellement à la base de données d'apprentissage. Cependant, comme souligné par l'état de l'art, et comme nous avons pu le voir lors de notre étude comparative 3.1, les méthodes bayésiennes performent en général moins bien en termes de précision et de calibration que certaines méthodes qui sont, à l'origine, présentées comme "non-bayésiennes" (ensemble profond et méthode SWAG notamment).

Remarque 17.

Khan and Rue (2021, Sec. 4.4, Fig.1) présente des éléments permettant d'expliquer ce phénomène. Tandis que les algorithmes d'optimisation classiques tel que le SGD cherchent des minimums pouvant être dans des régions "étroites" et "profondes", les méthodes bayésiennes, quant à elles, cherchent des zones de l'espace présentant des solutions plus stables aux petites variations, pouvant engendrer des performances plus faibles.

Cette dichotomie entre méthodes "bayésiennes" et méthodes "non-bayésiennes" présente dans l'état de l'art, ne nous semble pas pertinente, et ce pour plusieurs raisons :

— l'ensemble des méthodes dites bayésiennes, nécessitent de nombreuses approximations ne per-

mettant plus de les qualifier de 100% bayésiennes (Wilson, 2020), à l'image de l'approximation de Laplace où des méthodes d'inférence variationnelle (cf. section 1.2.2). On peut également citer les méthodes de MCMC qui intègrent également des approximations afin de les adapter aux algorithmes d'apprentissage par **mini**-batchs,

— de part la stochasticité de l'algorithme de descente de gradient, ainsi que des symétries inhérentes aux réseaux de neurones, le postérieur des poids ne "s'effondre" pas (un en Dirac) avec une base de données d'apprentissage infinie. De plus, certaines méthodes classées comme bayésiennes, telle que le MCD, ne présentent également pas cette propriété. Dans cet esprit, on note que l'espace des paramètres d'un réseau de neurones n'est pas identifiable : les réseaux de neurones présentent un très grand nombre de permutations, laissant le prédicteur inchangé,

— il semble compliqué de pouvoir appliquer un prior informatif sur les paramètres du réseau de neurones, du fait, notamment, de la haute dimension du problème,

— les algorithmes d'optimisation des réseaux de neurones ne garantissent pas une convergence vers un minimum global. Au mieux, les optimisations conduisent à une convergence au sein d'une boule de rayon ϵ autour d'un minimum local.

Remarque 18.

Ce point de vue peut être renforcé par les travaux de Khan and Rue (2021) proposant une vision de différents algorithmes d'apprentissage statistique comme des instances d'un seul et même algorithme : La "Bayesian learning rule" (BLR) (ou règle d'apprentissage bayésienne en français), où différentes approximations conduisent à différents algorithmes connus d'apprentissage statistique. Ainsi, l'optimisation SGD peut être vu comme une optimisation BLR où le postérieur est approché par une distribution gaussienne multivariée, de moyenne inconnue, et de matrice de variance/covariance égale à l'identité. L'optimisation SGD conduit alors à l'estimation de cette moyenne. Les auteurs soulignent néanmoins que les méthodes de premier-ordre ne permettent pas une bonne estimation du postérieur.

Enfin, (Wilson and Izmailov, 2020) et (Izmailov et al., 2021) présentent des éléments en faveur de l'utilisation des ensembles profonds afin d'effectuer du BMA (équation A.2). Nous reprenons ici leur figure A.1 présente dans (Wilson and Izmailov, 2020).

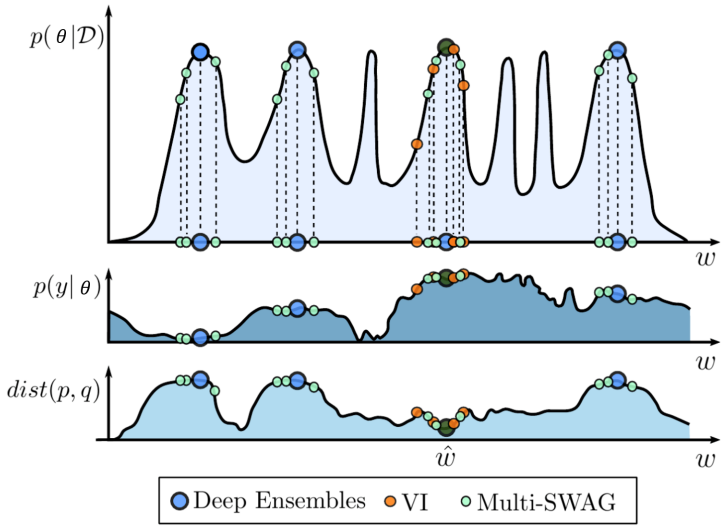


FIGURE A.1 – Représentation des méthodes d'ensemble profond (bleu), d'inférence variationelle (orange) et Multi-SWAG (vert) au sein du postérieur $p(\theta \mid \mathcal{D}_n)$ (**haut**), de la distribution prédictive conditionnellement aux paramètres $p(y \mid x, \theta)$ (**milieu**) et la distance entre la vraie distribution et l'approximation (**bas**).

Tout d'abord, cette représentation intuite le fait que les méthodes d'inférence variationelle telle que l'approximation de Laplace (Ritter et al., 2018) et Bayes-by-Backprop (Blundell et al., 2015) ne permettent pas une bonne estimation de l'incertitude prédictive. En effet, au sein d'un même mode de $p(\theta \mid \mathcal{D}_n)$, il y a peu de variation au sein de $p(\mathbf{y} \mid \mathbf{x}, \theta)$, engendrant un risque de sous-estimation de l'incertitude prédictive. Autrement dit, ajouter de nombreux échantillons d'un même mode introduit une redondance dans l'information intégrée au sein de l'équation A.2. De plus, cette représentation tend à suggérer que la distribution prédictive conditionnellement aux paramètres $p(\mathbf{y}^* \mid \mathbf{x}^*, \theta)$, bien que quasi-constante localement (au sein d'un même mode), montre de fortes variabilités globales (entre les différents modes).

La figure précédente peut être mathématiquement illustrée de la manière suivante : Pour une fonction de coût $\mathcal{L}$ consistant à minimiser l'opposé du logarithme de la vraisemblance, notée $\mathcal{L}_{NLL}$ (pour Negative Log Likelihood), on minimise la quantité suivante :

$$-\log\left(p(\theta \mid \mathcal{D}_N)\right) \propto -\log\left(p(\mathcal{D}_N \mid \theta)p(\theta)\right) \propto -\log\left(p(\mathcal{D}_N \mid \theta)\right) - \log\left(p(\theta)\right) \tag{A.3}$$

Ainsi, pendant l'entraînement, maximiser la vraisemblance, i.e. minimiser $-\log\left(p(\mathcal{D}_N \mid \theta)\right)$, revient à maximiser $p(\theta \mid \mathcal{D}_N)$, et donc à rechercher les zones de fortes probabilités du postérieur des paramètres du réseau de neurones conditionnellement à la base de données d'apprentissage. Une répétition de cette optimisation conduit donc à venir échantillonner les zones de fortes probabilités de $p(\theta \mid \mathcal{D}_n)$. Cette vision peut être renforcée à la vue des travaux de Staber and Da Veiga (2022) montrant sur leur cas expérimental que les ensembles profonds offrent la meilleure approximation à la référence HMC, considérée comme la référence pour la quantification de l'incertitude d'échantillonnage.

A.2 Une question difficile à appréhender et à trancher !

Dans notre vision, et comme souligné au sein de la section 1.5.2, l'incertitude des paramètres d'un réseau de neurones est composée de l'intrication de l'incertitude d'échantillonnage (provenant du manque de connaissance) et de répétabilité (provenant de la stochasticité lors de l'optimisation). Ces deux sources d'incertitude définissent un "paysage" dans lequel les paramètres d'un réseau de neurones évoluent au cours de l'entraînement et définissent des "zones" plus probables que d'autres après convergence. Une bonne exploration de ce paysage permet alors une meilleure quantification de l'incertitude prédictive issue de l'incertitude des paramètres. Ces deux sources d'incertitude étant intriquées, il est difficile de pouvoir réellement trancher sur le(s) type(s) de source(s) quantifiée(s) par les différentes méthodes de l'état de l'art.

En revanche, il est clair que, par définition, les ensembles profonds permettent une quantification de l'incertitude de répétabilité. Ils peuvent également intégrer une part de l'incertitude d'échantillonnage, motivé par les différents arguments présentés précédemment, sans pour autant avoir de certitude dans un cadre général. En effet, les ensembles profonds constituent une méthode générale applicable à n'importe quelle algorithme d'optimisation. À ce titre, ils sont dépendant de l'algorithme considéré ainsi que de ses hyperparamètres (un choix de régularisation par exemple). A titre d'exemple, dans le cadre d'une optimisation déterministe, il est clair que les ensembles profonds ne permettent pas de "capter" une variabilité due au manque de connaissance au sein de la base de données.

Il faut donc se montrer prudent quant à la source d'incertitude quantifiée par les méthodes de l'état de l'art. Une chose est sûre à la vue de nos travaux (voir notamment le chapitre 3), la quantification de l'incertitude de répétabilité par le biais des ensembles profonds, conduisant à explorer différents modes du "paysage" des paramètres est essentielle afin de fournir un bilan d'incertitude plus robuste ainsi que des performances élevées.

Remarque 19.

Les différents prédicteurs d'un ensemble profond convergent naturellement vers des minima locaux différents les uns des autres du fait de l'initialisation aléatoire de ses paramètres ainsi que du caractère multi-modale du "paysages" des paramètres du réseau considéré.

Remarque 20.

Dans un cadre de métrologie, il semble important de noter qu'à notre connaissance, aucune méthode de l'état de l'art ne fournit des garanties théoriques sur l'erreur d'estimation issue de la quantification de l'incertitude des paramètres.

Annexe B

Méthodes de réduction de dimension

Comme souligné précédemment, une des principales difficultés de la quantification d'incertitude au sein des réseaux de neurones réside dans la dimension élevée de l'espace dans lequel gisent les paramètres du réseau. Afin de réduire la complexité calculatoire, plusieurs méthodes ont été proposées.

n-dernières couches (Fiedler and Lucia, 2023)

Cette approche consiste à implémenter les méthodes de quantification d'incertitude épistémique uniquement sur les n-dernières couches du réseau. Cette approche permet de réduire significativement l'empreinte calculatoire de la méthode implémentée, au risque de fournir une estimation dégradée de l'incertitude.

Distillation de connaissance

La seconde approche consiste à "distiller la connaissance" (Hinton et al., 2015; Gou et al., 2021) d'un ensemble de N réseaux à un simple réseau, et reproduisant directement la distribution fournit par cet ensemble considéré. Cette approche permet de réduire grandement le temps de calcul pour la prédiction d'une nouvelle donnée.

À retenir

Dans un cadre métrologique, il semble difficile de pouvoir quantifier l'incertitude d'un réseau de neurones en se référant à une solution qui repose elle-même sur des réseaux de neurones. Des méthodes statistiques sont à privilégiées.

Elagage de model (Model Pruning)

L'élagage de modèle (communément appelé Model Pruning) (par exemple (Cavalcanti et al., 2016; Martinez, 2021)) permet de réduire le nombre de paramètres d'un réseau de neurones tout en

gardant un niveau de performance constant. Dans (Zhu and Gupta, 2017), les auteurs effectuent une expérimentation poussée sur les capacités d'élagage de certains modèles classiques en apprentissage profond. On note notamment que pour un réseau sous architecture InceptionV3 (Szegedy et al., 2016), les auteurs ont pu réduire le nombre de paramètres de 50% avec une diminution négligeable de la performance globale. Ces techniques peuvent être pratiques afin de rendre scalable certaines méthodologies trop gourmande en temps et/ou en mémoire.

Annexe C

Analyse de l'utilisation de la divergence de Kullback-Leibler comme critère de division au sein du paradigme Split&Merge

Dans cette section, nous analysons l'utilisation de la KL divergence comme critère de division, comme proposé dans le travail de Zhang and Shin (2021).

Définition C.0.1. (divergence KL) Soit P et Q des distributions sur le même ensemble U. Alors la divergence KL de p par rapport à q, notée $D_{KL}(P\|Q)$, est définie comme suit

$$D_{KL}(P\|Q) = \int_U \frac{dP}{d\lambda}(x) \ln \frac{\frac{dP}{d\lambda}(x)}{\frac{dQ}{d\lambda}(x)} d\lambda(x)$$

où $\frac{dP}{d\lambda}$ et $\frac{dQ}{d\lambda}$ sont les densités de P et Q par rapport à une mesure dominante commune (qui existe toujours). Dans le cadre discret, cela revient à

$$D_{KL}(P\|Q) = \sum_{x\in U} P(x) \log \frac{P(x)}{Q(x)}.$$

Lorsque P et Q ont tous deux une densité par rapport à la mesure de Lebesgue, on obtient

$$D_{KL}(P\|Q) = \int_{-\infty}^{\infty} p(x) \ln \frac{p(x)}{q(x)} dx.$$

La divergence de Kullback-Leibler est un cas particulier de la famille de la divergence f. Ce n'est pas une distance à proprement parler puisqu'elle n'est pas symétrique et ne vérifie pas l'inégalité triangulaire. De manière plus générale, la divergence KL devient infinie lorsque Q ne domine pas P,

c'est-à-dire qu'il existe un événement A tel que $P(A) \neq 0$ et $Q(A) = 0$. Cela se produit, par exemple si le support de P n'est pas inclus dans celui de Q. Cette situation se présente pour les réseaux de neurones utilisant des ReLU. Zhang and Shin (2021) proposent de contourner ce problème en proposant une approximation de la ReLU en Leaky-Relu de faible pente δ (par exemple $\delta = 10^{-3}$). Malheureusement, cette approximation conduit toujours à un très grand nombre de composantes dans la mixture propagée augmentant drastiquement la mémoire utilisée. De plus, l'utilisation de la divergence KL au sein du paradigme Split&Merge conduit à une propagation couche par couche (l'algorithme est appliqué sur chaque couche du réseau successivement) conduisant à de nombreuses limitations :

1. dès la sortie de la première couche, chaque composante de la mixture présente une matrice de variance covariance différente. Cela augmente drastiquement l'empreinte mémoire et calculatoire de l'algorithme puisqu'il est nécessaire à partir de la deuxième couche d'effectuer une opération d'inverse par composante,
2. la complexité dépend alors de la largeur du réseau et plus uniquement des dimensions des entrées et des sorties,
3. une potentielle explosion du nombre de composantes, obligeant à effectuer une étape de fusion très coûteuse en temps,
4. le critère de fusion basée sur la divergence KL fait intervenir des logarithmes de déterminants de matrices souvent singulières,
5. le critère de division ne quantifie pas directement l'erreur effectuée entre la vraie distribution de sortie et la distribution estimée.

Le choix d'un critère de Wasserstein permet de répondre à l'intégralité de ces problématiques, tout en fournissant des garanties théoriques de convergence.

Annexe D

Critère de fusion basée sur la distance 2-Wasserstein

Bien que nous ne recommandions pas l'utilisation d'une étape de fusion, nous proposons ici un critère de fusion basé une nouvelle fois sur la distance de 2-Wasserstein. Nous rappelons que l'étape de fusion permet de réduire le nombre de composantes dans la mixture en fusionnant les composantes similaires après propagation au sein de la fonction non linéaire.

Soit $p(\boldsymbol{Y}) = \sum_{k=1}^{M} w_k p_k(\boldsymbol{Y})$, avec $p_k(\boldsymbol{Y}) = \mathcal{N}(\boldsymbol{\mu_k}, \boldsymbol{\Sigma_k})$, une mixture de gaussiennes et $p^{(i,j)}(\boldsymbol{Y}) = \sum_{k=1}^{M-1} w_k p_k^{(i,j)}(\boldsymbol{Y})$ avec $p_k^{(i,j)}(\boldsymbol{Y}) = \mathcal{N}(\boldsymbol{\mu_k^{(i,j)}}, \boldsymbol{\Sigma_k^{(i,j)}})$ le mélange résultant de la fusion des i-ième et j-ième composantes de $p(\boldsymbol{Y})$. Intuitivement, nous voulons fusionner des composantes gaussiennes proches en termes de moyenne et de variance. Ici, nous mettons en parallèle notre critère de détection de non-linéarité en utilisant la métrique de 2-Wasserstein. Sans perte de généralité, nous pouvons supposer que $(i, j) = (M-1, M)$. On obtient alors :

$$C_{M-1,M}^{merge} = W_2^2(p, p^{(M-1,M)}) \leq w_{M-1} W_2^2(p_{M-1}, p_{M-1}^{(M-1,M)}) + w_M W_2^2(p_M, p_{M-1}^{(M-1,M)}) \tag{D.1}$$

Les couples gaussiens présentant le critère de fusion le plus bas (et sous un seuil spécifié) sont fusionnés. La procédure de fusion est coûteuse en temps et, par construction, dégrade la qualité de la prédiction. Pour ces raisons, il est recommandé d'appliquer une étape de fusion uniquement en cas d'atteinte des limites de la mémoire matérielle.

Nous rappelons ici que deux distributions gaussiennes (i et j de poids, moyennes et matrice de covariances respectives w_i, $\boldsymbol{\mu}_i$, $\boldsymbol{\Sigma}_i$ et w_j, $\boldsymbol{\mu}_j$, $\boldsymbol{\Sigma}_j$) peuvent être fusionnées en suivant les formules suivantes :

$$
\begin{aligned}
w_{ij} &= w_i + w_j \\
\boldsymbol{\mu}_{ij} &= \frac{w_i \boldsymbol{\mu}_i + w_j \boldsymbol{\mu}_j}{w_i + w_j} \\
\boldsymbol{\Sigma}_{ij} &= \frac{w_i \boldsymbol{\Sigma}_i + w_j \boldsymbol{\Sigma}_j}{w_i + w_j} + \frac{w_i w_j \left(\boldsymbol{\mu}_i - \boldsymbol{\mu}_j\right) \left(\boldsymbol{\mu}_i - \boldsymbol{\mu}_j\right)^T}{\left(w_i + w_j\right)^2}
\end{aligned}
\tag{D.2}
$$

Annexe E

Comparaison des variantes de WGMprop

Notre méthodologie WGMprop repose sur plusieurs choix structurants qui peuvent être comparés et discutés. La méthode que nous proposons considère le réseau de neurones comme une boîte noire et la propagation est donc effectuée de manière *directe* en utilisant l'échantillonnage UT pour l'estimation des deux premiers moments de la distribution de sortie. Nous souhaitons effectuer une comparaison des implémentations couche par couches de notre approche, rendu possible par la petite taille de la tache considérée (faible dimension d'entrée et de sortie du réseau et faible largeur des couches intermédiaires). Nous étudions en outre l'impact des estimations analytiques des moments à l'aide du théorème 2.1.9. Ainsi, nous fournissons dans cette section les résultats de l'expérience MNIST pour les différentes variantes de WGMprop présentées dans le tableau E.1 :

— WGMProp@LPN : propagation par couches d'une MG à l'aide d'une formule analytique (pas d'estimation des covariances),

— WGMProp@HPN : propagation par couches d'une MG utilisant une formule analytique (estimation des covariances),

— WGMProp@UT : propagation par couches d'une MG utilisant l'échantillonnage UT

TABLE E.1 – Caractéristiques de propagation pour les variantes de cadre WGMprop.

	Propagation		PDF de sortie		Estimation des moments		Covariance		Critère	
	Couche/couche	Réseau entier	Simple Gaussienne	Mixture	Échantillonnage UT	Analytique	Avec	Sans	Wasserstein	KL
WGMProp@LPN	✓	✗	✗	✓	✗	✓	✗	✓	✓	✗
WGMProp@HPN	✓	✗	✗	✓	✗	✓	✓	✗	✓	✗
WGMProp@UT	✓	✗	✗	✓	✓	✗	✓	✗	✓	✗

Le tableau E.2 présente l'intégralité des résultats pour tous les bruit et toutes les intensités avec $T_{split} = 10^{-3}$ pour toutes les méthodes.

TABLE E.2 – Comparaison des performances pour tous les types de bruit et d'intensité. Les écarts types sont présentés en ().

			CRITÈRE DE PERFORMANCE						
Bruit	**Intensité**	**Méthode**	**# Gaussienne**	**2W**	**KL**	**MEAN**	**STD**	**IOU_{95}**	**TIME (s)**
Gaussien	I_1	WGMprop@LPN	10.14(6.18)	0.389(0.351)	0.29(0.26)	0.15(0.12)	1227.99(1576.32)	0.167(0.117)	0.79(0.39)
		WGMprop@HPN	10.14(6.18)	0.005(0.006)	0.02(0.02)	0.00(0.00)	0.61(0.56)	0.952(0.036)	1.48(0.81)
		WGMprop@UT	1.91(2.15)	0.008(0.008)	0.02(0.02)	0.08(0.09)	3.98(2.90)	0.932(0.038)	0.30(0.20)
	I_2	WGMprop@LPN	10.14(6.18)	0.389(0.351)	0.29(0.26)	0.15(0.12)	1227.99(1576.32)	0.167(0.117)	0.79(0.39)
		WGMprop@HPN	10.14(6.18)	0.005(0.006)	0.02(0.02)	0.00(0.00)	0.61(0.56)	0.952(0.036)	1.48(0.81)
		WGMprop@UT	1.91(2.15)	0.008(0.008)	0.02(0.02)	0.08(0.09)	3.98(2.90)	0.932(0.038)	0.30(0.20)
	I_3	WGMprop@LPN	10.14(6.18)	0.389(0.351)	0.29(0.26)	0.15(0.12)	1227.99(1576.32)	0.167(0.117)	0.79(0.39)
		WGMprop@HPN	10.14(6.18)	0.005(0.006)	0.02(0.02)	0.00(0.00)	0.61(0.56)	0.952(0.036)	1.48(0.81)
		WGMprop@UT	1.91(2.15)	0.008(0.008)	0.02(0.02)	0.08(0.09)	3.98(2.90)	0.932(0.038)	0.30(0.20)
Flou	I_1	WGMprop@LPN	12.52(9.95)	0.150(0.325)	0.74(0.65)	0.32(0.21)	13.21(11.50)	0.794(0.092)	0.92(0.60)
		WGMprop@HPN	10.04(4.13)	0.007(0.014)	0.11(0.10)	0.01(0.01)	0.19(0.26)	0.983(0.016)	1.40(0.51)
		WGMprop@UT	7.46(1.79)	0.011(0.024)	0.13(0.11)	0.04(0.04)	0.68(0.52)	0.970(0.021)	0.76(0.15)
	I_2	WGMprop@LPN	155.03(104.93)	0.410(0.229)	1.76(14.55)	1.20(9.27)	2.58(4.03)	0.945(0.058)	9.38(5.88)
		WGMprop@HPN	64.16(17.25)	0.017(0.016)	0.34(0.24)	0.05(0.04)	0.15(0.25)	0.980(0.017)	8.20(2.16)
		WGMprop@UT	43.25(11.87)	0.034(0.033)	0.39(0.24)	0.20(0.17)	0.52(0.70)	0.961(0.026)	3.73(0.98)
	I_3	WGMprop@LPN	235.25(190.77)	0.498(0.428)	10.11(41.46)	4.08(12.83)	7.54(8.57)	0.878(0.106)	14.04(9.96)
		WGMprop@HPN	88.19(23.17)	0.030(0.023)	0.47(0.31)	0.10(0.07)	0.43(0.43)	0.952(0.030)	11.81(3.05)
		WGMprop@UT	62.61(15.17)	0.053(0.040)	0.62(0.33)	0.25(0.19)	0.95(0.93)	0.927(0.042)	5.46(1.27)
Contraste	I_1	WGMprop@LPN	14.06(9.89)	0.186(0.348)	0.85(0.81)	0.33(0.27)	10.76(5.89)	0.803(0.082)	1.02(0.61)
		WGMprop@HPN	11.97(5.01)	0.018(0.031)	0.17(0.17)	0.02(0.02)	0.29(0.68)	0.977(0.025)	1.63(0.61)
		WGMprop@UT	8.13(2.58)	0.030(0.052)	0.20(0.19)	0.06(0.06)	0.85(0.71)	0.958(0.035)	0.79(0.20)
	I_2	WGMprop@LPN	281.92(39.79)	0.267(0.125)	0.63(1.17)	0.09(0.08)	0.24(0.22)	0.992(0.008)	16.82(2.34)
		WGMprop@HPN	84.25(13.48)	0.019(0.012)	1.45(2.21)	0.07(0.06)	0.06(0.07)	0.993(0.007)	10.58(1.66)
		WGMprop@UT	68.63(12.98)	0.040(0.029)	1.47(2.34)	0.17(0.15)	0.23(0.19)	0.985(0.015)	5.65(1.03)
	I_3	WGMprop@LPN	670.06(148.56)	0.551(0.316)	32.22(42.94)	22.27(30.94)	13.27(20.02)	0.790(0.131)	35.92(6.72)
		WGMprop@HPN	108.17(9.88)	0.022(0.012)	1.46(3.43)	0.11(0.07)	0.12(0.10)	0.989(0.006)	14.08(1.30)
		WGMprop@UT	97.02(9.56)	0.044(0.026)	1.23(3.52)	0.24(0.16)	0.31(0.24)	0.980(0.012)	8.02(0.77)

Les prédictions de l'ensemble des variantes de WGMprop restent très précises : inférieur à 0.25%. L'approche analytique WGMprop@HPN atteint la valeur $\mathbf{IOU_{95}}$ la plus élevée pour tous les modèles de bruit étudiés. WGMprop@HPN produit également les valeurs d'erreur les plus faibles pour 2W, KL, $\mathbf{MAPE}_{\text{MEAN}}$ et $\mathbf{MAPE}_{\text{STD}}$; sauf pour le bruit de contraste où WGMprop@FN est légèrement plus performant sur $\mathbf{MAPE}_{\text{MEAN}}$ et $\mathbf{MAPE}_{\text{STD}}$ au prix d'un temps de prédiction élevé. En termes de temps de prédiction moyen, WGMprop@UT est le plus rapide pour l'ensemble de l'expérience. Cette vitesse plus élevée est clairement corrélée avec le nombre réduit de composantes gaussiennes dans le mélange. On note également les très faibles performances de WGMprop@LPN pour un bruit gaussien. Cette variante présente également un grand nombre de composants dans la mixture au regard des autres méthodes couche par couche où le seuil vaut 10^{-3}. Ces éléments suggèrent l'importance de l'estimation des covariances lors de la propagation afin d'obtenir une bonne estimation de la sortie.

De plus, la figure E.1 présente les diagrammes en boîte du MAPE de plusieurs percentiles prédits (1, 2.5, 25, 25, 97.5 et 99).

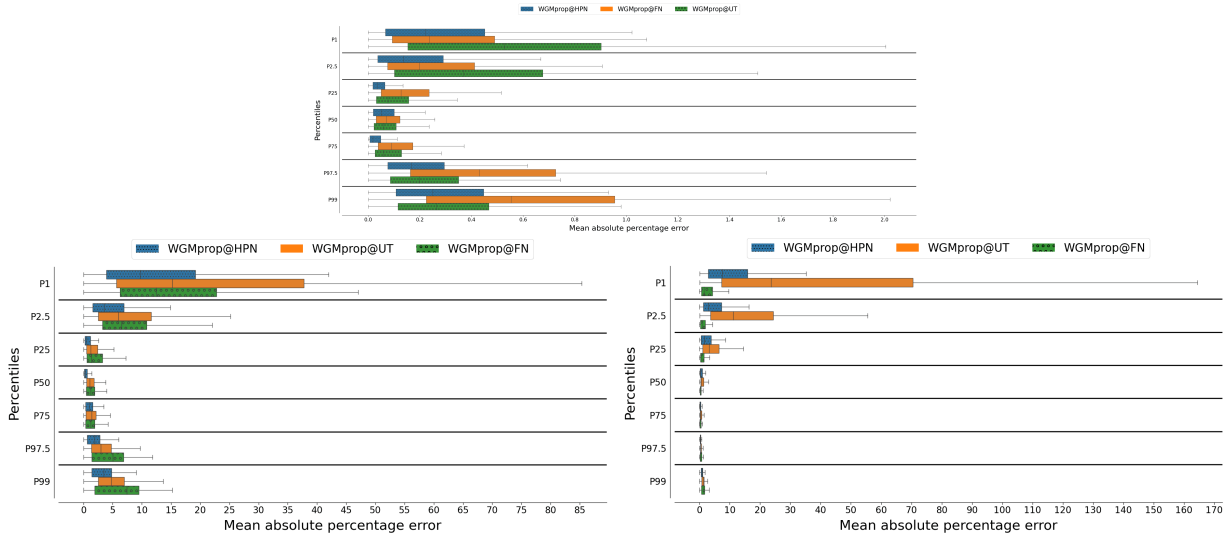


FIGURE E.1 – Erreur moyenne en pourcentage absolu sur les percentiles 1, 2,5, 25, 50, 75, 97,5 et 99 entre les méthodes WGMprop et la référence MC pour un bruit de contraste d'intensité I_3 et $K = 1$ (**Haut**), $K = 2$ (**Milieu**) et $K = 3$ (**Bas**).

Ces diagrammes en boîte présentent les distributions MAPE sur les différents centiles de la PDF de sortie pour les 3 méthodes : WGMprop@HPN, WGMprop@FN (notre méthode WGMprop) et WGMprop@UT. Tout d'abord, les méthodes proposées présentent des performances comparables avec un chevauchement des distributions MAPE pour tous les centiles estimés. Les méthodes présentent des erreurs MAPE plus importantes à mesure que l'on se rapproche des queues de la distribution (P1, P99), mais se comportent de manière similaire. WGMprop@HPN présente à la fois les valeurs médianes les plus faibles et la plus petite dispersion d'erreurs de P1 à P99. WGMprop@FN présente également d'excellente performance sur l'ensemble des bruits a haute intensité. Nous pouvons observer visuellement ces performances au sein de la figure E.3

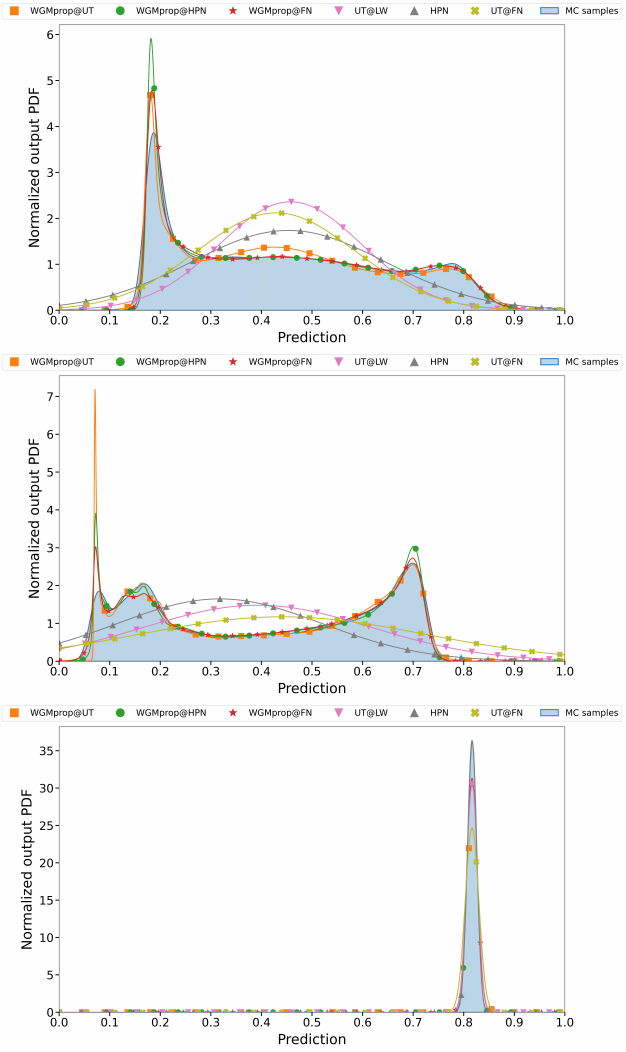


FIGURE E.2 – PDF de sortie estimées (marginale principale) pour l'échantillon n°58 corrompu par un bruit de flou (**Haut**), un bruit de contraste (**Milieu**) et un bruit additif gaussien (**Bas**) (intensité I_3 et $K = 1$), la référence MC est remplie en bleu.

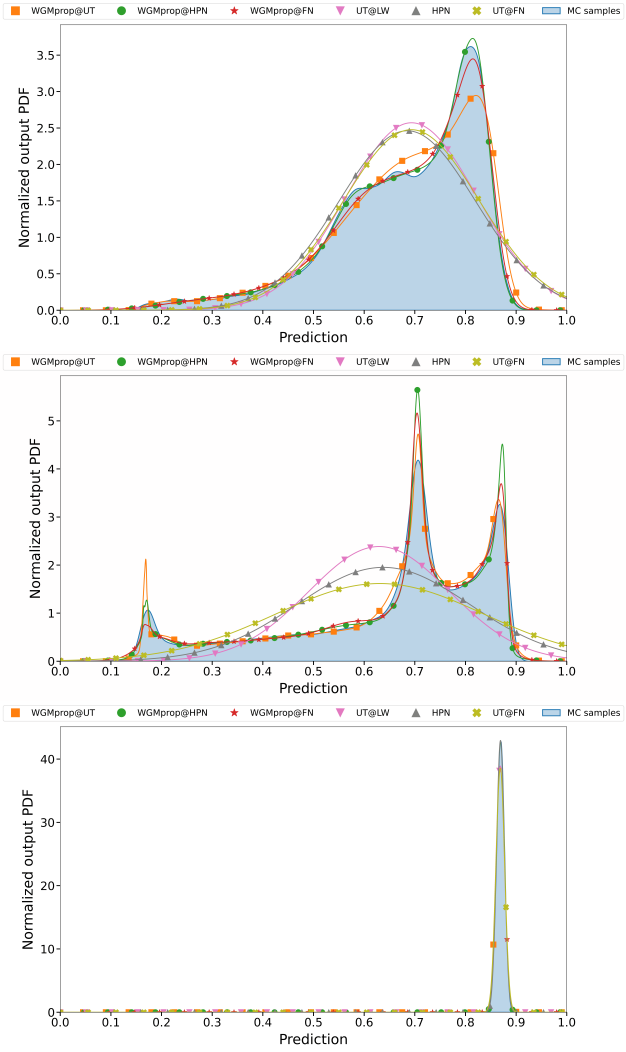


FIGURE E.3 – PDF de sortie estimées (marginale principale) pour l'échantillon n°181 corrompu par un bruit de flou (**Haut**), un bruit de contraste (**Milieu**) et un bruit additif gaussien (**Bas**) (intensité I_3 et $K = 1$), la référence MC est remplie en bleu.

L'ensemble de ces éléments nous amène à conclure que l'approche WGMprop@HPN (estimation analytique des moments) fournit les meilleures performances en augmentant légèrement le coût calculatoire (à la vue de WGMprop@UT utilisant l'échantillonnage UT pour estimer les moments propagés). Cependant, dans un paradigme de propagation par couches tel que WGMprop@HPN et WGMprop@UT, la propagation peut devenir infaisable pour des réseaux de neurones plus profonds et plus larges. En effet, une propagation par couches induit de nombreux coûts de calcul supplémentaires, comme détaillé en section C.Pour toutes ces raisons, notre approche de réseau complet WGMprop@FN, basée sur l'échantillonnage UT et un critère de Wasserstein, est à privilégier pour sa généricité, sa simplicité et son faible coût mémoire.

Annexe F

Résultats complémentaire de WGMprop

F.1 Base de données CIFAR-10

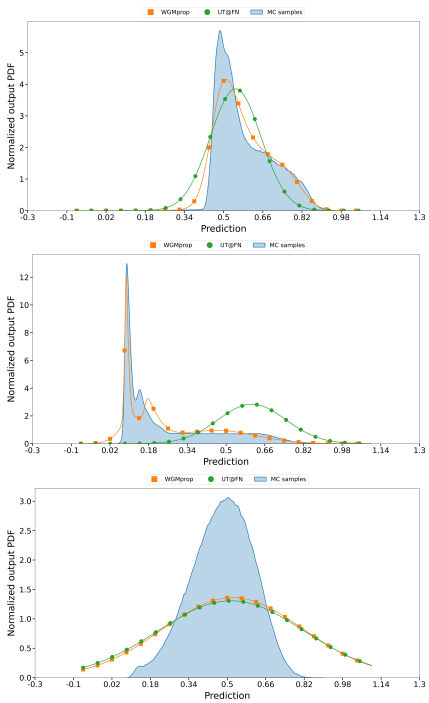


FIGURE F.1 – PDF de sortie estimés (marge principale) pour l'échantillon n° 2 de l'ensemble de données CIFAR10 corrompu par un noyau de flou (**Top**), un bruit de contraste (**Middle**) et un bruit additif à distribution gaussienne (**Bottom**) (intensité I_3). La référence de Monte Carlo est indiqué en bleu.

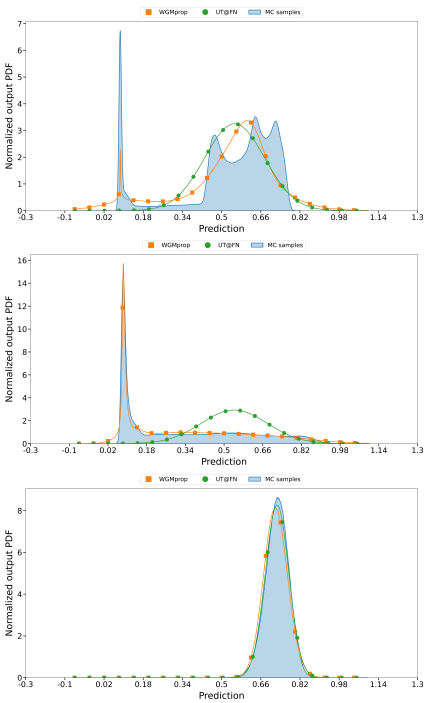


FIGURE F.2 – PDF de sortie estimés (marge principale) pour l'échantillon n°89 de l'ensemble de données CIFAR10 corrompu par un noyau de flou (**Top**), un bruit de contraste (**Middle**) et un bruit additif à distribution gaussienne (**Bottom**) (intensité I_3). La référence de Monte Carlo est indiqué en bleu.

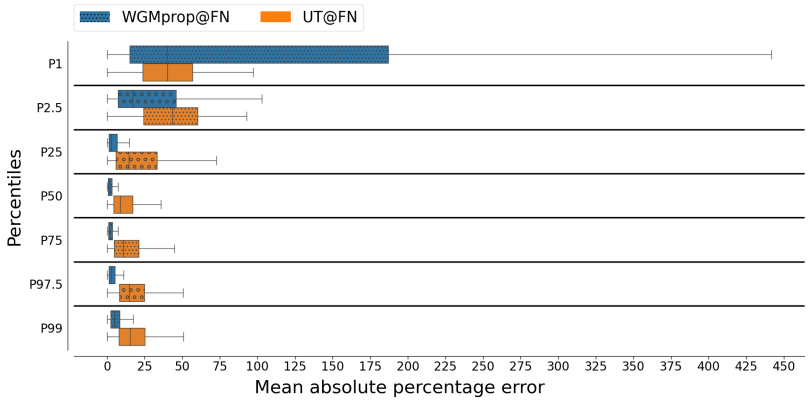


FIGURE F.3 – Diagrammes en boîte MAPE pour l'ensemble de données CIFAR10 et pour les percentiles 1, 2,5, 25, 50, 75, 97,5 et 99. Les images d'entrée ont été dégradées par un noyau de contraste important (I_3).

F.2 CIFAR-100

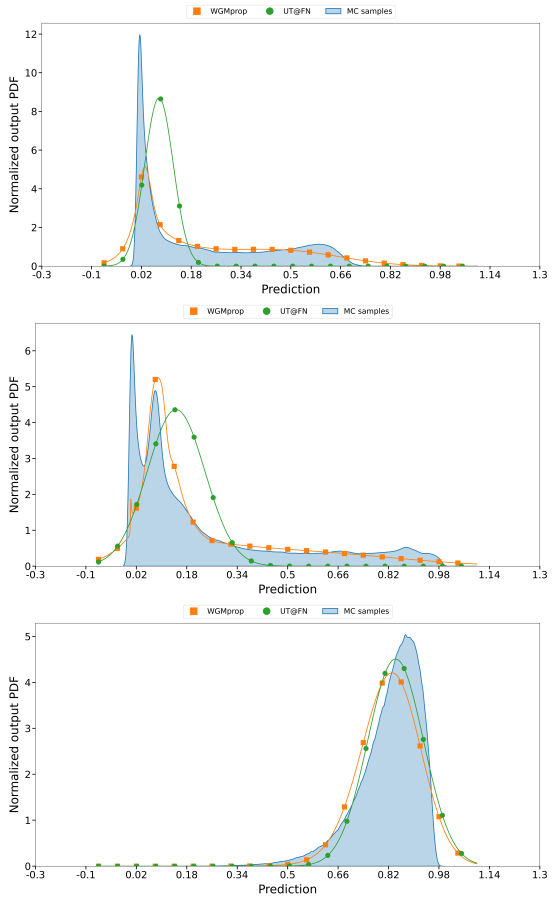


FIGURE F.4 – PDF de sortie estimés (marge principale) pour l'échantillon n° 2 de l'ensemble de données CIFAR100 corrompu par un noyau de flou (**Top**), un bruit de contraste (**Middle**) et un bruit additif à distribution gaussienne (**Bottom**) (intensité I_3). Le PDF de sortie estimé à l'aide de la propagation de Monte Carlo (référence) est indiqué en bleu.

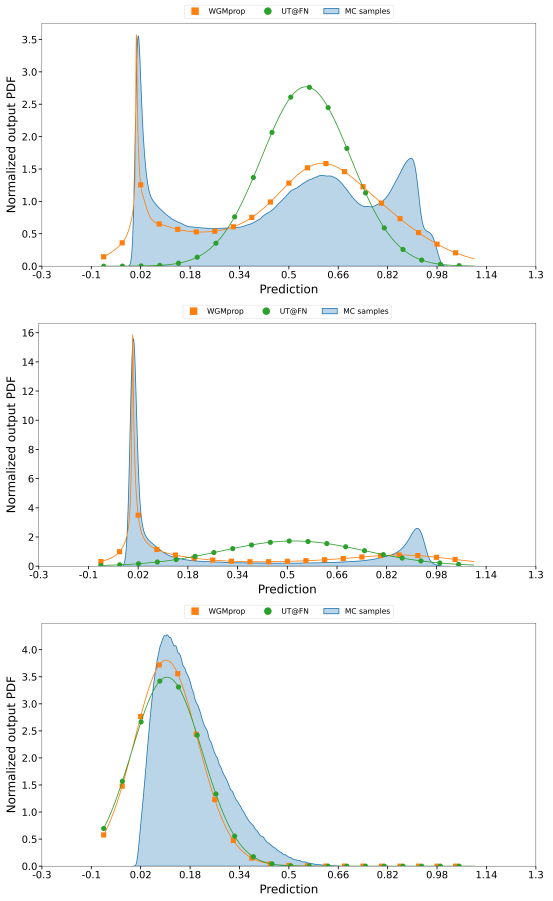


FIGURE F.5 – PDF de sortie estimés (marge principale) pour l'échantillon n°89 de l'ensemble de données CIFAR100 corrompu par un noyau de flou (**Top**), un bruit de contraste (**Middle**) et un bruit additif à distribution gaussienne (**Bottom**) (intensité I_3). Le PDF de sortie estimé à l'aide de la propagation de Monte Carlo (référence) est indiqué en bleu.

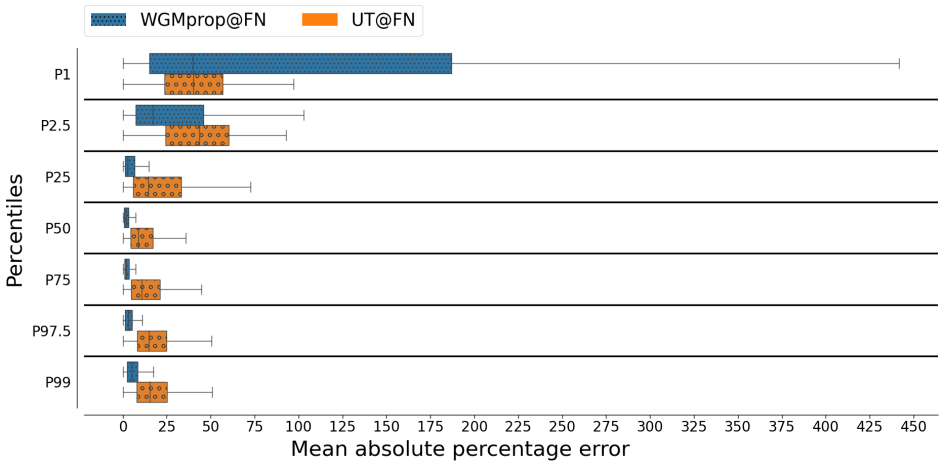


FIGURE F.6 – Diagrammes en boîte MAPE pour l'ensemble de données CIFAR100 et pour les percentiles 1, 2,5, 25, 50, 75, 97,5 et 99. Les images d'entrée ont été dégradées par un noyau de contraste important (I_3).

F.3 Camelyon

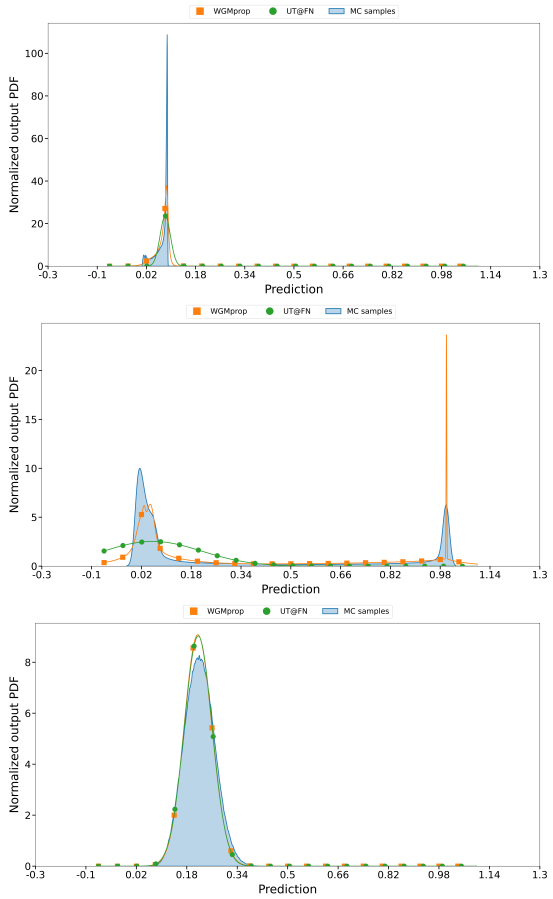


FIGURE F.7 – PDF de sortie estimés (marge principale) pour l'échantillon n°2 de l'ensemble de données Camelyon corrompu par un noyau de flou (**Top**), un bruit de contraste (**Middle**) et un bruit additif à distribution gaussienne (**Bottom**) (intensité I_3). Le PDF de sortie estimé à l'aide de la propagation de Monte Carlo (référence) est indiqué en bleu.

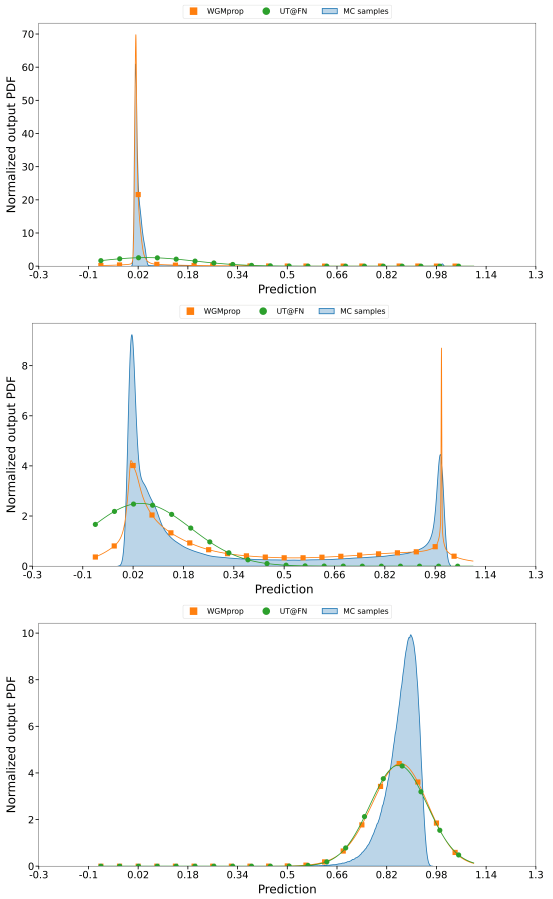


FIGURE F.8 – PDF de sortie estimés (marge principale) pour l'échantillon n°89 de l'ensemble de données Camelyon corrompu par un noyau de flou (**Top**), un bruit de contraste (**Middle**) et un bruit additif à distribution gaussienne (**Bottom**) (intensité I_3). Le PDF de sortie estimé à l'aide de la propagation de Monte Carlo (référence) est indiqué en bleu.

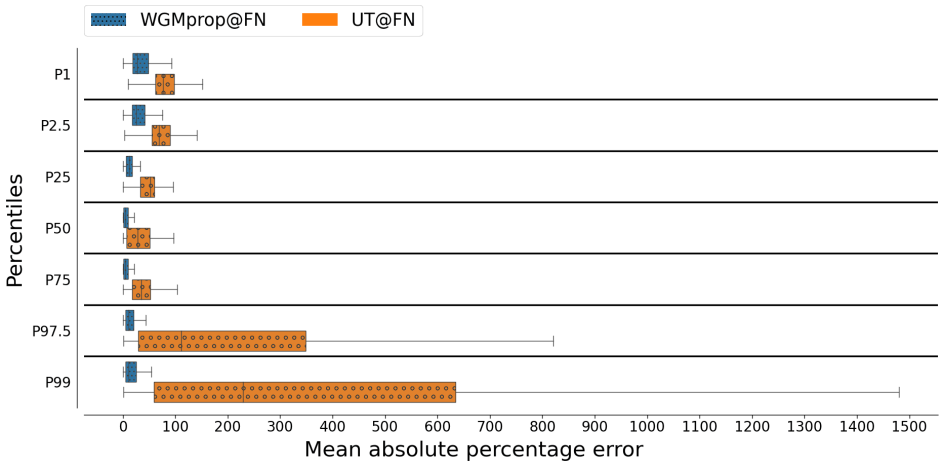


FIGURE F.9 – Diagrammes en boîte du MAPE pour l'ensemble de données Camelyon et pour les percentiles 1, 2,5, 25, 50, 75, 97,5 et 99. Les images d'entrée ont été dégradées par un noyau de contraste important (I_3)..

Annexe G

Formules analytiques de propagation des deux premiers moments d'une distribution gaussienne à travers la fonction d'activation Leaky ReLU (Maas et al., 2013) et ELU (Clevert et al., 2015)

G.1 Fonction d'activation Leaky ReLU

Proposition G.1.1. *Soit X une variable aléatoire avec $X \sim \mathcal{N}_d(\boldsymbol{\mu}, \boldsymbol{\Sigma})$, $\boldsymbol{\mu} \in \mathbb{R}^d$, $\boldsymbol{\Sigma} \in \mathbb{S}_d^+(\mathbb{R})$ avec rang$(\boldsymbol{\Sigma}) = r$, $r \in \mathbb{N}^*$ et f be the leaky rectified linear function de paramètre $\lambda \in [0, 1[$, alors pour la variable aléatoire $\boldsymbol{Y} = (Y_1, ..., Y_d) = f(\boldsymbol{X}) = (f(X_1), ..., f(X_d))$, $\forall k, k' \in \{1...d\}$:*

$$
\begin{aligned}
\mathbb{E}[Y_k] &= (1-\lambda)c_k\phi_k + (1-\lambda)\mu_k\Phi_k + \lambda\mu_k && \text{(G.1)}\\
\mathbb{E}[Y_k^2] &= (1-\lambda^2)(\mu_k^2 + c_k^2)\Phi_k + (1-\lambda^2)\mu_k c_k\phi_k + \lambda^2(c_k^2 + \mu_k^2) && \text{(G.2)}\\
\mathbb{E}[Y_k Y_{k'}] &= (1-\lambda)^2 c_k \alpha \phi_{k'}\phi_{NS}\\
&+ (\mu_k\mu_{k'} + c_k\beta)\left((1-\lambda)\Phi_{k'} - (1-\lambda)^2\Phi_{kk'} + \lambda(1-\lambda)\Phi_k + \lambda^2\right)\\
&+ (1-\lambda)^2\mu_k c_{k'}\phi_{k'}\Phi_S\\
&+ (1-\lambda)^2\mu_{k'}c_k\phi_k\Phi_{NS}\\
&+ \lambda(1-\lambda)(\mu_{k'}c_k\phi_k + \mu_k c_{k'}\phi_{k'}) && \text{(G.3)}
\end{aligned}
$$

où : $\mu_k = \mathbb{E}[X_k]$, la moyenne de la k^e composante de la variable aléatoire $\boldsymbol{X}$, $Q_r = (q_k)_{k\in\{1...d\}} \in$

$\mathcal{M}_{r,d}(\mathbb{R})$ les r premières lignes de Q, telles que $\Sigma = Q^T \Lambda Q = Q_r^T \Lambda_r Q_r$, and $\Lambda_r \in \mathcal{D}_r(\mathbb{R})$. On a $c_k = \left\|\sqrt{\Lambda_r} q_k\right\|$, $c_{k-k'} = \left\|\sqrt{\Lambda_r}(q_k - q_{k'})\right\|$, $\beta = \frac{c_k^2 + c_j^2 - c_{k-k'}^2}{2c_k}$, $\alpha = \sqrt{c_j^2 - \beta^2}$. Enfin, on note $\phi_k = \phi_{0,1}\left(\frac{\mu_k}{c_k}\right)$, $\Phi_{k'} = \Phi_{0,1}\left(\frac{\mu_{k'}}{c_{k'}}\right)$, $\Phi_{NS} = \Phi_{0,1}\left(\frac{\mu_{k'}}{\alpha} - \frac{\beta \mu_k}{\alpha c_k}\right)$, $\Phi_S = \Phi_{0,1}\left(\frac{c_{k'}\mu_k}{c_k \alpha} - \frac{\beta \mu_{k'}}{\alpha c_{k'}}\right)$, $\phi_{NS} = \phi_{0,1}\left(\frac{\beta \mu_{k'}}{\alpha c_{k'}} - \frac{\mu_k c_{k'}}{c_k \alpha}\right)$ and $\Phi_{kk'} = \Phi_{\begin{bmatrix}0\\0\end{bmatrix}, \begin{bmatrix}1 & -\frac{\beta}{c_{k'}} \\ -\frac{\beta}{c_{k'}} & 1\end{bmatrix}}\left(\left[\frac{\mu_{k'}}{c_{k'}}, -\frac{\mu_k}{c_k}\right]^T\right)$ and ϕ et Φ sont la densité de probabilité et la fonction de répartition de la distribution normale centrée réduite.

G.2 Fonction d'activation ELU

Proposition G.2.1. *Soit X une variable aléatoire, avec $X \sim \mathcal{N}_d(\boldsymbol{\mu}, \boldsymbol{\Sigma})$, $\boldsymbol{\mu} \in \mathbb{R}^d$, $\boldsymbol{\Sigma} \in \mathbb{S}_d^+(\mathbb{R})$ avec rang$(\boldsymbol{\Sigma}) = r$, $r \in \mathbb{N}^*$ et f l'unité exponentielle linéaire, alors pour la variable aléatoire $\boldsymbol{Y} = (Y_1, ..., Y_d) = f(\boldsymbol{X}) = (f(X_1), ..., f(X_d))$, $\forall k, k' \in \{1...d\}$:*

$$\mathbb{E}[Y_k] = c_k \phi_k + (\mu_k + \lambda)\Phi_k + \lambda e^{\mu_k + \frac{c_k^2}{2}}(1 - \Phi_{kc_k}) - \lambda \tag{G.4}$$

$$\begin{aligned}\mathbb{E}[Y_k^2] &= (\mu_i^2 + c_i^2 - \lambda^2)\Phi_i + \mu_i c_i \phi_i + \lambda^2 e^{2(c_i^2 + \mu_i)}(1 - \Phi_{2c_i}) - 2\lambda^2 e^{\mu_i + \frac{c_i^2}{2}}(1 - \Phi_{c_i}) \\ &+ \lambda^2\end{aligned} \tag{G.5}$$

$$\begin{aligned}\mathbb{E}[Y_k Y_{k'}] &= c_i \alpha \phi_j \phi_{NS} \\ &+ (\mu_i \mu_j + c_i \beta + \lambda \mu_i)\, \Phi_j \\ &- \left(\mu_i \mu_j + c_i \beta + \lambda(\mu_i + \mu_j) + \lambda^2\right) \Phi_{ij} \\ &+ (\mu_i c_j \phi_j + \lambda c_j \phi_j + \frac{\lambda \beta c_i}{c_j}((\phi_j - e^{\mu_j + \frac{c_j^2}{2}} \phi_{c_j}))\Phi_S \\ &+ (\mu_j c_i \phi_i + \lambda c_i \phi_i + \lambda \beta(\phi_i - e^{\mu_i + \frac{c_i^2}{2}} \phi_{c_i}))\Phi_{NS} \\ &+ \lambda c_i e^{\mu_j + \frac{c_j^2}{2}} \phi_{\beta_i} \Phi_\alpha \\ &+ \lambda c_j e^{\mu_i + \frac{c_i^2}{2}} \phi_a \Phi_e \\ &+ \lambda^2 e^{\mu_i + \frac{c_i^2}{2}} \Phi_{c_i} \\ &- \lambda e^{\mu_j + \frac{c_j^2}{2}} (\mu_i + \beta c_i) \Phi_{c_j} \\ &+ \lambda e^{\mu_i + \frac{c_i^2}{2}} (\lambda + \beta c_i + \mu_j) \Phi_A \\ &+ \lambda e^{\mu_j + \frac{c_j^2}{2}} (\lambda + \beta c_i + \mu_i)(\Phi_{\beta_i} + \Phi_B) \\ &+ \lambda^2 e^{\mu_i + \mu_j + \frac{c_i^2 + c_j^2}{2} + c_i \beta}(1 - \Phi_{\beta + c_i} - \Phi_C) \\ &- \lambda(\lambda e^{\mu_i + \frac{c_i^2}{2}} + \lambda e^{\mu_j + \frac{c_j^2}{2}} + c_i \phi_i + c_j \phi_j - \lambda)\end{aligned} \tag{G.6}$$

où : $\mu_k = \mathbb{E}[X_k]$, la moyenne de la k^e composante de la variable aléatoire $\boldsymbol{X}$, $Q_r = (q_k)_{k \in \{1...d\}} \in$

$\mathcal{M}_{r,d}(\mathbb{R})$ les r premières lignes de Q, telles que $\Sigma = Q^T \Lambda Q = Q_r^T \Lambda_r Q_r$, and $\Lambda_r \in \mathcal{D}_r(\mathbb{R})$. Alors, on a $c_k = \left\| \sqrt{\Lambda_r} q_k \right\|$, $c_{k-k'} = \left\| \sqrt{\Lambda_r}(q_k - q_{k'}) \right\|$, $\beta = \frac{c_k^2 + c_j^2 - c_{k-k'}^2}{2c_k}$, $\alpha = \sqrt{c_j^2 - \beta^2}$. Enfin, on note $\phi_k = \phi_{0,1}\left(\frac{\mu_k}{c_k}\right)$, $\Phi_{k'} = \Phi_{0,1}\left(\frac{\mu_{k'}}{c_{k'}}\right)$, $\Phi_{NS} = \Phi_{0,1}\left(\frac{\mu_{k'}}{\alpha} - \frac{\beta \mu_k}{\alpha c_k}\right)$, $\Phi_S = \Phi_{0,1}\left(\frac{c_{k'}\mu_k}{c_k \alpha} - \frac{\beta \mu_{k'}}{\alpha c_{k'}}\right)$, $\phi_{NS} = \phi_{0,1}\left(\frac{\beta \mu_{k'}}{\alpha c_{k'}} - \frac{\mu_k c_{k'}}{c_k \alpha}\right)$, $\Phi_{kk'} = \Phi_{\begin{bmatrix}0\\0\end{bmatrix}, \begin{bmatrix}1 & -\frac{\beta}{c_{k'}} \\ -\frac{\beta}{c_{k'}} & 1\end{bmatrix}}\left(\left[\frac{\mu_{k'}}{c_{k'}}, -\frac{\mu_k}{c_k}\right]^T\right)$, $\phi_a = \phi_{0,1}\left(\frac{\mu_j + \beta c_i}{c_j}\right)$, $\Phi_{\beta_i} = \Phi_{0,1}\left(\frac{\mu_i}{c_i} + \beta\right)$, $\Phi_e = \Phi_{0,1}\left(\frac{\beta}{\alpha}\frac{\mu_j + \beta c_i}{c_j} - \frac{c_j \mu_i}{\alpha c_i} - \frac{c_j c_i}{\alpha}\right)$, $\Phi_\alpha = \Phi_{0,1}\left(\frac{\beta \mu_i}{\alpha c_i} - \frac{\mu_j}{\alpha} - \alpha\right)$, $\Phi_{\beta + c_i} = \Phi_{0,1}(\frac{\mu_i}{c_i} + c_i + \beta)$, $\Phi_{c_i} = \Phi_{0,1}\left(\frac{\mu_i}{c_i} + c_i\right)$, $\Phi_{2c_i} = \Phi_{0,1}\left(\frac{\mu_i}{c_i} + 2c_i\right)$, $\Phi_A = \Phi_{\begin{bmatrix}0\\0\end{bmatrix}, \begin{bmatrix}1 & -\frac{\beta}{c_j} \\ -\frac{\beta}{c_j} & 1\end{bmatrix}}\left(\left[\frac{\mu_j + \beta c_i}{c_j}, -\frac{\mu_i}{c_i} - c_i\right]^T\right)$, $\Phi_B = \Phi_{\begin{bmatrix}0\\0\end{bmatrix}, \begin{bmatrix}1 & -\frac{\beta}{c_j} \\ -\frac{\beta}{c_j} & 1\end{bmatrix}}$ et $\Phi_C = \Phi_{\begin{bmatrix}0\\0\end{bmatrix}, \begin{bmatrix}1 & -\frac{\beta}{c_j} \\ -\frac{\beta}{c_j} & 1\end{bmatrix}}\left(\left[\frac{\mu_j + c_j^2 + \beta c_i}{c_j}, -(\frac{\mu_i}{c_i} + c_i + \beta)\right]^T\right)$,

Finalement, ϕ et Φ sont la densité de probabilité et la fonction de répartition de la distribution normale centrée réduite.

G.3 Preuve pour la fonction d'activation Leay-ReLU

G.3.1 Estimation de la moyenne

Notons f_i telle que :

$$f_i \colon \mathbb{R}^n \to \mathbb{R}_+$$
$$x \mapsto \begin{cases} x_i \text{ if } x_i \geq 0 \\ \lambda x_i \text{ sinon, } \lambda \in [0,1] \end{cases}$$

$$\begin{aligned} \mathbb{E}[Y_i] &= \mathbb{E}_{z \sim p(Z)}\left[f_i(Q_r z + \mu)\right] \\ &= \int_{-\infty}^{+\infty} f_i(Q_r z + \mu)\phi_{0,\Lambda_r}(z)dz \end{aligned} \tag{G.7}$$

Notons

$$Z_i^+ = \left\{ z \in \mathbb{R}^r : z^T q_i + \mu_i \geq 0 \right\},$$

et :

$$Z_i^- = \left\{ z \in \mathbb{R}^r : z^T q_i + \mu_i < 0 \right\},$$

Alors :

$$\begin{aligned} \mathbb{E}[Y_i] &= \int_{Z_i^+} f_i(Q_r z + \mu)\phi_{0,\Lambda_r}(z)dz + \int_{Z_i^-} f_i(Q_r z + \mu)\phi_{0,\Lambda_r}(z)dz \\ &= \int_{Z_i^+} (z^T q_i + \mu_i)\phi_{0,\Lambda_r}(z)dz + \lambda \int_{Z_i^-} (z^T q_i + \mu_i)\phi_{0,\Lambda_r}(z)dz \end{aligned} \tag{G.8}$$

Par l'équation (**??**), on a :

$$\int_{Z_i^+} (z^T q_i + \mu_i)\phi_{0,\Lambda_r}(z)dz = c_i \phi_{0,1}\left(\frac{\mu_i}{c_i}\right) + \mu_i \Phi_{0,1}\left(\frac{\mu_i}{c_i}\right)$$

et :

$$\begin{aligned} \int_{Z_i^-} (z^T q_i + \mu_i)\phi_{0,\Lambda_r}(z)dz &= \int_{Z_i^-} (z^T q_i)\phi_{0,\Lambda_r}(z)dz + \mu_i \int_{Z_i^-} \phi_{0,\Lambda_r}(z)dz \\ &= \int_{-\infty}^{-\frac{\mu_i}{c_i}} c_i b_r \phi_{0,1}(b_r)db_r + \mu_i \int_{-\infty}^{-\frac{\mu_i}{c_i}} \phi_{0,1}(b_r)db_r \\ &= c_i \left[\left[-\phi_{0,1}(x) \right]_{-\infty}^{-\frac{\mu_i}{c_i}} \right] + \mu_i \Phi_{0,1}\left(-\frac{\mu_i}{c_i}\right) \\ &= -c_i \phi_{0,1}(\frac{\mu_i}{c_i}) + \mu_i \left(1 - \Phi_{0,1}\left(\frac{\mu_i}{c_i}\right)\right) \\ &= -\left(c_i \phi_{0,1}\left(\frac{\mu_i}{c_i}\right) + \mu_i \Phi_{0,1}\left(\frac{\mu_i}{c_i}\right)\right) + \mu_i \end{aligned} \tag{G.9}$$

Alors, on obtient :

$$\boxed{\mathbb{E}[Y_i] = (1-\lambda) c_i \phi_{0,1}\left(\frac{\mu_i}{c_i}\right) + (1-\lambda)\mu_i \Phi_{0,1}\left(\frac{\mu_i}{c_i}\right) + \lambda \mu_i} \tag{G.10}$$

G.3.2 Estimation de la variance

$$\begin{aligned}\mathbb{E}[Y_i^2] &= \mathbb{E}_{z\sim p(Z)}\left[f_i(Q_r z+\mu)^2\right] \\ &= \int_{-\infty}^{+\infty} f_i(Q_r z+\mu)^2\phi_{0,\Lambda_r}(z)dz \\ &= \int_{Z_i^+}(z^T q_i+\mu_i)^2\phi_{0,\Lambda_r}(z)dz + \lambda^2\int_{Z_i^-}(z^T q_i+\mu_i)^2\phi_{0,\Lambda_r}(z)dz\end{aligned}$$

Par l'équation (**??**) :

$$\int_{Z_i^+}(z^T q_i+\mu_i)^2\phi_{0,\Lambda_r}(z)dz = (\mu_i^2+c_i^2)\Phi_{0,1}\left(\frac{\mu_i}{c_i}\right)+\mu_i c_i\phi_{0,1}\left(\frac{\mu_i}{c_i}\right)$$

Maintenant :

$$\begin{aligned}\int_{Z_i^-}(z^T q_i+\mu_i)^2\phi_{0,\Lambda_r}(z)dz = \int_{Z_i^-}(z^T q_i)^2\phi_{0,\Lambda_r}(z)dz &+ 2\mu_i\int_{Z_i^-} z^T q_i\phi_{0,\Lambda_r}(z)dz \\ &+ \mu_i^2\int_{Z_i^-}\phi_{0,\Lambda_r}(z)dz\end{aligned}$$

avec :

$$\int_{Z_i^-}\phi_{0,\Lambda_r}(z)dz = 1-\Phi_{0,1}\left(\frac{\mu_i}{c_i}\right) \tag{G.11}$$

$$\int_{Z_i^-} z^T q_i\phi_{0,\Lambda_r}(z)dz = -c_i\phi_{0,1}\left(\frac{\mu_i}{c_i}\right) \tag{G.12}$$

$$\begin{aligned}\int_{Z_i^-}(z^T q_i)^2+ &= c_i^2\int_{-\infty}^{-\frac{\mu_i}{c_i}} b_r^2\phi_{0,1}(b_r)db_r \\ &= c_i^2\left([-x\phi_{0,1}(x)]_{-\infty}^{-\frac{\mu_i}{c_i}}+\int_{-\infty}^{-\frac{\mu_i}{c_i}}\phi_{0,1}(b_r)db_r\right) \\ &= \mu_i c_i\phi_{0,1}\left(\frac{\mu_i}{c_i}\right)+c_i^2\left(1-\Phi_{0,1}\left(\frac{\mu_i}{c_i}\right)\right)\end{aligned} \tag{G.13}$$

Alors :

$$\int_{Z_i^-} (z^T q_i + \mu_i)^2 \phi_{0,\Lambda_r}(z) dz = -\left((\mu_i^2 + c_i^2)\Phi_{0,1}\left(\frac{\mu_i}{c_i}\right) + \mu_i c_i \phi_{0,1}\left(\frac{\mu_i}{c_i}\right)\right) + c_i^2 + \mu_i^2 \tag{G.14}$$

Finallement, on obtient :

$$\boxed{\mathbb{E}[Y_i^2] = (1-\lambda^2)(\mu_i^2 + c_i^2)\Phi_{0,1}\left(\frac{\mu_i}{c_i}\right) + (1-\lambda^2)\mu_i c_i \phi_{0,1}\left(\frac{\mu_i}{c_i}\right) + \lambda^2(c_i^2 + \mu_i^2)} \tag{G.15}$$

G.3.3 Estimation des covariances

Notons f_{ij} telle que :

$$\begin{aligned} f_{ij} \colon \mathbb{R}^n &\to \mathbb{R}_+ \\ x &\mapsto f_i(x) f_j(x) \end{aligned}$$

avec f_i définie en (G.7).

$$\begin{aligned} \mathbb{E}[Y_i Y_j] &= \mathbb{E}_{z\sim p(Z)}\left[f_{ij}(Q_r z + \mu)\right] \\ &= \mathbb{E}_{z\sim p(Z)}\left[f_i(Q_r z + \mu) f_j(Q_r z + \mu)\right] \end{aligned} \tag{G.16}$$

En notant :

$$\begin{aligned} Z_{ij}^{+} &= \left\{ z \in \mathbb{R}^r : z^T q_i + \mu_i \geq 0,\ z^T q_j + \mu_j \geq 0 \right\} \\ Z_{ij}^{-} &= \left\{ z \in \mathbb{R}^r : z^T q_i + \mu_i < 0,\ z^T q_j + \mu_j < 0 \right\} \\ Z_{ij}^{-+} &= \left\{ z \in \mathbb{R}^r : z^T q_i + \mu_i \geq 0,\ z^T q_j + \mu_j < 0 \right\} \\ Z_{ij}^{+-} &= \left\{ z \in \mathbb{R}^r : z^T q_i + \mu_i < 0,\ z^T q_j + \mu_j \geq 0 \right\} \end{aligned}$$

On a :

$$\begin{aligned}
\mathbb{E}[Y_i Y_j] &= \int_{z_{ij}^{+}} f_{ij}(Q_r z+\mu)\phi_{0,\Lambda_r}(z)dz \\
&+ \int_{z_{ij}^{-}} f_{ij}(Q_r z+\mu)\phi_{0,\Lambda_r}(z)dz \\
&+ \int_{z_{ij}^{+-}} f_{ij}(Q_r z+\mu)\phi_{0,\Lambda_r}(z)dz \\
&+ \int_{z_{ij}^{-+}} f_{ij}(Q_r z+\mu)\phi_{0,\Lambda_r}(z)dz \\
&= (+)+(-)+(+-)+(-+) \qquad \text{(G.17)}
\end{aligned}$$

Quart d'espace Z_{ij}^{-} :

$$\begin{aligned}
(-) &= \int_{z_{ij}^{-}} f_{ij}(Q_r z+\mu)\phi_{0,\Lambda_r}(z)dz \\
&= \lambda^2 \int_{z_{ij}^{-}} (z^T q_i+\mu_i)(z^T q_j+\mu_j)\phi_{0,\Lambda_r}(z)dz \\
&= \lambda^2[\int_{z_{ij}^{-}} z^T q_i z^T q_j \phi_{0,\Lambda_r}(z)dz + \mu_j \int_{z_{ij}^{-}} z^T q_i \phi_{0,\Lambda_r}(z)dz + \mu_i \int_{z_{ij}^{-}} z^T q_j \phi_{0,\Lambda_r}(z)dz \\
&+ \mu_i\mu_j \int_{z_{ij}^{-}} \phi_{0,\Lambda_r}(z)dz] \\
&= \lambda^2[(-*)+\mu_j(-**)_i+\mu_i(-**)_j+\mu_i\mu_j(-***)] \qquad \text{(G.18)}
\end{aligned}$$

Avec :

$$
\begin{aligned}
(-**) &= \int_{z_{ij}^-} \phi_{0,\Lambda_r}(z)dz \\
&= \int_{-\infty}^{-\frac{\mu_i}{c_i}} \int_{-\infty}^{-\frac{\mu_j+\beta b_r}{\alpha}} \phi_{0,1}(b_r)\phi_{0,1}(b_{r-1})db_r db_{r-1} \\
&= \int_{-\infty}^{-\frac{\mu_i}{c_i}} \phi_{0,1}(b_r)\Phi_{0,1}(-\frac{\mu_j+\beta b_r}{\alpha})db_r \\
&= \int_{-\infty}^{-\frac{\mu_i}{c_i}} \phi_{01}(b_r)db_r - \int_{-\infty}^{-\frac{\mu_i}{c_i}} \phi_{0,1}(b_r)\Phi_{0,1}(\frac{\mu_j+\beta b_r}{\alpha})db_r \\
&= \Phi_{0,1}\left(-\frac{\mu_i}{c_i}\right) - \Phi_{\begin{bmatrix}0\\0\end{bmatrix},\begin{bmatrix}1 & -\frac{\beta}{c_j}\\ -\frac{\beta}{c_j} & 1\end{bmatrix}}\left(\left[\frac{\mu_j}{c_j}, -\frac{\mu_i}{c_i}\right]^T\right) \\
&= 1-\Phi_i-\Phi_{ij} \qquad \text{(G.19)}
\end{aligned}
$$

et :

$$
\begin{aligned}
(-**)_i &= \int_{z_{ij}^-} z^T q_i \phi_{0,\Lambda_r}(z)dz \\
&= c_i \int_{-\infty}^{-\frac{\mu_i}{c_i}} \int_{-\infty}^{-\frac{\mu_j+\beta b_r}{\alpha}} b_r\phi_{0,1}(b_r)\phi_{0,1}(b_{r-1})db_r db_{r-1} \\
&= c_i \int_{-\infty}^{-\frac{\mu_i}{c_i}} b_r\phi_{0,1}(b_r)\Phi_{0,1}\left(-\frac{\mu_j+\beta b_r}{\alpha}\right)db_r \\
&= c_i[\left[-\phi_{0,1}(b_r)\Phi_{0,1}\left(-\frac{\mu_j+\beta b_r}{\alpha}\right)\right]_{-\infty}^{-\frac{\mu_i}{c_i}} \\
&- \frac{\beta}{\alpha}\int_{-\infty}^{-\frac{\mu_i}{c_i}} \phi_{0,1}(b_r)\phi_{0,1}\left(\frac{\mu_j+\beta b_r}{\alpha}\right)db_r] \\
&= c_i[-\phi_{0,1}\left(\frac{\mu_i}{c_i}\right)\Phi_{0,1}\left(-\frac{\mu_j}{\alpha}+\frac{\beta\mu_i}{\alpha c_i}\right) \\
&- \frac{\beta}{\alpha}\phi_{0,1}\left(\frac{\mu_j}{c_j}\right)\int_{-\infty}^{-\frac{\mu_i}{c_i}} \phi_{0,1}\left(\frac{\mu_j+\beta b_r}{\alpha}\right)db_r] \\
&= -c_i\phi_{0,1}\left(\frac{\mu_i}{c_i}\right)\Phi_{0,1}\left(-\frac{\mu_j}{\alpha}+\frac{\beta\mu_i}{\alpha c_i}\right) - \frac{c_i\beta}{c_j}\phi_{0,1}\left(\frac{\mu_j}{c_j}\right)\Phi_{0,1}\left(\frac{\beta\mu_j}{\alpha c_j}-\frac{c_j\mu_i}{\alpha c_i}\right) \\
&= -c_i\phi_i(1-\Phi_{NS}) - \frac{c_i\beta}{c_j}\phi_j(1-\Phi_S) \qquad \text{(G.20)}
\end{aligned}
$$

et :

$$
\begin{aligned}
(-**)_j &= \int_{z_{ij}^-} z^T q_j \phi_{0,\Lambda_r}(z) dz \\
&= \int_{-\infty}^{-\frac{\mu_i}{c_i}} \int_{-\infty}^{-\frac{\mu_j+\beta b_r}{\alpha}} (\alpha b_{r-1} + \beta b_r)\phi_{0,1}(b_r)\phi_{0,1}(b_{r-1}) db_r db_{r-1} \\
&= \beta \int_{-\infty}^{-\frac{\mu_i}{c_i}} b_r \phi_{0,1}(b_r)\Phi_{0,1}\left(-\frac{\mu_j+\beta b_r}{\alpha}\right) db_r \\
&- \alpha \int_{-\infty}^{-\frac{\mu_i}{c_i}} b_r \phi_{0,1}(b_r)\phi_{0,1}\left(\frac{\mu_j+\beta b_r}{\alpha}\right) db_r \\
&= \beta \left[-\phi_{0,1}(b_r)\Phi_{0,1}\left(-\frac{\mu_j+\beta b_r}{\alpha}\right)\right]_{-\infty}^{-\frac{\mu_i}{c_i}} \\
&- (\frac{\beta^2}{\alpha}+\alpha) \int_{-\infty}^{-\frac{\mu_i}{c_i}} b_r \phi_{0,1}(b_r)\phi_{0,1}\left(\frac{\mu_j+\beta b_r}{\alpha}\right) db_r \\
&= -\beta\phi_{0,1}\left(\frac{\mu_i}{c_i}\right)\Phi_{0,1}\left(-\frac{\mu_j}{\alpha}+\frac{\beta\mu_i}{\alpha c_i}\right) - \frac{c_j^2}{\alpha}\int_{-\infty}^{-\frac{\mu_i}{c_i}} \phi_{0,1}\left(\frac{\mu_j+\beta b_r}{\alpha}\right) db_r \\
&= -\beta\phi_{0,1}\left(\frac{\mu_i}{c_i}\right)\Phi_{0,1}\left(\frac{\beta\mu_i}{\alpha c_i}-\frac{\mu_j}{\alpha}\right) - c_j\phi_{0,1}\left(\frac{\mu_j}{c_j}\right)\Phi_{0,1}\left(\frac{\beta\mu_j}{\alpha c_j}-\frac{c_j\mu_i}{\alpha c_i}\right) \\
&= -\beta\phi_i(1-\Phi_{NS}) - c_j\phi_j(1-\Phi_S) \qquad \text{(G.21)}
\end{aligned}
$$

et :

$$
\begin{aligned}
(-*) &= \int_{z_{ij}^-} z^T q_i z^T q_j \phi_{0,\Lambda_r}(z) dz \\
&= c_i\beta \int_{-\infty}^{-\frac{\mu_i}{c_i}} \int_{-\infty}^{-\frac{\mu_j+\beta b_r}{\alpha}} b_r^2 \phi_{0,1}(b_r)\phi_{0,1}(b_{r-1}) db_r db_{r-1} \\
&+ \alpha c_i \int_{-\infty}^{-\frac{\mu_i}{c_i}} \int_{-\infty}^{-\frac{\mu_j+\beta b_r}{\alpha}} b_{r-1} \phi_{0,1}(b_r)\phi_{0,1}(b_{r-1}) db_r db_{r-1} \\
&= c_i\beta \int_{-\infty}^{-\frac{\mu_i}{c_i}} b_r^2 \phi_{0,1}(b_r)\Phi_{0,1}\left(-\frac{\mu_j+\beta b_r}{\alpha}\right) db_r \\
&- \alpha c_i \int_{-\infty}^{-\frac{\mu_i}{c_i}} b_{r-1} \phi_{0,1}(b_r)\phi_{0,1}\left(\frac{\mu_j+\beta b_r}{\alpha}\right) db_r \qquad \text{(G.22)}
\end{aligned}
$$

Avec :

$$\int_{-\infty}^{-\frac{\mu_i}{c_i}} b_r\phi_{0,1}(b_r)\phi_{0,1}\left(\frac{\mu_j+\beta b_r}{\alpha}\right)db_r = \phi_{0,1}\left(\frac{\mu_j}{c_j}\right)\int_{-\infty}^{-\frac{\mu_i}{c_i}} b_r\phi_{0,1}\left(\frac{c_j}{\alpha}b_r+\frac{\beta\mu_j}{\alpha c_j}\right)db_r$$

$$\begin{aligned}
&= \phi_{0,1}\left(\frac{\mu_j}{c_j}\right)\left[\frac{\alpha^2}{c_j^2}\int_{-\infty}^{-\frac{c_j\mu_i}{\alpha c_i}+\frac{\beta\mu_j}{\alpha c_j}} x\phi_{0,1}(x)dx - \frac{\alpha\beta\mu_j}{c_j^3}\int_{-\infty}^{-\frac{c_j\mu_i}{\alpha c_i}+\frac{\beta\mu_j}{\alpha c_j}}\phi_{0,1}(x)dx\right]\\
&= -\frac{\alpha^2}{c_j^2}\phi_{0,1}\left(\frac{\mu_j}{c_j}\right)\phi_{0,1}\left(\frac{\beta\mu_j}{\alpha c_j}-\frac{c_j\mu_i}{\alpha c_i}\right) - \frac{\alpha\beta\mu_j}{c_j^3}\phi_{0,1}\left(\frac{\mu_j}{c_j}\right)\Phi_{0,1}\left(\frac{\beta\mu_j}{\alpha c_j}-\frac{c_j\mu_i}{\alpha c_i}\right)\\
&= -\frac{\alpha^2}{c_j^2}\phi_j\phi_{NS} - \frac{\alpha\beta\mu_j}{c_j^3}\phi_j(1-\Phi_S) \qquad \text{(G.23)}
\end{aligned}$$

et :

$$\int_{-\infty}^{-\frac{\mu_i}{c_i}} b_r^2\phi_{0,1}(b_r)\Phi_{0,1}\left(-\frac{\mu_j+\beta b_r}{\alpha}\right)db_r = \left[-b_r\phi_{0,1}(b_r)\Phi_{0,1}\left(-\frac{\mu_j+\beta b_r}{\alpha}\right)\right]_{-\infty}^{-\frac{\mu_i}{c_i}}$$

$$\begin{aligned}
&+ \int_{-\infty}^{-\frac{\mu_i}{c_i}} \phi_{0,1}(b_r)\Phi_{0,1}\left(-\frac{\mu_j+\beta b_r}{\alpha}\right)db_r - \frac{\beta}{\alpha}\int_{-\infty}^{-\frac{\mu_i}{c_i}} b_r\phi_{0,1}(b_r)\phi_{0,1}\left(\frac{\mu_j+\beta b_r}{\alpha}\right)db_r\\
&= \frac{\mu_i}{c_i}\phi_{0,1}\left(\frac{\mu_i}{c_i}\right)\Phi_{0,1}\left(-\frac{\mu_j}{\alpha}+\frac{\beta\mu_i}{\alpha c_i}\right) + (-**) + \frac{\beta\alpha}{c_j^2}\phi_{0,1}\left(\frac{\mu_j}{c_j}\right)\phi_{0,1}\left(-\frac{c_j\mu_i}{\alpha c_i}+\frac{\beta\mu_j}{\alpha c_j}\right)\\
&+ \frac{\beta^2\mu_j}{c_j^3}\phi_{0,1}\left(\frac{\mu_j}{c_j}\right)\Phi_{0,1}\left(-\left(\frac{c_j\mu_i}{\alpha c_i}-\frac{\beta\mu_j}{\alpha c_j}\right)\right)\\
&= \frac{\mu_i}{c_i}\phi_i(1-\Phi_{NS}) + (-**) + \frac{\beta\alpha}{c_j^2}\phi_j\phi_{NS} + \frac{\beta^2\mu_j}{c_j^3}\phi_j(1-\Phi_S) \qquad \text{(G.24)}
\end{aligned}$$

Finalement :

$$(-*) = \beta\mu_i\phi_i(1-\Phi_{NS}) + \beta c_i(-**) + \alpha c_i\phi_j\phi_{NS} + \frac{\beta\mu_j c_i}{c_j}\phi_j(1-\Phi_S) \qquad \text{(G.25)}$$

Par conséquent :

$$\begin{aligned}
(-) &= \lambda^2[\beta\mu_i\phi_i(1-\Phi_{NS}) + (\beta c_i+\mu_i\mu_j)(1-\Phi_i-\Phi_{ij}) + \alpha c_i\phi_j\phi_{NS}\\
&+ \frac{\beta\mu_j c_i}{c_j}\phi_j(1-\Phi_S) - \mu_i\beta\phi_i(1-\Phi_{NS}) - \mu_i c_j\phi_j(1-\Phi_S) - \mu_j c_i\phi_i(1-\Phi_{NS})\\
&- \frac{c_i\beta\mu_j}{c_j}\phi_j(1-\Phi_S)]\\
&= \lambda^2[(\beta c_i+\mu_i\mu_j)(1-\Phi_i-\Phi_{ij}) + \alpha c_i\phi_j\phi_{NS} - \mu_j c_i\phi_i(1-\Phi_{NS})\\
&- \mu_i c_j\phi_j(1-\Phi_S)] \qquad \text{(G.26)}
\end{aligned}$$

Quart d'espace Z_{ij}^{+-} :

Par des intégrations par parties, et des changements de variables analogues, on obtient :

$$\begin{aligned}
(+-) &= \lambda[(+-*) + \mu_j(+-**)_i + \mu_i(+-**)_j + \mu_i\mu_j(+-***)] \\
&= \lambda[-\beta\mu_i\phi_i(1-\Phi_{NS}) + \beta c_i(\Phi_i - \Phi_j + \Phi_{ij}) - \alpha c_i\phi_j\phi_{NS} + \frac{\beta c_i\mu_j}{c_j}\phi_j\Phi_S \\
&+ c_i\mu_j\phi_i(1-\Phi_{NS}) - \frac{\beta c_i\mu_j}{c_j}\phi_j\Phi_S \\
&+ \beta\mu_i\phi_i(1-\Phi_{NS}) - c_j\mu_i\phi_j\Phi_S \\
&+ \mu_i\mu_j(\Phi_i - \Phi_j + \Phi_{ij})] \\
&= \lambda[(\beta c_i + \mu_i\mu_j)(\Phi_i - \Phi_j + \Phi_{ij}) - \alpha c_i\phi_j\phi_{NS} - c_j\mu_i\phi_j\Phi_S \\
&+ c_i\mu_j\phi_i(1-\Phi_{NS})] \qquad \text{(G.27)}
\end{aligned}$$

Quart d'espace Z_{ij}^{-+} :

De la même manière, on obtient :

$$\begin{aligned}
(-+) &= \lambda[(-+*) + \mu_j(-+**)_i + \mu_i(-+**)_j + \mu_i\mu_j(-+***)] \\
&= \lambda[\beta\mu_i\phi_i\Phi_{NS} + \beta c_i\Phi_{ij} - \alpha c_i\phi_j\phi_{NS} - \frac{\beta c_i\mu_j}{c_j}\phi_j(1-\Phi_S) \\
&- c_i\mu_j\phi_i\Phi_{NS} + \frac{\beta c_i\mu_j}{c_j}\phi_j(1-\Phi_S) \\
&- \beta\mu_i\phi_i\Phi_{NS} + c_j\mu_i\phi_j(1-\Phi_S) \\
&+ \mu_i\mu_j\Phi_{ij}] \\
&= \lambda[(\beta c_i + \mu_i\mu_j)\Phi_{ij} - \alpha c_i\phi_j\phi_{NS} + c_j\mu_i\phi_j(1-\Phi_S) - c_i\mu_j\phi_i\Phi_{NS}] \qquad \text{(G.28)}
\end{aligned}$$

Résultat final :

Finalement, par l'équation (**??**), on a :

$$(+) = \mathbb{E}[Y_iY_j] = c_i\alpha\phi_j\phi_{NS} + (\mu_i\mu_j + c_i\beta)(\Phi_j - \Phi_{ij}) + \mu_i c_j\phi_j\Phi_S + \mu_j c_i\phi_i\Phi_{NS}$$

Par conséquent :

$$
\begin{aligned}
\mathbb{E}[Y_iY_j] &= (+) + (-) + (+-) + (-+) \\
&= (\beta c_i + \mu_i\mu_j)\left(\Phi_j - \Phi_{ij}\right) + c_i\alpha\phi_j\phi_{NS} + \mu_i c_j\phi_j\Phi_S + \mu_j c_i\phi_i\Phi_{NS} \\
&+ \lambda^2(\beta c_i + \mu_i\mu_j)(1 - \Phi_i - \Phi_{ij}) + \lambda^2\alpha c_i\phi_j\phi_{NS} - \lambda^2\mu_j c_i\phi_i(1 - \Phi_{NS}) \\
&- \lambda^2\mu_i c_j\phi_j(1 - \Phi_S) \\
&+ \lambda(\beta c_i + \mu_i\mu_j)(\Phi_i - \Phi_j + \Phi_{ij}) - \lambda\alpha c_i\phi_j\phi_{NS} - \lambda c_j\mu_i\phi_j\Phi_S + \lambda c_i\mu_j\phi_i(1 - \Phi_{NS}) \\
&+ \lambda(\beta c_i + \mu_i\mu_j)\Phi_{ij} - \lambda\alpha c_i\phi_j\phi_{NS} + \lambda c_j\mu_i\phi_j(1 - \Phi_S) - \lambda c_i\mu_j\phi_i\Phi_{NS} \qquad \text{(G.29)}
\end{aligned}
$$

Finalement, on obtient :

$$
\begin{aligned}
\mathbb{E}[Y_iY_j] &= (1-\lambda)^2 c_i\alpha\phi_j\phi_{NS} \\
&+ (\mu_i\mu_j + c_i\beta)\left((1-\lambda)\Phi_j - (1-\lambda)^2\Phi_{ij} + \lambda(1-\lambda)\Phi_i + \lambda^2\right) \\
&+ (1-\lambda)^2\mu_i c_j\phi_j\Phi_S \\
&+ (1-\lambda)^2\mu_j c_i\phi_i\Phi_{NS} \\
&+ \lambda(1-\lambda)(\mu_j c_i\phi_i + \mu_i c_j\phi_j) \qquad \text{(G.30)}
\end{aligned}
$$

G.4 Preuve pour la fonction d'activation ELU

G.4.1 Estimation de la moyenne

Notons f_i telle que :

$$
\begin{aligned}
f_i &: \mathbb{R}^n \to \mathbb{R}_+ \\
x &\mapsto \begin{cases} x_i \text{ if } x_i \geq 0 \\ \lambda\left(e^{x_i} - 1\right) \text{ sinon, } \lambda \in [0,1] \end{cases}
\end{aligned}
$$

$$
\begin{aligned}
\mathbb{E}[Y_i] &= \mathbb{E}_{z\sim p(Z)}\left[f_i(Q_r z + \mu)\right] \\
&= \int_{-\infty}^{+\infty} f_i(Q_r z + \mu)\phi_{0,\Lambda_r}(z)dz \qquad \text{(G.31)}
\end{aligned}
$$

Notons :

$$
Z_i^+ = \left\{z \in \mathbb{R}^r : z^T q_i + \mu_i \geq 0\right\},
$$

et :

$$Z_i^- = \left\{ z \in \mathbb{R}^r : z^T q_i + \mu_i < 0 \right\},$$

Alors :

$$\begin{aligned} \mathbb{E}[Y_i] &= \int_{Z_i^+} f_i(Q_r z + \mu)\phi_{0,\Lambda_r}(z)dz + \int_{Z_i^-} f_i(Q_r z + \mu)\phi_{0,\Lambda_r}(z)dz \\ &= \int_{Z_i^+} (z^T q_i + \mu_i)\phi_{0,\Lambda_r}(z)dz + \lambda \int_{Z_i^-} (e^{z^T q_i + \mu_i} - 1)\phi_{0,\Lambda_r}(z)dz \end{aligned} \tag{G.32}$$

Par l'équation (**??**), on a :

$$\int_{Z_i^+} (z^T q_i + \mu_i)\phi_{0,\Lambda_r}(z)dz = c_i \phi_{0,1}\left(\frac{\mu_i}{c_i}\right) + \mu_i \Phi_{0,1}\left(\frac{\mu_i}{c_i}\right)$$

et :

$$\begin{aligned} \int_{Z_i^-} (e^{z^T q_i + \mu_i} - 1)\phi_{0,\Lambda_r}(z)dz &= e^{\mu_i} \int_{Z_i^-} e^{z^T q_i}\phi_{0,\Lambda_r}(z)dz - \int_{Z_i^-} \phi_{0,\Lambda_r}(z)dz \\ &= e^{\mu_i} \int_{-\infty}^{-\frac{\mu_i}{c_i}} e^{c_i b_r}\phi_{0,1}(b_r)db_r - \int_{-\infty}^{-\frac{\mu_i}{c_i}} \phi_{0,1}(b_r)db_r \\ &= e^{\mu_i + \frac{c_i^2}{2}} \int_{-\infty}^{-\frac{\mu_i}{c_i}} \phi_{0,1}(b_r - c_i)db_r - (1 - \Phi_{0,1}\left(\frac{\mu_i}{c_i}\right)) \\ &= e^{\mu_i + \frac{c_i^2}{2}}(1 - \Phi_{0,1}\left(\frac{\mu_i}{c_i} + c_i\right)) - 1 + \Phi_{0,1}\left(\frac{\mu_i}{c_i}\right) \end{aligned} \tag{G.33}$$

On obient alors :

$$\begin{aligned} \mathbb{E}[Y_i] &= c_i \phi_{0,1}\left(\frac{\mu_i}{c_i}\right) + \mu_i \Phi_{0,1}\left(\frac{\mu_i}{c_i}\right) + \lambda e^{\mu_i + \frac{c_i^2}{2}}(1 - \Phi_{0,1}\left(\frac{\mu_i}{c_i} + c_i\right)) \\ &- \lambda(1 - \Phi_{0,1}\left(\frac{\mu_i}{c_i}\right)) \end{aligned} \tag{G.34}$$

En notant : $\phi_i = \phi_{0,1}\left(\frac{\mu_i}{c_i}\right)$, $\Phi_i = \Phi_{0,1}\left(\frac{\mu_i}{c_i}\right)$ et $\Phi_{c_i} = \Phi_{0,1}\left(\frac{\mu_i}{c_i} + c_i\right)$ on a :

$$\boxed{\mathbb{E}[Y_i] = c_i \phi_i + (\mu_i + \lambda)\Phi_i + \lambda e^{\mu_i + \frac{c_i^2}{2}}(1 - \Phi_{c_i}) - \lambda} \tag{G.35}$$

G.4.2 Estimation de la variance

$$\begin{aligned}
\mathbb{E}[Y_i^2] &= \mathbb{E}_{z\sim p(Z)}\left[f_i(Q_r z+\mu)^2\right] \\
&= \int_{-\infty}^{+\infty} f_i(Q_r z+\mu)^2 \phi_{0,\Lambda_r}(z)dz \\
&= \int_{Z_i^+} (z^T q_i+\mu_i)^2 \phi_{0,\Lambda_r}(z)dz + \lambda^2 \int_{Z_i^-} (e^{z^T q_i+\mu_i}-1)^2 \phi_{0,\Lambda_r}(z)dz
\end{aligned}$$

Par l'équation (**??**) :

$$\int_{Z_i^+} (z^T q_i+\mu_i)^2 \phi_{0,\Lambda_r}(z)dz = (\mu_i^2+c_i^2)\Phi_{0,1}\left(\frac{\mu_i}{c_i}\right) + \mu_i c_i \phi_{0,1}\left(\frac{\mu_i}{c_i}\right)$$

Maintenant :

$$\begin{aligned}
\int_{Z_i^-} (e^{z^T q_i+\mu_i}-1)^2 \phi_{0,\Lambda_r}(z)dz &= e^{2\mu_i}\int_{Z_i^-} e^{2z^T q_i}\phi_{0,\Lambda_r}(z)dz \\
&- 2e^{\mu_i}\int_{Z_i^-} e^{z^T q_i}\phi_{0,\Lambda_r}(z)dz + \int_{Z_i^-}\phi_{0,\Lambda_r}(z)dz \qquad \text{(G.36)}
\end{aligned}$$

avec :

$$\int_{Z_i^-} \phi_{0,\Lambda_r}(z)dz = 1-\Phi_{0,1}\left(\frac{\mu_i}{c_i}\right) \qquad \text{(G.37)}$$

$$\begin{aligned}
\int_{Z_i^-} e^{z^T q_i}\phi_{0,\Lambda_r}(z)dz &= \int_{-\infty}^{-\frac{\mu_i}{c_i}} e^{c_i b_r}\phi_{0,1}(b_r)db_r \\
&= e^{\frac{c_i^2}{2}}\int_{-\infty}^{-\frac{\mu_i}{c_i}} \phi_{0,1}(b_r-c_i)db_r \\
&= e^{\frac{c_i^2}{2}}(1-\Phi_{0,1}\left(\frac{\mu_i}{c_i}+c_i\right)) \qquad \text{(G.38)}
\end{aligned}$$

$$\begin{aligned}
\int_{Z_i^-} e^{2z^T q_i}\phi_{0,\Lambda_r}(z)dz &= e^{2c_i^2}\int_{-\infty}^{-\frac{\mu_i}{c_i}} \phi_{0,1}(b_r-2c_i)db_r \\
&= e^{2c_i^2}(1-\Phi_{0,1}\left(\frac{\mu_i}{c_i}+2c_i\right)) \qquad \text{(G.39)}
\end{aligned}$$

Alors :

$$\begin{aligned}\int_{Z_i^-}(e^{z^T q_i+\mu_i}-1)^2\phi_{0,\Lambda_r}(z)dz &= e^{2(c_i^2+\mu_i)}(1-\Phi_{0,1}\left(\frac{\mu_i}{c_i}+2c_i\right))\\ &- 2e^{\mu_i+\frac{c_i^2}{2}}(1-\Phi_{0,1}\left(\frac{\mu_i}{c_i}+c_i\right))+1-\Phi_{0,1}\left(\frac{\mu_i}{c_i}\right)\end{aligned}$$

En notans : $\phi_i=\phi_{0,1}\left(\frac{\mu_i}{c_i}\right)$, $\Phi_i=\Phi_{0,1}\left(\frac{\mu_i}{c_i}\right)$, $\Phi_{c_i}=\Phi_{0,1}\left(\frac{\mu_i}{c_i}+c_i\right)$ et $\Phi_{2c_i}=\Phi_{0,1}\left(\frac{\mu_i}{c_i}+2c_i\right)$ we have :

$$\begin{aligned}\mathbb{E}[Y_i^2] &= (\mu_i^2+c_i^2)\Phi_i+\mu_i c_i\phi_i+\lambda^2 e^{2(c_i^2+\mu_i)}(1-\Phi_{2c_i})-2\lambda^2 e^{\mu_i+\frac{c_i^2}{2}}(1-\Phi_{c_i})\\ &+ \lambda^2(1-\Phi_i)\end{aligned} \tag{G.40}$$

On obtient finalement :

$$\boxed{\mathbb{E}[Y_i^2]=(\mu_i^2+c_i^2-\lambda^2)\Phi_i+\mu_i c_i\phi_i+\lambda^2 e^{2(c_i^2+\mu_i)}(1-\Phi_{2c_i})-2\lambda^2 e^{\mu_i+\frac{c_i^2}{2}}(1-\Phi_{c_i})+\lambda^2} \tag{G.41}$$

G.4.3 Covariance value estimation

Notons f_{ij}, telle que :

$$\begin{aligned}f_{ij}\colon \mathbb{R}^n &\to \mathbb{R}_+\\ x &\mapsto f_i(x)f_j(x)\end{aligned}$$

avec f_i définie comme en (G.31).

$$\begin{aligned}\mathbb{E}[Y_iY_j] &= \mathbb{E}_{z\sim p(Z)}\left[f_{ij}(Q_r z+\mu)\right]\\ &= \mathbb{E}_{z\sim p(Z)}\left[f_i(Q_r z+\mu)f_j(Q_r z+\mu)\right]\end{aligned} \tag{G.42}$$

En notant :

$$\begin{aligned}Z_{ij}^{+} &= \left\{z\in\mathbb{R}^r : z^Tq_i+\mu_i\geq 0,\ z^Tq_j+\mu_j\geq 0\right\}\\ Z_{ij}^{-} &= \left\{z\in\mathbb{R}^r : z^Tq_i+\mu_i< 0,\ z^Tq_j+\mu_j< 0\right\}\\ Z_{ij}^{-+} &= \left\{z\in\mathbb{R}^r : z^Tq_i+\mu_i\geq 0,\ z^Tq_j+\mu_j< 0\right\}\\ Z_{ij}^{+-} &= \left\{z\in\mathbb{R}^r : z^Tq_i+\mu_i< 0,\ z^Tq_j+\mu_j\geq 0\right\}\end{aligned}$$

On a :

$$\begin{aligned}
\mathbb{E}[Y_iY_j] &= \int_{z_{ij}^+} f_{ij}(Q_r z+\mu)\phi_{0,\Lambda_r}(z)dz \\
&+ \int_{z_{ij}^-} f_{ij}(Q_r z+\mu)\phi_{0,\Lambda_r}(z)dz \\
&+ \int_{z_{ij}^{+-}} f_{ij}(Q_r z+\mu)\phi_{0,\Lambda_r}(z)dz \\
&+ \int_{z_{ij}^{-+}} f_{ij}(Q_r z+\mu)\phi_{0,\Lambda_r}(z)dz \qquad \text{(G.43)} \\
&= (+)+(-)+(+-)+(-+) \qquad \text{(G.44)}
\end{aligned}$$

Quart d'espace Z_{ij}^- :

Par des changements de variables classiques et intégrations par parties, on obtient :

$$\begin{aligned}
(-) &= \lambda^2[e^{\mu_i+\mu_j}(-*) - e^{\mu_i}(-**)_i - e^{\mu_j}(-**)_j + (-***)] \\
&= \lambda^2[e^{\mu_i+\mu_j+\frac{c_j^2+c_i^2}{2}+c_i\beta}(1-\Phi_{\beta+c_i}-\Phi_C) \\
&- e^{\mu_i+\frac{c_i^2}{2}}(1-\Phi_{c_i}-\Phi_A) - e^{\mu_j+\frac{c_j^2}{2}}(1-\Phi_{\beta_i}-\Phi_B) + 1 - \Phi_i - \Phi_{ij}] \qquad \text{(G.45)}
\end{aligned}$$

où $\Phi_A = \Phi_{\begin{bmatrix}0\\0\end{bmatrix},\begin{bmatrix}1 & -\frac{\beta}{c_j}\\ -\frac{\beta}{c_j} & 1\end{bmatrix}}\left(\left[\frac{\mu_j+\beta c_i}{c_j}, -\frac{\mu_i}{c_i}-c_i\right]^T\right)$,

$\Phi_B = \Phi_{\begin{bmatrix}0\\0\end{bmatrix},\begin{bmatrix}1 & -\frac{\beta}{c_j}\\ -\frac{\beta}{c_j} & 1\end{bmatrix}}\left(\left[\frac{\mu_j}{c_j}+c_j, -\frac{\mu_i}{c_i}-\beta\right]^T\right)$,

$\Phi_C = \Phi_{\begin{bmatrix}0\\0\end{bmatrix},\begin{bmatrix}1 & -\frac{\beta}{c_j}\\ -\frac{\beta}{c_j} & 1\end{bmatrix}}\left(\left[\frac{\mu_j+c_j^2+\beta c_i}{c_j}, -(\frac{\mu_i}{c_i}+c_i+\beta)\right]^T\right)$),

et $\Phi_{\beta+c_i} = \Phi_{0,1}(\frac{\mu_i}{c_i}+c_i+\beta)$, $\Phi_{\beta_i} = \Phi_{0,1}\left(\frac{\mu_i}{c_i}+\beta\right)$ et $\Phi_{c_i} = \Phi_{0,1}\left(\frac{\mu_i}{c_i}+c_i\right)$

Quart d'espace Z_{ij}^{+-} :

De manière analogue, on obtient :

$$\begin{aligned}
(+-) &= \lambda[e^{\mu_j}(+-*) - (+-**)_i + \mu_i e^{\mu_j}(+-**)_j - \mu_i(+-***)] \\
&= \lambda[c_i e^{\mu_j+\frac{c_j^2}{2}}(\phi_{\beta_i}\Phi_\alpha - \frac{\beta}{c_j}\phi_{c_j}\Phi_S + \beta(\Phi_{\beta_i} - \Phi_{c_j} + \Phi_B)) - c_i\phi_i(1-\Phi_{NS}) \\
&+ \frac{\beta c_i}{c_j}\phi_j\Phi_S + \mu_i e^{\mu_j+\frac{c_j^2}{2}}(\Phi_{\beta_i} - \Phi_{c_j} + \Phi_B) - \mu_i(\Phi_i - \Phi_j + \Phi_{ij})] \\
&= \lambda[e^{\mu_j+\frac{c_j^2}{2}}(\mu_i + \beta c_i)(\Phi_{\beta_i} - \Phi_{c_j} + \Phi_B) + \frac{\beta c_i}{c_j}(\phi_j - e^{\mu_j+\frac{c_j^2}{2}}\phi_{c_j})\Phi_S \\
&+ c_i e^{\mu_j+\frac{c_j^2}{2}}\phi_{\beta_i}\Phi_\alpha - c_i\phi_i(1-\Phi_{NS}) - \mu_i(\Phi_i - \Phi_j + \Phi_{ij})]
\end{aligned} \tag{G.46}$$

Quart d'espace Z_{ij}^{-+} :

De manière analogue, on obtient :

$$\begin{aligned}
(-+) &= \lambda[e^{\mu_i}(-+*) - (-+**)_j + \mu_j e^{\mu_i}(-+**)_i - \mu_j(-+***)] \\
&= \lambda[e^{\mu_i+\frac{c_i^2}{2}}(c_j\phi_a\Phi_e - \beta\phi_{c_i}\Phi_{NS} + \beta c_i\Phi_A) + \beta\phi_i\Phi_{NS} - c_j\phi_j(1-\Phi_S) \\
&+ \mu_j e^{\mu_i+\frac{c_i^2}{2}}\Phi_A - \mu_j\Phi_{ij}] \\
&= \lambda[e^{\mu_i+\frac{c_i^2}{2}}(\beta c_i + \mu_j)\Phi_A + \beta(\phi_i - e^{\mu_i+\frac{c_i^2}{2}}\phi_{c_i})\Phi_{NS} + c_j e^{\mu_i+\frac{c_i^2}{2}}\phi_a\Phi_e \\
&- c_j\phi_j(1-\Phi_S) - \mu_j\Phi_{ij}]
\end{aligned} \tag{G.47}$$

Résulat final :

Par l'équation (**??**), on a :

$$(+) = \mathbb{E}[Y_iY_j] = c_i\alpha\phi_j\phi_{NS} + (\mu_i\mu_j + c_i\beta)(\Phi_j - \Phi_{ij}) + \mu_i c_j\phi_j\Phi_S + \mu_j c_i\phi_i\Phi_{NS}$$

Avec :

$$\mathbb{E}[Y_iY_j] = (+) + (-) + (+-) + (-+) \tag{G.48}$$

On obtient finalement :

$$\begin{aligned}
\mathbb{E}[Y_iY_j] &= c_i\alpha\phi_j\phi_{NS} + (\mu_i\mu_j + c_i\beta)\left(\Phi_j - \Phi_{ij}\right) + \mu_i c_j\phi_j\Phi_S + \mu_j c_i\phi_i\Phi_{NS} \\
&+ \lambda^2[e^{\mu_i+\mu_j+\frac{c_j^2+c_i^2}{2}+c_i\beta}(1-\Phi_{\beta+c_i}-\Phi_C) \\
&- e^{\mu_i+\frac{c_i^2}{2}}(1-\Phi_{c_i}-\Phi_A) - e^{\mu_j+\frac{c_j^2}{2}}(1-\Phi_{\beta_i}-\Phi_B) + 1 - \Phi_i - \Phi_{ij}] \\
&+ \lambda[e^{\mu_j+\frac{c_j^2}{2}}(\mu_i+\beta c_i)(\Phi_{\beta_i}-\Phi_{c_j}+\Phi_B) + \frac{\beta c_i}{c_j}(\phi_j - e^{\mu_j+\frac{c_j^2}{2}}\phi_{c_j})\Phi_S \\
&+ c_i e^{\mu_j+\frac{c_j^2}{2}}\phi_{\beta_i}\Phi_\alpha - c_i\phi_i(1-\Phi_{NS}) - \mu_i(\Phi_i - \Phi_j + \Phi_{ij})] \\
&+ \lambda[e^{\mu_i+\frac{c_i^2}{2}}(\beta c_i+\mu_j)\Phi_A + \beta(\phi_i - e^{\mu_i+\frac{c_i^2}{2}}\phi_{c_i})\Phi_{NS} + c_j e^{\mu_i+\frac{c_i^2}{2}}\phi_a\Phi_e \\
&- c_j\phi_j(1-\Phi_S) - \mu_j\Phi_{ij}] \qquad \text{(G.49)}
\end{aligned}$$

Par conséquent :

$$\begin{aligned}
\mathbb{E}[Y_iY_j] &= c_i\alpha\phi_j\phi_{NS} \\
&+ \left(\mu_i\mu_j + c_i\beta + \lambda\mu_i\right)\Phi_j \\
&- \left(\mu_i\mu_j + c_i\beta + \lambda(\mu_i+\mu_j) + \lambda^2\right)\Phi_{ij} \\
&+ (\mu_i c_j\phi_j + \lambda c_j\phi_j + \frac{\lambda\beta c_i}{c_j}((\phi_j - e^{\mu_j+\frac{c_j^2}{2}}\phi_{c_j}))\Phi_S \\
&+ (\mu_j c_i\phi_i + \lambda c_i\phi_i + \lambda\beta(\phi_i - e^{\mu_i+\frac{c_i^2}{2}}\phi_{c_i}))\Phi_{NS} \\
&+ \lambda c_i e^{\mu_j+\frac{c_j^2}{2}}\phi_{\beta_i}\Phi_\alpha \\
&+ \lambda c_j e^{\mu_i+\frac{c_i^2}{2}}\phi_a\Phi_e \\
&+ \lambda^2 e^{\mu_i+\frac{c_i^2}{2}}\Phi_{c_i} \\
&- \lambda e^{\mu_j+\frac{c_j^2}{2}}(\mu_i+\beta c_i)\Phi_{c_j} \\
&+ \lambda e^{\mu_i+\frac{c_i^2}{2}}(\lambda+\beta c_i+\mu_j)\Phi_A \\
&+ \lambda e^{\mu_j+\frac{c_j^2}{2}}(\lambda+\beta c_i+\mu_i)(\Phi_{\beta_i}+\Phi_B) \\
&+ \lambda^2 e^{\mu_i+\mu_j+\frac{c_i^2+c_j^2}{2}+c_i\beta}(1-\Phi_{\beta+c_i}-\Phi_C) \\
&- \lambda(\lambda e^{\mu_i+\frac{c_i^2}{2}} + \lambda e^{\mu_j+\frac{c_j^2}{2}} + c_i\phi_i + c_j\phi_j - \lambda) \qquad \text{(G.50)}
\end{aligned}$$

Annexe H

Rebasement et caractérisation du simplexe résultant

La conjecture 3.2.1 n'est, en général, pas vérifiée. Cependant, la figure 4 de (Ainsworth et al., 2022) suggère qu'il est possible de rebaser des réseaux lorsque le nombre de paramètres par couche d'un réseau augmente. Ainsi, on observe que pour une architecture VGG16, la valeur de la barrière devient faible pour un facteur 4 sur le nombre de filtres des couches du réseau. Il en est de même pour un Resnet20 ou une barrière nulle est atteinte pour un facteur multiplicateur de 32.

La figure H.1 présente l'évolution de la valeur de la barrière en fonction du nombre de neurones par couche, dans le cas d'un perceptron muni de 4 couches connectées, chacune munie d'une couche de batch normalisation et d'une ReLU.

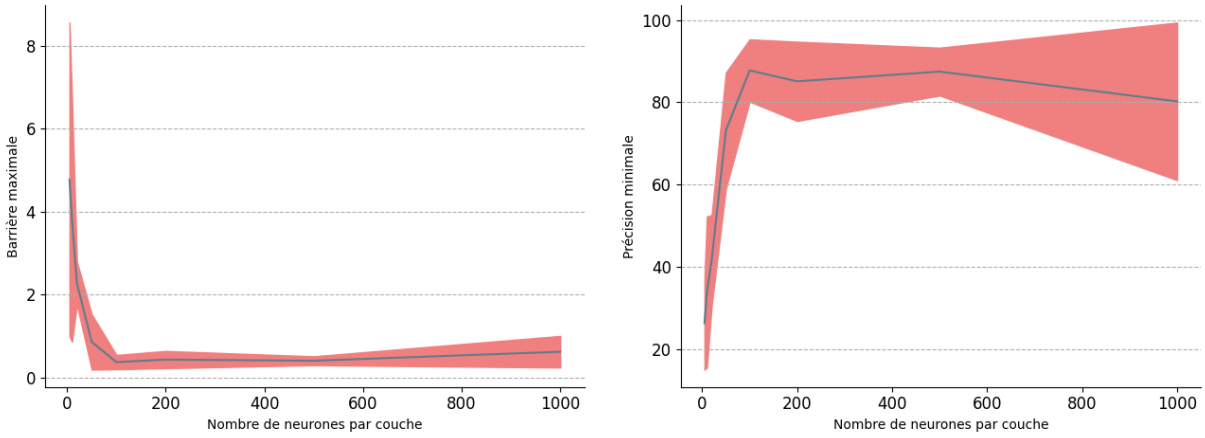


FIGURE H.1 – Evolution de la valeur de barrière (**Gauche**) et de la précision (**Droite**) en fonction du nombre de neurones par couches.

On observe, similairement à Ainsworth et al. (2022), que lorsque le nombre de neurones par couche augmente, la valeur de la barrière entre deux réseaux rebasés diminue, sans pour autant

atteindre une valeur nulle.

Remarque 21.

Nous n'avons malheureusement pas pu reproduire ces résultats sur les architectures VGG16 et ResNet50 munies du jeu de données CIFAR10, comme présenté par Ainsworth et al. (2022).

Une première approche consiste alors à entraîner N réseau de neurones munis d'un nombre de neurones permettant de vérifier la conjecture 3.2.1 (ici 4 couches de 1000 neurones), de les rebaser au sein d'un même bassin locale et d'analyser le simplexe résultant.

Plus formellement notons $(\hat{\theta}_i)_{1\leq i\leq N}$ l'ensemble des paramètres des réseaux (rebasés) obtenues après N entraînement successifs. On note $\mathcal{S}_n = \left\{\lambda_0\hat{\theta}_0 + \cdots + \lambda_n\hat{\theta}_n \mid \sum_{i=0}^{n}\lambda_i = 1 \text{ et } \lambda_i \geq 0 \text{ pour } i = 0, \ldots, n\right\}$ le simplexe dont les sommets sont les $(\hat{\theta}_i)$.

Échantillonnage de $\mathcal{S}_n$ par hypercube latin Une première approche consiste à échantillonner aléatoirement au sein de ce simplexe. La FigureH.2 présente la distribution des coûts des réseaux échantillonner à l'aide d'un hypercube latin au sein des simplexe $\mathcal{S}_n$, et pour différentes valeurs de n.

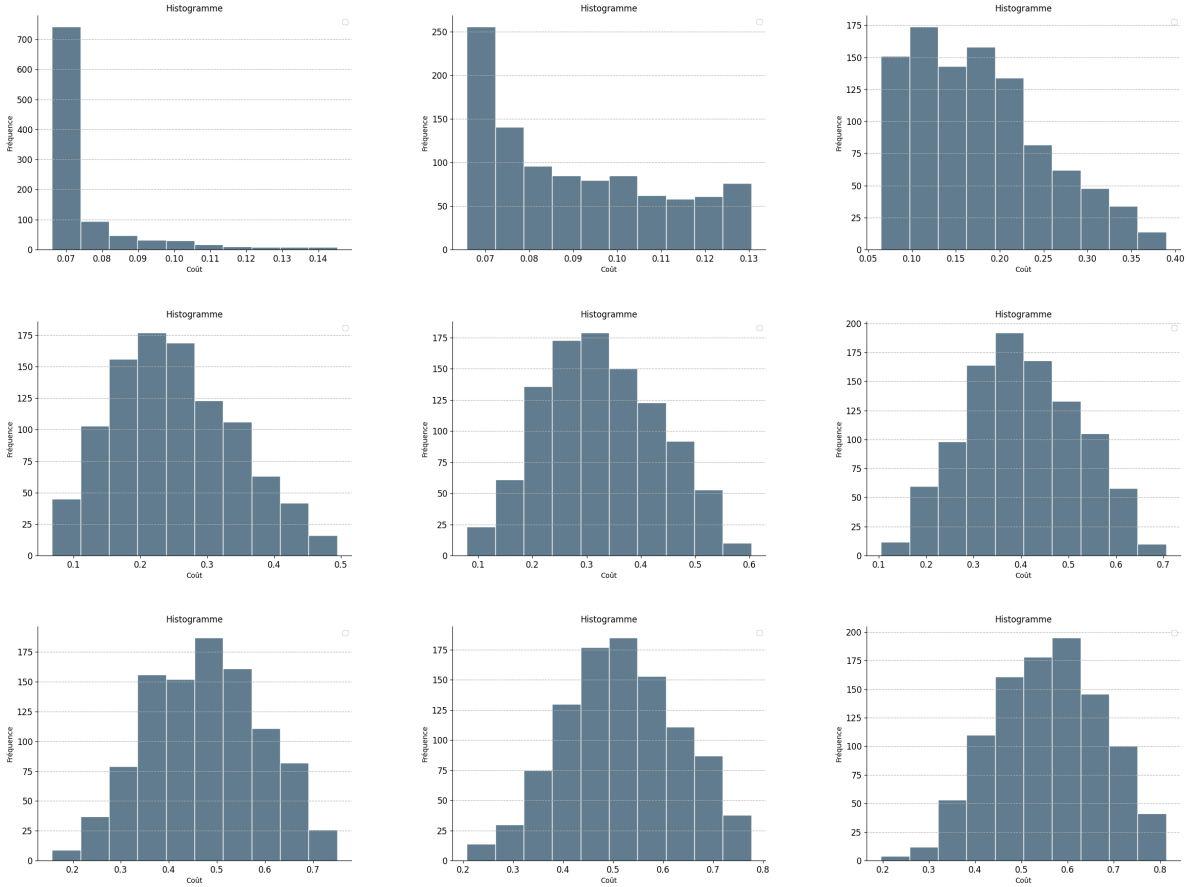


FIGURE H.2 – Histogrammes des valeurs de coût des réseaux échantillonnés par LHS au sein du simplexe $\mathcal{S}_n$. De **Gauche** à **Droite** et de **Haut** en **Bas** : $n = 2, 3, 4, 5, 6, 7, 8, 9, 10$.

Très rapidement (pour $n > 4$) il devient très difficile de trouver des valeurs de λ représentant des réseaux de neurones à faible valeur de coût. Le volume des "bons" réseaux semble très petit et donc difficilement échantillonable.

Annexe I

Pourquoi les ensembles profonds marchent-ils si bien ?

Dans la littérature, plusieurs approches (Welling and Teh, 2011) (Maddox et al., 2019) ont pu être améliorées en réalisant un "restart" (froid ou chaud) de l'entraînement (Seedat and Kanan, 2019)(Wilson and Izmailov, 2020). Ce "restart" permet d'explorer différents modes du paysage des paramètres et plus simplement un unique bassin d'attraction local situé autour du mode courant (Loshchilov and Hutter, 2016).

Afin de mieux comprendre ce phénomène, nous nous intéressons, ici, au concept de diversité d'un ensemble de prédicteurs. Nous nous plaçons, dans cette partie, dans un cadre de classification, où le prédicteur est muni d'une fonction softmax comme couche de sortie.

Intuitivement, un ensemble présente une diversité élevée lorsque ses constituants ne présentent pas des prédictions similaires pour les mêmes entrées. Afin que la performance moyenne d'un ensemble soit supérieure à celle de ses composants, les éléments constitutifs ne doivent pas effectuer leurs erreurs sur les mêmes entrées. Pour illustrer ce point, la table I.1 présente les prédictions de deux hypothétiques prédicteurs sur une entrée dont la classe à prédire est la classe 0.

TABLE I.1 – Exemple de deux prédicteurs se trompant sur une entrée quand leur ensemble prédit la bonne classe 0.

Prédicteurs	Classe 0	Classe 1	Classe 2	Classe 3
0	0.38	0.18	0.01	0.43
1	0.30	0.45	0.09	0.16
0/1	0.34	0.315	0.05	0.295

Ici, les deux prédicteurs se trompent individuellement (sur deux classes distinctes), alors que l'en-

semble constitué de ces deux prédicteurs (uniformément pondéré) prédit la bonne classe. Cependant, il est nécessaire de garder à l'esprit que la diversité d'un ensemble ne peut pas à elle-seule garantir une bonne performance globale.

Dans la suite, nous adoptons en plus des notations générales les notations suivantes. On note $\mathbf{y}_i$ la prédiction de la i^e entrée de test. Dans un cas de classification, la dimension de sortie d' est égale au nombre de classes. On note ainsi $\mathbf{y}_{ij}$ le score prédit par le réseau de la j^e classe de l'entrée i. De plus, on note $\mathbf{y}_{i\infty} = \text{argmax}_j\, \mathbf{y}_{ij}$ la classe prédite par le réseau. $\bar{\mathbf{y}}_i$ dénote la prédiction d'un ensemble sur l'entrée i. Nous reprenons la notation $\mathcal{D}_N = (\mathbf{x}_i, \mathbf{y}_i)_{1 \leq i \leq N}$ pour désigner notre jeu de données de test. Enfin, nous notons $\mathbb{D}$ une mesure de diversité sur un ensemble de prédicteurs noté communément E.

Le développement de métriques permettant de quantifier ce phénomène peut s'avérer utile pour 1) mieux comprendre la performance des ensembles de prédicteurs, 2) améliorer la constitution d'ensembles de prédicteurs et 3) être inséré comme outil au sein d'algorithmes. Après un court état de l'art sur les notions de diversité d'ensembles de prédicteurs, nous utilisons ces métriques afin d'étudier la diversité des différents ensembles construits pendant notre étude comparative.

De nombreux papiers sont présents dans la littérature proposant différentes métriques de quantification de la diversité d'un ensemble de prédicteurs. Le concept de diversité est un pan de recherche (complexe) en lui-même. Nous fournissons ici les métriques de quantification de diversité des deux plus récentes publications de l'état de l'art, sans pour autant nous attarder sur la théorie dont découlent ces métriques. On cite notamment les travaux de thèse de Brown (2004) qui étudie en profondeur ces concepts.

Ainsi Ortega et al. (2022) propose de quantifier la diversité d'un ensemble E par :

$$\mathbb{D}(E) = \mathbb{E}_{\mathcal{D}_N}\left[\frac{p(\mathbf{y} \mid \boldsymbol{x}, \boldsymbol{\theta})^2 - p(\mathbf{y} \mid \boldsymbol{x}, \boldsymbol{\theta}) p\left(\mathbf{y} \mid \boldsymbol{x}, \boldsymbol{\theta}'\right)}{2 \max_{\boldsymbol{\theta} \in \boldsymbol{\Theta}} p(\mathbf{y} \mid \boldsymbol{x}, \boldsymbol{\theta})^2}\right] \tag{I.1}$$

Wood et al. (2023) proposent alternativement la quantité suivante :

$$\mathbb{D}(E) = \frac{1}{N} \sum_{i=1}^{N} KL\left(\bar{\mathbf{y}}_i \| \mathbf{y}_i\right) \tag{I.2}$$

En mimant la métrique de diversité précédente, on intègre également des variantes en remplaçant la divergence de Kullback-Lieber par les distances L_1, L_2, L_∞, Jensen et la distance de Bhattacharyya.

Le tableau J.1 présente les résultats de ces métriques sur les méthodes SGHMC, cSGHMC, SGLD,

cSGLD, SWAG et Multi-SWAG sur nos deux cas expérimentaux.

TABLE I.2 – Comparaison des scores de diversité et la précision d'un ensemble en fonction du nombre de ses constituants.

		CRITÈRE DE PERFORMANCE					
Tâche	**Distance**	**SGHMC**	**cSGHMC**	**SGLD**	**cSGLD**	**SWAG**	**Multi-SWAG**
MNIST	L_1	0.003267	0.009696	0.003991	0.007933	0.000039	0.008489
	L_2	0.002044	0.006055	0.002582	0.005095	0.000024	0.005362
	L_∞	0.001597	0.004680	0.001952	0.003861	0.000019	0.004119
	Wood	0.000284	0.003644	0.000386	0.001922	0.000001	0.002948
	Ortega	0.000012	0.000082	0.000015	0.000055	0.000000	0.000079
	Bhattacharyya	0.000094	0.000827	0.000112	0.000467	0.000000	0.000679
	Jensen	0.000003	0.000678	0.000048	0.000386	0.000000	0.000543
CIFAR-10	L_1	0.011582	0.057476	0.012264	0.036259	0.000099	0.140531
	L_2	0.007696	0.037836	0.008173	0.023659	0.000062	0.089098
	L_∞	0.005717	0.028237	0.006052	0.017770	0.000048	0.068058
	Wood	0.001440	0.048574	0.001527	0.014893	0.000000	0.166621
	Ortega	0.000057	0.000846	0.000061	0.000379	0.000000	0.002478
	Bhattacharyya	0.000368	0.008194	0.000388	0.003142	0.000000	0.027080
	Jensen	0.000313	0.006730	0.000338	0.002823	0.000000	0.021512

Pour l'ensemble des mesures de diversité, les méthodes avec "restart" présente de plus hauts scores de diversité que leur contrepartie simple entraînement. Ces résultats mettent en évidence que l'exploration multimodale du paysage des paramètres permet d'augmenter la diversité de nos prédicteurs en captant diverses représentations de la tâche apprise et définie par la base de données fournie. Les ensembles profonds intègrent donc naturellement une variété de représentation de la tâche apprise et permettent d'atteindre de hauts niveaux de performance.

Annexe J

Corrélation de la précision d'un ensemble avec sa diversité.

L'approche de la diversité sur un ensemble généré par *ensemble profond* sur un VGG16 muni de la base d'entraînement CIFAR-10. Ainsi, pour les distances L_1, L_2, L_∞, Bhattacharyya, Ortega (Ortega et al., 2022), Wood (Wood et al., 2023), Buschjager (Buschjäger et al., 2020) et Zhang (Zhang et al., 2020a), nous visualisons l'évolution de la diversité en fonction du nombre d'éléments présents dans l'ensemble au sein de la figure J.1. Nous intégrons dans cette analyse notre approche munie de la norme L_∞ comme distance élémentaire, et noté "M".

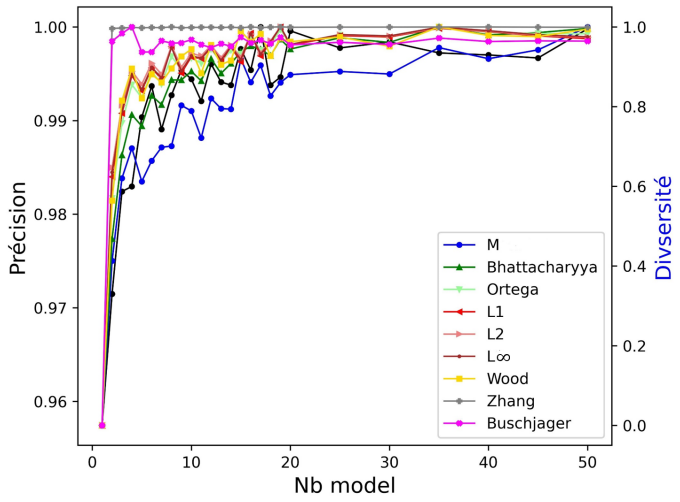


FIGURE J.1 – Évolution de la précision (courbe **noire** liée à l'axe des ordonnées de **gauche**) et de la diversité (courbes **colorées** liées à l'axe des ordonnées de **droite**) d'un ensemble de réseaux VGG16 entraînés sur la tâche CIFAR-10 en fonction de son nombre d'éléments.

Tout d'abord, nous observons que les métriques intitulées *Zhang* et *Buschjager* ne capturent pas la tendance globale de l'évolution de la précision d'un ensemble en fonction du nombre de ses éléments. Ensuite, bien que les autres métriques montrent des tendances cohérentes avec l'évolution

de la précision, il est difficile de conclure.

Afin de montrer la pertinence de notre approche, nous calculons donc la corrélation de Pearson (Freedman et al., 2007) entre les scores de diversité fournies par les différentes métriques et la précision des ensembles en fonction du nombre de ses constituants. Nous comparons alors les résultats obtenus à partir des métriques de l'état de l'art avec notre approche. Les résultats sont présentés au sein du tableau J.1.

TABLE J.1 – Comparaison de la corrélation de Pearson entre les scores de diversité et la précision d'un ensemble en fonction du nombre de ses constituants.

	CRITÈRE DE PERFORMANCE				
Distance	**Moyenne ↑**	**Écart-type ↓**	**Médiane ↑**	**Percentile 2.5 ↑**	**Percentile 97.5 ↑**
L_1	0.9353	0.0550	0.9494	0.8114	0.9831
L_2	0.9344	0.0555	0.9472	0.8101	0.9831
L_∞	0.9363	0.0545	0.9501	0.8136	0.9837
Wood	0.9631	0.0349	0.9774	0.8839	0.9897
Ortega	0.9645	0.0346	0.9762	0.8843	0.9883
Bhattacharyya	0.9673	0.0285	0.9801	0.9026	0.9880
Jensen	0.9639	0.0331	0.9782	0.8878	0.9846
M	0.9709	0.0254	0.9799	0.9130	0.9888

On note que notre approche fournit les meilleurs résultats sur l'ensemble des critères de performances, à l'exception de la médiane, où la distance de Bhattacharyya montre une valeur très légèrement supérieure (0.9801 pour la distance de Bhattacharyya contre 0.9799 pour notre approche) et du *percentile 97.5* où l'approche de Wood et al. (2023) devance notre approche (0.9897 contre 0.9888). Cependant, notre proposition offre les meilleurs résultats en termes de *moyenne* (atteignant une valeur de 0.9709), *écart-type*, ainsi que sur le *percentile 2.5*. Nous relevons que cette métrique de diversité n'a pas été motivée par un travail théorique, mais plutôt d'un travail d'observations sur les prédictions d'un réseau de neurones dans le cadr e d'une tâche de classification. Une analyse plus approfondie doit être poursuivie afin de valider plus fortement notre approche "intuitive".

Remarque 12.

Cette analyse de la diversité d'un ensemble de réseaux de neurones a été complétée par la recherche de familles de prédicteurs, en tentant de les classifier à la vue de leurs prédictions sur un jeu de données de validation. Cette recherche s'est effectuée à l'aide d'algorithmes de classification non-supervisée tel HDSCAN (Campello et al., 2013), K-means (Lloyd, 1982), MeanShift (Fukunaga and Hostetler, 1975), Affiny Propagation (Frey and Dueck, 2007) ou encore AgglomerativeClustering (Gowda and Krishna, 1978). Cette tentative de clustering s'est effectuée sur les prédictions des réseaux mais également sur une représentation bas niveau de ces prédictions en utilisant des algorithmes tels que le t-SNE (van der Maaten and Hinton, 2008) et l'algorithme UMAP (McInnes et al., 1802). Malheureusement, cette recherche s'est montrée infructueuse et nous a conduit à émettre la conjecture suivante :

Il existe un continuum de prédicteurs.

Cette courte analyse tend à valider notre approche de quantification de la diversité d'un ensemble tout en étant facilement intégrable à un paradigme d'entraînement, grâce à l'introduction d'un terme γ (adaptatif) permettant de ne pas dégrader les bonnes prédictions.

Annexe K

Augmentation des données d'images MEB pour la tâche de détection des particules.

L'objectif principal est de créer de "fausses" images de données aussi proches que possible des images réelles sans nécessiter de post-traitement pour obtenir la segmentation de référence. Nous voulons également éviter d'introduire une logique sous-jacente dans les images générées. En partant des images segmentées de référence, chaque agglomérat de chaque image segmentée est extrait pour construire une bibliothèque d'agglomérats (Figure K.1). Par exemple, dans la figure K.1, trois agglomérats sont extraits. L'amélioration des données consiste ensuite à simuler de nouvelles images en appliquant tout d'abord aléatoirement un retournement et une rotation à ces agglomérats, puis en positionnant aléatoirement ces agglomérats de particules sur différents fonds SEM vides.

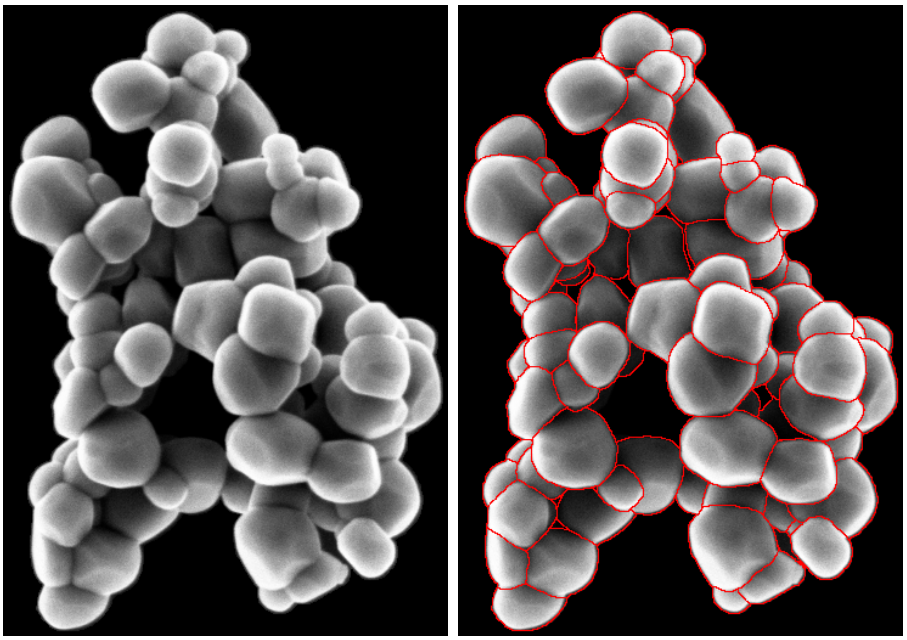


FIGURE K.1 – Agglomérat extrait de particules de TiO_2 (**figure de gauche**) et sa version segmentée (**figure de droite**).

Ainsi, après avoir inséré le premier agglomérat dans le cadre, huit sous-cadres différents sont "créés", dans lesquels l'agglomérat suivant sera placé de manière aléatoire (Figure K.2). La procédure se poursuit jusqu'à ce qu'il n'y ait plus d'espace disponible ou que le nombre maximum d'agglomérats à insérer soit atteint. Enfin, un filtre médian de 5×5 sur toutes les bordures des agglomérats est appliqué afin d'adoucir la transition entre l'image d'arrière-plan et les agglomérats insérés.

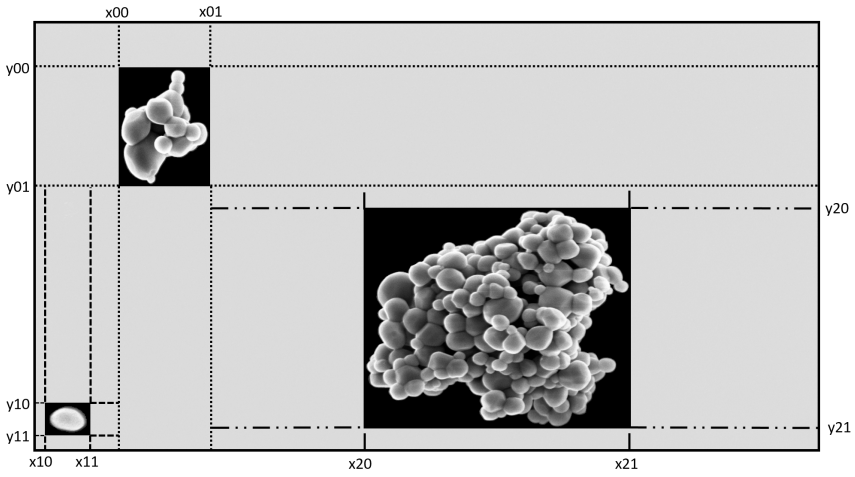


FIGURE K.2 – Schéma de la procédure d'augmentation des données.

La procédure d'augmentation des données proposée est associée à une stratégie d'augmentation des données plus courante, telle que l'application aléatoire d'une ou de plusieurs transformations parmi l'inversion horizontale, l'inversion verticale, le flou gaussien, la normalisation du contraste, le bruit gaussien additif et la multiplication de la valeur des pixels. La figure K.3 montre un exemple d'image simulée.

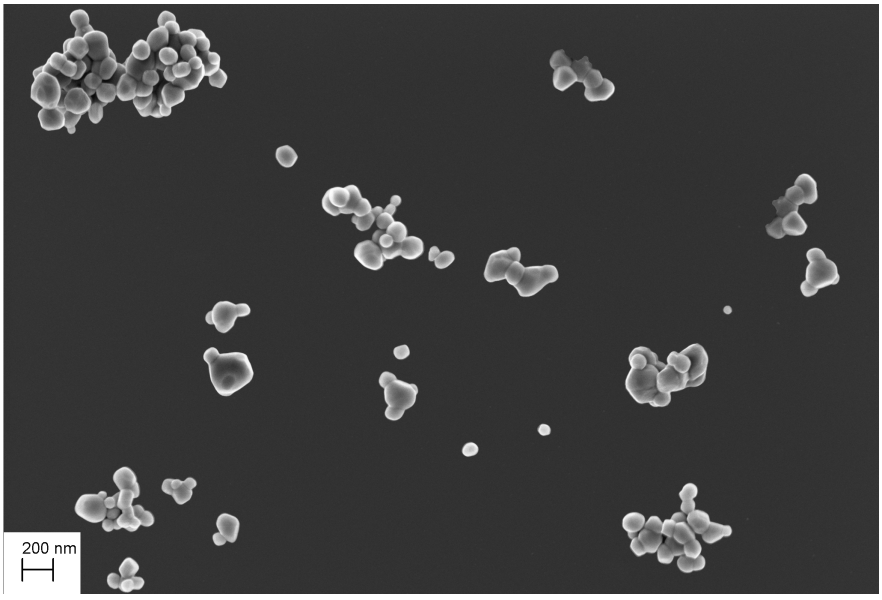


FIGURE K.3 – Image MEB simulée de particules de TiO_2.

ECOLE DOCTORALE

Titre : Quantification d'incertitudes au sein de réseaux de neurones: Application à la mesure automatisée de la taille de particules de TiO_2.

Mots clés : quantification d'incertitudes, réseau de neurones, métrologie, segmentation

Résumé : L'utilisation croissante de solutions technologiques fondées sur des algorithmes d'apprentissage profond a connu une explosion ces dernières années en raison de leurs performances sur des tâches de détection d'objets, de segmentation d'images et de vidéos ou encore de classification, et ce dans de nombreux domaines tels que la médecine, la finance, la conduite autonome ... Dans ce contexte, la recherche en apprentissage profond se concentre de plus en plus sur l'amélioration des performances et une meilleure compréhension des algorithmes utilisés en essayant de quantifier l'incertitude associée à leurs prédictions. Fournir cette incertitude est clé pour une dissémination massive de ces nouveaux outils dans l'industrie et lever les freins actuels pour des systèmes critiques notamment. En effet, fournir l'information de l'incertitude peut revêtir une importance réglementaire dans certains secteurs d'activité.

Ce manuscrit expose nos travaux menés sur la quantification de l'incertitude au sein des réseaux de neurones. Pour commencer, nous proposons un état des lieux approfondi en explicitant les concepts clés impliqués dans un cadre métrologique. Ensuite, nous avons fait le choix de nous concentrer sur la propagation de l'incertitude des entrées à travers un réseau de neurones d'ores-et-déjà entraîné afin de répondre à un besoin industriel pressant. La méthode de propagation de l'incertitude des entrées proposée, nommée WGMprop, modélise les sorties du réseau comme des mixtures de gaussiennes dont la propagation de l'incertitude est assurée par un algorithme Split&Merge muni d'une mesure de divergence choisie comme la distance de Wasserstein. Nous nous sommes ensuite focalisés sur la quantification de l'incertitude inhérente aux paramètres du réseau. Dans ce cadre, une étude comparative des méthodes à l'état de l'art a été réalisée. Cette étude nous a notamment conduit à proposer une méthode de caractérisation locale des ensembles profonds, méthode faisant office de référence à l'heure actuelle. Notre méthodologie, nommée WEUQ, permet une exploration des bassins d'attraction du paysage des paramètres des réseaux de neurones en prenant en compte la diversité des prédicteurs. Enfin, nous présentons notre cas d'application, consistant en la mesure automatisée de la distribution des tailles de nanoparticules de dioxyde de titane (TiO_2) à partir d'images acquises par microscopie électronique à balayage (MEB). Nous décrivons à cette occasion le développement de la brique technologique utilisée ainsi que les choix méthodologiques de quantification d'incertitudes découlant de nos recherches.

Title : Uncertainty quantification in neural networks: Application to the automated measurement of TiO_2 particle size.

Keywords : uncertainty quantification, neural networks, metrology, segmentation

Abstract : The growing use of technological solutions based on deep learning algorithms has exploded in recent years, due to their performance on tasks such as object detection, image and video segmentation and classification, in many fields such as medicine, finance, autonomous driving... In this context, deep learning research is increasingly focusing on improving the performance and understanding of the algorithms used, by attempting to quantify the uncertainty associated with their predictions. Providing this uncertainty is key to the mass dissemination of these new tools in industry, and to overcoming the current obstacles to their use, particularly in critical systems. Indeed, providing information on uncertainty may be of regulatory importance in certain sectors of activity.

This manuscript presents our work on uncertainty quantification in neural networks. To begin with, we provide an in-depth overview, explaining the key concepts involved in a metrological framework. Next, we have chosen to focus on the propagation of input uncertainty through an already-trained neural network, in response to a pressing industrial need. The proposed input uncertainty propagation method, named WGMprop, models the network outputs as mixtures of Gaussians, whose uncertainty propagation is ensured by a SplitMerge algorithm equipped with a divergence measure chosen as the Wasserstein distance. We then focused on quantifying the uncertainty inherent in the network parameters. In this context, a comparative study of state-of-the-art methods was carried out. In particular, this study led us to propose a method for local characterization of deep ensembles, which is currently the standard. Our methodology, named WEUQ, enables an exploration of the basins of attraction of the neural network parameter landscape, taking into account the diversity of predictors. Finally, we present our case study, involving the automated measurement of the size distribution of titanium dioxide (TiO_2) nanoparticles from images acquired by scanning electron microscopy (SEM). We take this opportunity to describe the development of the technology used, and the methodological choices for quantifying the uncertainties arising from our research.

Institut Polytechnique de Paris
91120 Palaiseau, France