Approches pour le renforcement d'une IA embarquée face aux attaques perturbant le federated learning

Molinier Camille
Univ Rennes, CNRS, Inria, IRISA
UMR 6074, F-35000 Rennes
camille.molinier@irisa.fr

Temple Paul
Univ Rennes, CNRS, Inria, IRISA
UMR 6074, F-35000 Rennes
paul.temple@irisa.fr

Zendra Olivier
Univ Rennes, CNRS, Inria, IRISA
UMR 6074, F-35000 Rennes
olivier.zendra@inria.fr

Barais Olivier
Univ Rennes, CNRS, Inria, IRISA
UMR 6074, F-35000 Rennes
olivier.barais@irisa.fr

***Abstract*—Le machine learning est de plus en plus intégré dans nos vies quotidiennes, notamment à travers des systèmes personnalisés. Toutefois, la confidentialité des données pose un défi majeur. L'apprentissage fédéré (FL) répond à cette problématique en permettant l'entraînement de modèles sans partager les données sensibles. Cependant, bien que FL garantisse la confidentialité des données, il reste vulnérable à diverses attaques, comme les attaques par empoisonnement des données ou l'inférence sur les poids du modèle. L'observation faite est que l'ensemble des modèles déployés dans ce contexte FL peuvent être considérés comme des variants du modèle global, faisant ainsi écho au monde de la variabilité logicielle. Cette thèse propose donc d'adapter des techniques de test logiciel et de gestion de la variabilité pour renforcer la sécurité et la robustesse du FL, en tenant compte de son évolution dans le temps et de ses spécificités.**

***Index Terms*—Machine learning, Federated learning, systèmes distribués, test logiciel, V&V**

I. Introduction

Le machine learning se trouve aujourd'hui partout dans notre quotidien (*e.g.*, avec la démocratisation des objets connectés tels que les smartphones, ou bien avec l'apparition des voitures autonomes). Il est notamment utilisé pour permettre la prise de décision personnalisée en prenant en compte les spécificités des utilisateurs (*e.g.*,les goûts musicaux ou de films, les achats récents, *etc*.). Avec le besoin de personnalisation, ces algorithmes se rapprochent des utilisateurs finaux. L'essor de l'IoT et des systèmes embarqués au quotidien fait que les systèmes d'apprentissages font face à de nouveaux défis. Étant proche de l'utilisateur, le modèle manipule des données très précises pour l'entraînement, mais les possibilités de calcul sont limitées. Cependant, lorsque les données sont sensibles ou confidentielles, il est préférable de les garder le plus possible sur l'appareil. Leur envoi vers un serveur distant, comme en machine learning centralisé, est donc impossible (ou non souhaitable). Pour autant, le modèle doit avoir les meilleures performances possibles, ce qui implique une mise à jour en continu pour suivre l'évolution naturelle des données sans perdre la personnalisation. L'apprentissage doit donc se faire localement, c'est ce à quoi répond le federated learning.

Ce travail est financé par la DGA et le projet CyberSkills4All.

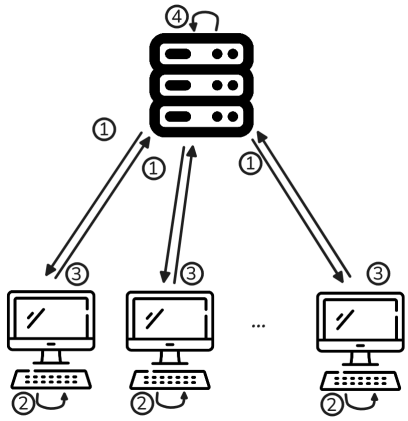

Fig. 1. Schéma de déroulement d'un round en federated learning. (1) Le server envoie le modèle global aux clients. (2) Les clients apprennent avec leurs données sur le modèle reçu. (3) Les clients envoient au server leur modèle local entraîné. (4) Le server agrège les poids des modèles locaux.

II. Background

A. Apprentissage fédéré

Introduit par Google en 2016 [1] pour le clavier Google, l'apprentissage fédéré (*federated learning* - FL) est une méthode d'apprentissage distribuée qui permet d'entraîner des modèles en parallèle et indépendamment sur plusieurs appareils. Cet apprentissage se fait sans partager les données, seuls les poids des modèles sont envoyés pour la mise à jour globale. Cette construction s'adapte bien au contexte du edge-computing avec les contraintes imposées par le Règlement Général sur la Protection des Données (RGPD). De plus, le

fait que les entraînements soient faits sur des données issues de sources différentes offre de la personnalisation mais aussi une certaine variabilité des modèles locaux.

L'entraînement en FL est composé d'une succession de rounds comme montré dans la Figure 1. Un round commence par l'envoi du modèle global du serveur aux différents clients (1). Les clients vont entraîner le modèle reçu avec leurs données locales (2). C'est lors de cette étape que le processus de personnalisation a lieu. Lorsque les modèles locaux sont entraînés, ils sont renvoyés au serveur (3). Le serveur va alors agréger les poids des modèles reçus (4).

Toute la difficulté du FL réside dans l'agrégation [2]. Il s'agit du mécanisme qui permet la combinaison des poids des différents modèles locaux. La méthode conventionnelle pour effectuer l'agrégation des modèles locaux est la FedAvg [3]. Cette méthode combine les poids des modèles en effectuant la moyenne des poids pondérées par la taille de leur jeu d'entraînement par rapport au total des données présentes vues par les clients comme le montre l'équation 1. L'agrégation est le premier mécanisme de confidentialité du FL. D'autres méthodes d'agrégation existent pour garantir plus de convergence [4], de robustesse [5] ou d'équité [6].

$$M_{avg} = \sum_{i=1}^{k} \frac{n_i}{N} M_i \tag{1}$$

Avec M_{avg} le nouveau modèle global issu de l'agrégation, k le nombre de clients, N la taille total des données d'entraînement de la fédération telle que $N = n_1 + n_2 + ... + n_k$ et M_i le modèle local reçu du client i.

B. L'apprentissage fédéré dans un contexte hostile

Comme le machine learning centralisé qui peut-être utilisé dans un contexte hostile, le FL peut être sujet à des attaques à partir de clients [7]. Le non partage des données en FL augmente le respect de la confidentialité, mais ne garantit pas la sécurité. Les attaques par empoisonnement [8] des données permettent à un client malveillant ou corrompu de perturber les données l'entraînement afin de corrompre son modèle local qui sera propagé pour l'agrégation. Un attaquant peut aussi introduire une backdoor [9] dans le modèle pour réagir à un pattern spécifique de manière à changer le comportement de ce dernier lorsqu'il détecte la présence du pattern en question. Une autre manière d'attaquer un modèle global s'appuie sur l'accès aux poids du modèle (puisque le modèle est téléchargé sur les clients). Il est ainsi possible d'en abuser de façon à inférer la présence ou non de données lors de l'entraînement et donc extraire des informations des autres clients [10].

Puisque des attaques sont possibles, d'autres travaux se sont aussi attachés à développer ou adapter des mécanismes de défense pour le FL [11]. La confidentialité différentielle [12] introduit du bruit, souvent gaussien ou laplacien, dans les poids du modèle. Cette technique limite l'impact de la présence ou non d'une donnée dans les jeux d'entraînement locaux. Le chiffrement homomorphe [13] est aussi utilisé pour prévenir la fuite des modèles lors des échanges entre le serveur et les clients. Cependant, ces techniques ne sont pas adoptées en pratique. Le chiffrement homomorphe est trop gourmand en ressources, ce qui n'est pas toujours favorable dans le contexte du FL. De plus, comme le serveur ne manipule que des poids chiffrés, il est impossible d'appliquer, au niveau du serveur, des mécanismes de défense ou de contrôle ce qui ouvre la porte aux attaques. La confidentialité différentielle permet de mitiger les impacts des attaques, mais dans certains contextes, cela rend plus facile d'effectuer certaines attaques [14].

III. Motivation

Une des propriétés du machine learning est que le comportement du modèle entrainé ne dépend pas du code, contrairement à un système classique, mais des données. L'emploi d'initialisation aléatoire, le découpage en batch de données, la séparation en ensemble d'apprentissage et de validation sont autant de points de variations qui peuvent influencer le comportement final du modèle. En machine learning distribué, les données étant issues de plusieurs clients, les modèles ne voient pas exactement la même distribution de données. Ces données étant collectées lors de la phase de production, elles sont aussi soumises à une évolution naturelle de leur distribution.

Cet ensemble de modèles créés par le cadre de travail de l'apprentissage fédéré fait écho à l'étude de la variabilité logicielle dans le domaine de la science du logiciel. La construction de logiciel moderne demande de prendre en compte les besoins utilisateur. Pour cela, les fonctionnalités sont découpés en briques composables et réutilisables. Ainsi, la génération d'une nouvelle version du logiciel pour un besoin spécifique se fait uniquement en sélectionnant les briques à utiliser et intégrer. La réutilisation comme première préoccupation du développement logiciel permet ainsi de faciliter le debugging, de réduire les coûts de développement de façon significative et de réduire le "time-to-market". Par contre, cela demande également de réfléchir en amont à des mécanismes permettant de facilement étendre les briques existantes et d'en intégrer de nouvelles provenant des demandes de clients. La composition permet alors de créer un nombre important de variants du même système. Le noyau Linux par exemple propose plus de 20000 briques pour un total estimé de plus de 2^{20000} variants uniques possibles. Cette notion de variants fait écho aux différents modèles qui sont en action dans un cadre d'apprentissage fédéré. Ainsi, s'assurer du bon fonctionnement des modèles dans un contexte d'apprentissage fédéré (notamment d'un point de vue de la dérive des modèles liée à l'évolution des donnés) pourrait bénéficier d'un point de vue provenant de la gestion de la variabilité logicielle.

IV. Approche

Le contexte du FL et la variabilité logicielle présentent de plusieurs intersections. Toutefois, ce sont deux communautés

interagissent peu, de part leurs centres d'intérêts et le prisme par lequel ils les abordent. L'approche de cette thèse tente de rapprocher ces deux mondes. S'inspirer des techniques de gestion de la variabilité logicielle et des techniques de tests logiciels associées et de les appliquer au FL.

Actuellement, nous mettons en place un framework configurable et extensible, construit autour du framework de federated learning Flower [15], qui a été choisi du fait de sa conception en tant que librairie python open-source, avec une communauté active, offrant une compatibilité avec la plupart des frameworks d'apprentissages, une liberté de contrôle de l'implémentation du serveur et des clients. Le framework que nous développons vise à permettre l'intégration de différentes attaques contre du FL et de les confronter à différentes défenses, y compris celles que nous chercherons à développer en s'inspirant du test logiciel.

La première attaque que nous avons choisie d'étudier est une attaque par empoisonnement des données centrée autour des Generative Adversarial Networks (GANs) [16]. Cette attaque se place dans un contexte dans lequel les différents participants du processus de federated learning apprennent sur des données très confidentielles (un algorithme de reconnaissance de visage pour déverrouiller un smartphone par exemple) et donc avec très peu de recouvrement. L'attaque profite de la capacité des GANs à imiter les distributions de données pour reconstruire des exemples de données des autres clients. Le GAN est entraîné sur les données locales de l'attaquant, puis utilisé pour générer de nouvelles données qui ressemblent à celles des autres participants. Ces données constituent un dataset dont les labels sont changés. Ce dataset est utilisé pour l'entraînement du modèle local de l'attaquant. Lors de la phase d'agrégation, les mises à jour du modèle local de l'attaquant, contenant les données empoisonnées, seront propagées au modèle global, corrompant ainsi le modèle global. Cette attaque peut avoir des conséquences graves sur le modèle global, en réduisant sa précision et en compromettant sa capacité à effectuer correctement sa tâche. Cette attaque peut aussi permettre de viser un participant du processus d'apprentissage fédéré de façon à lui introduire un déni de service spécifique.

V. Conclusion

La mise en place de plus en plus omniprésentes de systèmes utilisant des modèles de machine learning rapprochent ces derniers des utilisateurs finaux. Face aux préoccupations liées à la confidentialité des données de ces derniers, l'apprentissage fédéré semble être une solution adaptée. Cette technique d'apprentissage offre une variabilité et une personnalisation des modèles individuels de chaque client. Toutefois, ce système reste vulnérable aux attaques. Bien que chaque modèle soit différent dans son utilisation, ils visent tous à réaliser la même tâche ce qui fait écho au domaine de la gestion de la variabilité logicielle. En particulier, la préoccupation de s'assurer que chaque variant logiciel fonctionne correctement semble aligner avec le problème de s'assurer du bon comportement des modèles de machine learning qui peuvent se faire attaquer ou dévier au cours du temps. C'est pourquoi cette thèse vise à croiser ces deux mondes et tente de construire de nouveaux mécanismes de défense.

References

[1] H. B. McMahan, E. Moore, D. Ramage, and B. A. y Arcas, "Federated learning of deep networks using model averaging," *CoRR*, vol. abs/1602.05629, 2016. [Online]. Available: http://arxiv.org/abs/1602.05629

[2] K. Nakayama and G. Jeno, *Federated Learning with Python: Design and Implement a Federated Learning System and Develop Applications Using Existing Frameworks*. Packt Publishing, Limited, 2022. [Online]. Available: https://books.google.fr/books?id=6oN5zwEACAAJ

[3] H. B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," 2023. [Online]. Available: https://arxiv.org/abs/1602.05629

[4] S. Reddi, Z. Charles, M. Zaheer, Z. Garrett, K. Rush, J. Konečný, S. Kumar, and H. B. McMahan, "Adaptive federated optimization," 2021. [Online]. Available: https://arxiv.org/abs/2003.00295

[5] K. Pillutla, S. M. Kakade, and Z. Harchaoui, "Robust aggregation for federated learning," *IEEE Transactions on Signal Processing*, vol. 70, p. 1142–1154, 2022. [Online]. Available: http://dx.doi.org/10.1109/TSP.2022.3153135

[6] T. Li, S. Hu, A. Beirami, and V. Smith, "Ditto: Fair and robust federated learning through personalization," 2021. [Online]. Available: https://arxiv.org/abs/2012.04221

[7] J. Hasan, "Security and privacy issues of federated learning," 2023. [Online]. Available: https://arxiv.org/abs/2307.12181

[8] B. Biggio and F. Roli, "Wild patterns: Ten years after the rise of adversarial machine learning," *Pattern Recognition*, vol. 84, p. 317–331, Dec. 2018. [Online]. Available: http://dx.doi.org/10.1016/j.patcog.2018.07.023

[9] E. Bagdasaryan, A. Veit, Y. Hua, D. Estrin, and V. Shmatikov, "How to backdoor federated learning," 2019. [Online]. Available: https://arxiv.org/abs/1807.00459

[10] A. Suri, P. Kanani, V. J. Marathe, and D. W. Peterson, "Subject membership inference attacks in federated learning," 2023. [Online]. Available: https://arxiv.org/abs/2206.03317

[11] C. Zhang, Y. Xie, H. Bai, B. Yu, W. Li, and Y. Gao, "A survey on federated learning," *Knowledge-Based Systems*, vol. 216, p. 106775, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0950705121000381

[12] A. Banse, J. Kreischer, and X. O. i Jürgens, "Federated learning with differential privacy," 2024. [Online]. Available: https://arxiv.org/abs/2402.02230

[13] W. Jin, Y. Yao, S. Han, J. Gu, C. Joe-Wong, S. Ravi, S. Avestimehr, and C. He, "Fedml-he: An efficient homomorphic-encryption-based privacy-preserving federated learning system," 2024. [Online]. Available: https://arxiv.org/abs/2303.10837

[14] J. Wang, R. Schuster, I. Shumailov, D. Lie, and N. Papernot, "In differential privacy, there is truth: On vote leakage in ensemble private learning," 2022. [Online]. Available: https://arxiv.org/abs/2209.10732

[15] D. J. Beutel, T. Topal, A. Mathur, X. Qiu, J. Fernandez-Marques, Y. Gao, L. Sani, K. H. Li, T. Parcollet, P. P. B. de Gusmão, and N. D. Lane, "Flower: A friendly federated learning research framework," 2022. [Online]. Available: https://arxiv.org/abs/2007.14390

[16] J. Zhang, J. Chen, D. Wu, B. Chen, and S. Yu, "Poisoning attack in federated learning using generative adversarial nets," in *2019 18th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/13th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE)*, 2019, pp. 374–380.