République Algérienne démocratique et populaire

Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
École nationale supérieure d'informatique (ESI ex. INI)

Project BIG DATA MINING[BDM]

2CS
2024-2025

Option: Computer Systems and Software (SIL1)

Federated Learning

Réalisé par :

- Rezzoug Aicha
- Dinari Yasmine
- Djabri Maroua
- Bouyahiaoui Meriem

Encadré par :

HAMDAD Leila

Sommaire :

1. Introduction

Avec l'avancement rapide du Big Data et de l'intelligence artificielle, la gestion des données devient un enjeu majeur, notamment en termes de confidentialité, sécurité et efficacité de calcul. Le Federated Learning (FL), ou apprentissage fédéré, est une approche d'apprentissage automatique qui vient résoudre ces problèmes.

L'objectif de ce projet est d'explorer les concepts fondamentaux du Federated Learning, d'analyser son algorithme et d'évaluer ses performances à travers une implémentation pratique.

Ce rapport commencera par une présentation du Federated Learning, de ses caractéristiques et de ses applications. Nous détaillerons ensuite le dataset utilisé, la méthodologie adoptée et notre objectif de classification. Enfin, le lien vers le notebook de l'implémentation avec des explications plus détaillées.

2. Définition du Federated Learning

L'apprentissage fédéré (FL) est un paradigme émergent en apprentissage automatique qui permet l'entraînement collaboratif de modèles sur plusieurs appareils ou serveurs sans transférer les données brutes vers un emplacement central. Cette approche répond aux défis majeurs liés à la confidentialité des données, à la sécurité et à la conformité réglementaire, ce qui la rend particulièrement adaptée aux applications impliquant des informations sensibles.
Contrairement aux méthodes d'apprentissage automatique traditionnelles qui nécessitent des ensembles de données centralisés, l'apprentissage fédéré permet aux appareils d'entraîner les modèles localement et de ne partager que les mises à jour du modèle. Cela garantit que les données privées restent sur l'appareil de l'utilisateur, renforçant ainsi la confiance et la conformité aux réglementations en matière de confidentialité.

3. Caractéristiques du FL

1. Décentralisation des Données

L'une des principales caractéristiques de l'apprentissage fédéré est que les données restent réparties sur plusieurs appareils ou serveurs. Au lieu d'être envoyées vers un serveur central, chaque appareil effectue un entraînement local du modèle et ne transmet que les paramètres du modèle à l'agrégateur central.
Exemple : Dans une application de clavier mobile comme Google Gboard, les données de texte des utilisateurs ne quittent jamais leurs téléphones. L'application apprend localement la manière dont les utilisateurs tapent et envoie uniquement des mises à jour du modèle au serveur central.

2. Préservation de la Confidentialité

En conservant les données brutes sur les appareils locaux, l'apprentissage fédéré améliore la confidentialité et la sécurité des données. Cette approche réduit le risque de violations de données et assure la conformité avec des réglementations comme le Règlement Général

sur la Protection des Données (RGPD) et la loi HIPAA sur la portabilité et la responsabilité en matière d'assurance maladie.
Exemple : Dans le domaine de la santé, l'apprentissage fédéré permet aux hôpitaux d'entraîner des modèles sur des dossiers médicaux sans partager les données sensibles elles-mêmes.

3. Apprentissage Collaboratif

Plusieurs appareils ou nœuds contribuent à l'entraînement d'un modèle global unique. Chaque appareil entraîne le modèle localement sur ses propres données et envoie ensuite des mises à jour du modèle, qui sont agrégées pour améliorer la performance globale du modèle.
Exemple : Les véhicules autonomes peuvent utiliser l'apprentissage fédéré pour partager des informations issues de leurs expériences de conduite locales sans divulguer de données propriétaires.

4. Communication Asynchrone

Les systèmes d'apprentissage fédéré peuvent fonctionner de manière asynchrone, ce qui signifie que les appareils peuvent mettre à jour le modèle à des moments différents, en fonction de leur disponibilité et de leur capacité de calcul.
Exemple : Les smartphones peuvent effectuer des mises à jour du modèle uniquement lorsqu'ils sont branchés et connectés en Wi-Fi, afin d'économiser la batterie et la consommation de données.

5. Efficacité des Ressources

L'apprentissage fédéré réduit les coûts de transmission des données en envoyant uniquement des mises à jour du modèle au lieu d'ensembles de données volumineux. Cela rend cette approche adaptée aux appareils ayant une bande passante réseau limitée.
Exemple : Les appareils IoT dans les maisons intelligentes peuvent entraîner des modèles localement sur les données des capteurs et envoyer des mises à jour périodiques au cloud sans saturer le réseau.

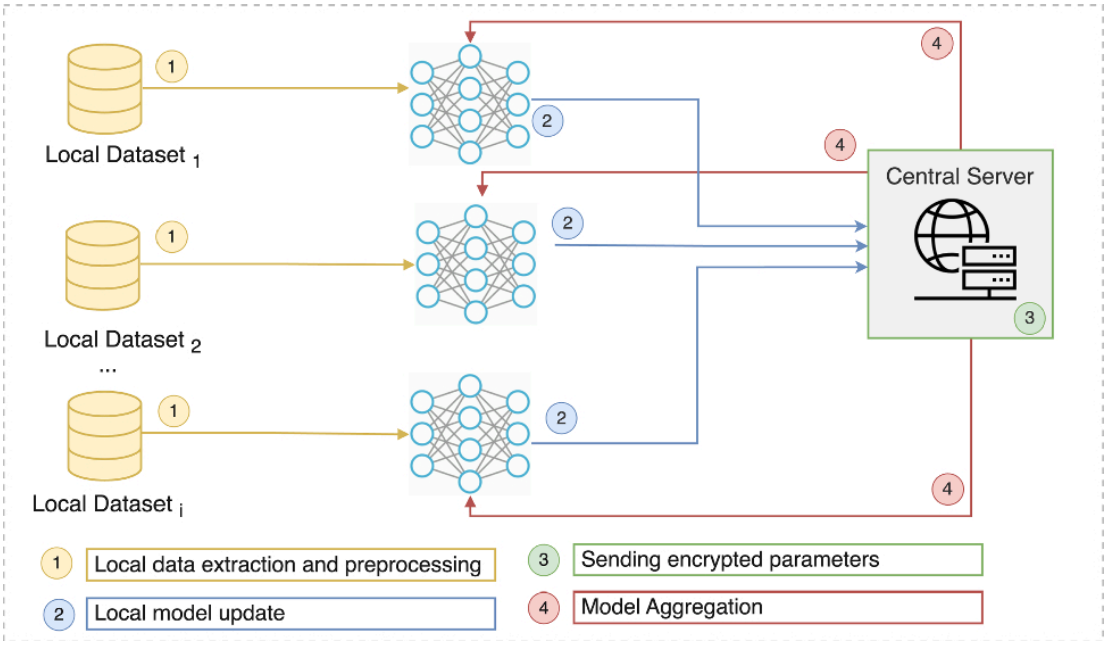
6. Modèles Personnalisés

L'apprentissage fédéré permet de personnaliser les modèles pour chaque appareil tout en contribuant à l'amélioration du modèle global. Cela améliore l'expérience utilisateur sans compromettre les performances globales du modèle.
Exemple : Les applications de suivi d’activité physique peuvent adapter les prédictions du nombre de pas en fonction des habitudes de chaque utilisateur tout en améliorant le modèle général pour tous les utilisateurs.

7. Tolérance aux Pannes

Le système d'apprentissage fédéré est robuste face aux pannes d'appareils. Même si certains appareils sont hors ligne ou ne peuvent pas envoyer de mises à jour, le modèle continue de s'entraîner avec les appareils restants.
Exemple : Dans un réseau de compteurs intelligents, si un compteur devient inactif, le système continue d'agréger les mises à jour des autres compteurs sans interruption.

3. L'objectif du FL



L'Apprentissage Fédéré vise à atteindre plusieurs objectifs principaux, qui reflètent ses avantages et ses applications uniques :

→ Préservation de la confidentialité des données : L'objectif central de FL est de permettre l'entraînement de modèles d'apprentissage automatique sans nécessiter le transfert des données brutes vers un serveur central. Cela réduit les risques de violation de la vie privée et garantit que les données sensibles restent localisées sur les appareils des utilisateurs ou dans les organisations 145.

→ Exploitation des données décentralisées : FL permet d'utiliser efficacement des ensembles de données répartis sur plusieurs appareils ou entités (par exemple, smartphones, capteurs IoT, institutions médicales) tout en respectant les contraintes de confidentialité et de sécurité 27.

→ Création d'un modèle global optimisé : L'un des objectifs techniques est de produire un modèle global performant en combinant les mises à jour des modèles locaux, tout en minimisant une fonction de perte globale. Ce modèle global est conçu pour répondre aux besoins collectifs des clients participants 14.

→ Réduction des coûts liés à la centralisation : En décentralisant le processus d'entraînement, FL diminue la dépendance aux infrastructures cloud

centralisées, ce qui réduit les coûts liés à la bande passante et au stockage nécessaires pour transférer de grandes quantités de données 15.

→ Adaptation à l'hétérogénéité des données : Contrairement à l'apprentissage distribué classique, FL est conçu pour fonctionner avec des ensembles de données hétérogènes et non identiques (non-IID), souvent déséquilibrés entre les différents clients 26.

→ Applications dans des domaines sensibles : FL permet d'entraîner des modèles dans des secteurs où la confidentialité et la sécurité sont critiques, comme la santé, la finance ou les télécommunications. Par exemple, il peut être utilisé pour développer des modèles prédictifs sans compromettre les données médicales ou financières des utilisateurs 78.

En résumé, l'Apprentissage Fédéré cherche à concilier efficacité dans l'entraînement des modèles d'IA avec le respect strict de la confidentialité et une gestion optimale des ressources décentralisées.

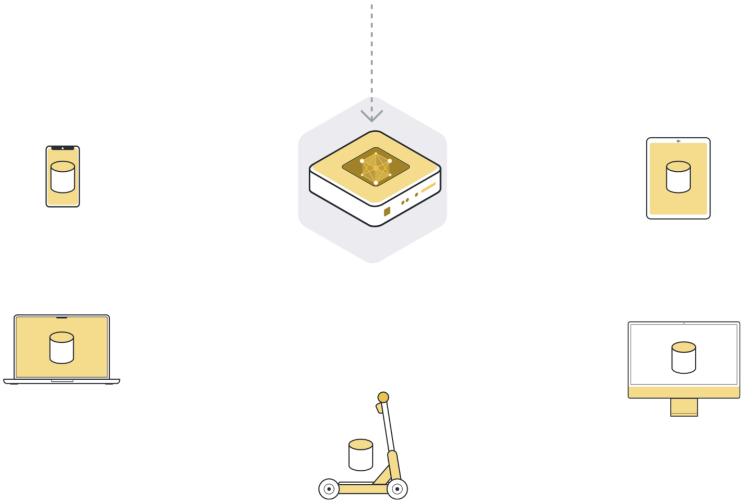
4. L'algorithme de FL

L'algorithme de l'apprentissage fédéré peut être réparti en 5 grandes étapes:

Étape 0 : Initialisation du modèle global

Nous commençons par initialiser le modèle sur le serveur. C'est exactement la même chose dans l'apprentissage centralisé classique : nous initialisons les paramètres du modèle, soit de façon aléatoire, soit à partir d'un point de contrôle précédemment sauvegardé.

Soit w_0 les poids initiaux du modèle global : $w_0 \sim N(0, \sigma^2)$

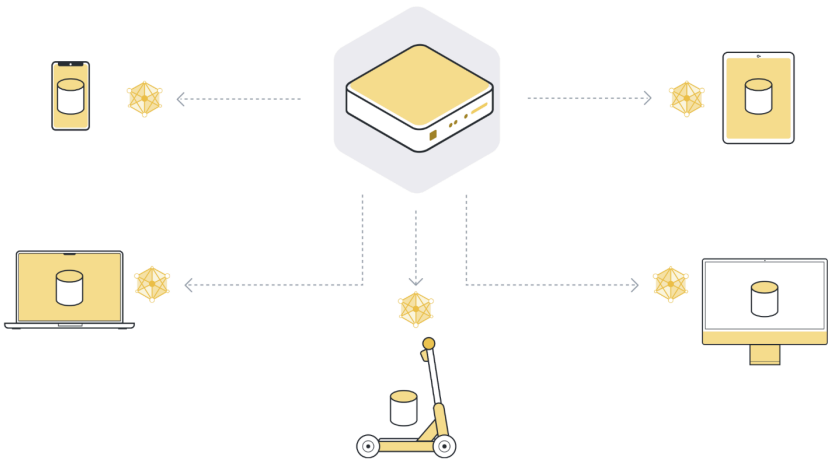


Étape 1 : Envoyer le modèle à un certain nombre d'organisations/appareils connectés (nœuds clients)

Ensuite, nous envoyons les paramètres du modèle global aux nœuds clients connectés (pensez aux appareils en périphérie comme les smartphones ou aux serveurs appartenant à des organisations). Cela permet de s'assurer que chaque nœud participant commence son entraînement local avec les mêmes paramètres de modèle. Nous utilisons souvent seulement quelques-uns des nœuds connectés plutôt que l'ensemble des nœuds. La raison en est que l'ajout d'un nombre croissant de nœuds clients apporte des bénéfices décroissants.

Le serveur envoie les paramètres du modèle w_t aux K clients sélectionnés : $w_t^{(k)} = w_t$ pour tout $k \in K$.

où K est l'ensemble des clients participant à cette itération.



Étape 2 : Entraîne le modèle localement sur les données de chaque organisation/appareil (nœud client)

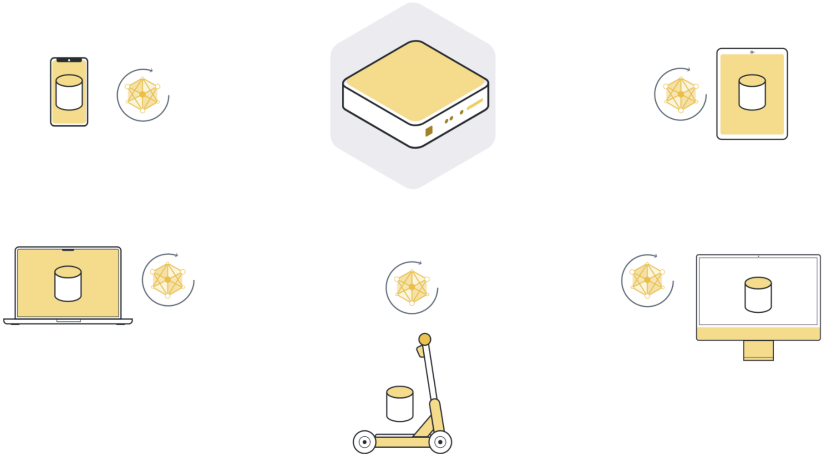
Maintenant que tous les nœuds clients (sélectionnés) disposent de la dernière version des paramètres du modèle global, ils commencent l'entraînement local. Ils utilisent leur propre ensemble de données locales pour entraîner leur propre modèle local. Ils n'entraînent pas le modèle jusqu'à la convergence totale, mais ils ne s'entraînent que pendant un petit moment. Il peut s'agir d'une seule époque sur les données locales, ou même de quelques étapes (mini-batchs).

Chaque client met à jour ses poids en fonction de ses propres données locales D_k en appliquant la descente de gradient stochastique (SGD) : $w_{(t+1)}^{(k)} = w_t^{(k)} - \eta \nabla F_k(w_t^{(k)})$

où :

- η est le taux d'apprentissage,
- $\nabla F_k(w_t^{(k)})$ est le gradient de la fonction de perte locale du client k.

Les clients s'entraînent seulement pendant quelques époques ou mini-batchs.

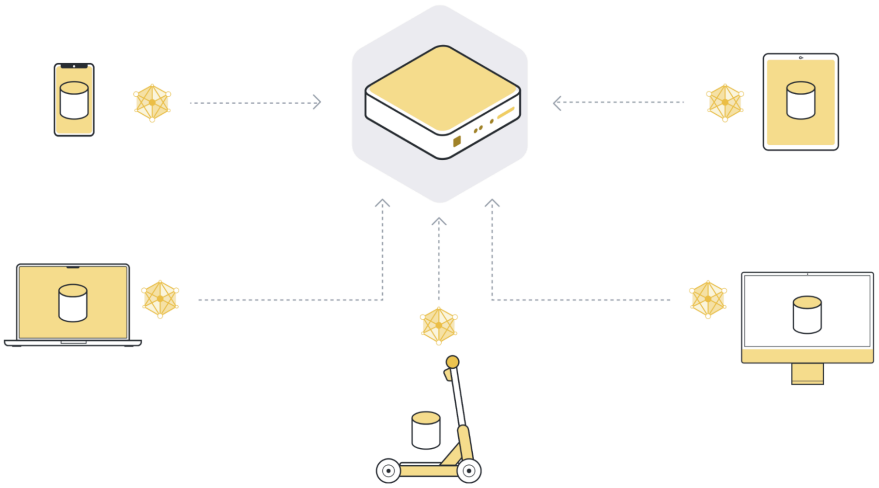


Étape 3 : Renvoyer les mises à jour du modèle au serveur

Après l'entraînement local, chaque nœud client possède une version légèrement différente des paramètres du modèle qu'il a reçus à l'origine. Les paramètres sont tous différents parce que chaque nœud client a des exemples différents dans son ensemble de données local. Les nœuds clients renvoient ensuite ces mises à jour du modèle au serveur. Les mises à jour du modèle qu'ils envoient peuvent être soit les paramètres complets du modèle, soit seulement les gradients qui ont été accumulés au cours de l'entraînement local.

Chaque client envoie les mises à jour au serveur sous l'une des formes suivantes :

1. Envoi des poids complets : $w_{t+1}^{(k)}$
2. Envoi des gradients calculés : $\nabla F_k(w_t^{(k)})$



Étape 4 : Agréger les mises à jour des modèles dans un nouveau modèle global

Le serveur reçoit les mises à jour du modèle des nœuds clients sélectionnés. S'il a sélectionné 100 nœuds clients, il dispose maintenant de 100 versions légèrement différentes du modèle global original, chacune ayant été formée sur les données locales d'un client.

Afin d'obtenir un modèle unique, nous devons combiner toutes les mises à jour de modèle reçues des nœuds clients. Ce processus est appelé agrégation, et il existe plusieurs méthodes pour le réaliser. La méthode la plus basique est appelée Federated Averaging (McMahan et al., 2016), souvent abrégée en FedAvg.

FedAvg prend les 100 mises à jour de modèle et, comme son nom l'indique, en calcule la moyenne. Plus précisément, il effectue une moyenne pondérée des mises à jour du modèle, en pondérant par le nombre d'exemples utilisés par chaque client pour l'entraînement. Cette pondération est essentielle pour garantir que chaque exemple de données ait la même 'influence' sur le modèle global résultant.

Par exemple, si un client possède 10 exemples et un autre 100 exemples, alors sans pondération, chaque exemple du premier client influencerait le modèle global dix fois plus que chaque exemple du second client.

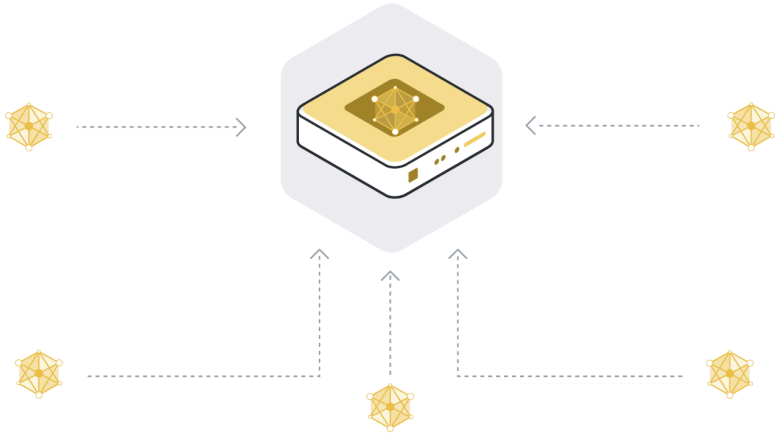
Le serveur calcule la moyenne pondérée des modèles locaux pour mettre à jour le modèle global :

- $w_{t+1} = \Sigma_{(k \in K)} (n_k / N) * w_{t+1}^{(k)}$

où :

- $N = \Sigma_{(k \in K)} n_k$ est le nombre total d'exemples de données,

- n_k est le nombre d'exemples locaux pour le client k,
- w_(t+1)^(k) est le modèle mis à jour par le client k.



Étape 5 : Répéter les étapes 1 à 4 jusqu'à ce que le modèle converge

Les étapes 1 à 4 constituent ce que nous appelons un cycle unique d'apprentissage fédéré. Les paramètres du modèle global sont envoyés aux nœuds clients participants (étape 1), les nœuds clients s'entraînent sur leurs données locales (étape 2), ils envoient leurs modèles mis à jour au serveur (étape 3), et le serveur agrège ensuite les mises à jour du modèle pour obtenir une nouvelle version du modèle global (étape 4).

Lors d'un seul tour, chaque nœud client participant à cette itération ne s'entraîne que pendant une courte période. Cela signifie qu'après l'étape d'agrégation (étape 4), nous obtenons un modèle qui a été entraîné sur l'ensemble des données des nœuds clients participants, mais seulement pendant un temps limité. Nous devons donc répéter ce processus d'entraînement encore et encore afin d'obtenir, à terme, un modèle entièrement entraîné et performant sur l'ensemble des données des nœuds clients.

Le modèle est mis à jour jusqu'à ce que la fonction de perte atteigne un seuil ε :

$|F(w_{t+1}) - F(w_t)| < \varepsilon$

où F(w) est la fonction de perte globale.

- Comparaison des Algorithmes de Federated Learning :

Algorithme	Description	Avantages	Inconvénients	Complexité Computationnelle	Stabilité sur Données Non-IID
FedAvg	Algorithme de base basé sur la descente de gradient avec une simple moyenne pondérée des modèles locaux.	- Faible coût de communication et de calcul - Facile à implémenter	- Convergence lente sur données hétérogènes (non-IID) - Peut générer des modèles instables	(Faible)	(Mauvaise stabilité)
FedProx	Ajoute un terme de régularisation pour limiter les mises à jour locales trop divergentes du modèle global.	- Améliore la convergence sur données non-IID - Plus stable que FedAvg	- Nécessite un réglage fin du paramètre de régularisation - Augmente légèrement le coût computationnel	(Moyenne)	(Moyenne stabilité)
FedAdam	Utilise une optimisation adaptative côté serveur inspirée de l'algorithme Adam.	- Convergence plus rapide que FedAvg sur des données hétérogènes - Réduit la variabilité entre clients	- Sensible au choix des hyperparamètres	(Moyenne)	(Bonne stabilité)

FedYogi	Version améliorée de FedAdam, introduisant une adaptation plus fine des gradients.	- Meilleure convergence que FedAdam - Plus robuste aux variations des gradients	- Peut être instable sur certaines distributions de données	(Moyenne)	(Bonne stabilité)
SCAFFOLD	Corrige la variance des gradients en introduisant des contrôles variés pour mieux aligner les modèles locaux et globaux.	- Convergence rapide en non-IID - Meilleure correction des écarts entre clients	- Double le coût de communication (envoi des variables de contrôle) - Sensible aux déséquilibres de classes	(Élevée)	(Mauvaise stabilité)
FedDyn	Minimise l'écart entre objectif local et global via une régularisation dynamique.	- Meilleure précision et stabilité entre clients - Moins sensible à l'hétérogénéité des données	- Très coûteux computationnellement - Sensible aux problèmes de convergence sans clipping de gradient	(Très élevée)	(Excellente stabilité)

5. L'implémentation du FL avec classification supervisée (RN)

➔ Choix de la dataset

Notre choix de la dataset s'est porté sur MIMIC-III (Medical Information Mart for Intensive Care III), qui est une base de données médicales publique.Elle contient des données cliniques anonymisées de 46 520 patients distincts, représentant 58 976 hospitalisations dans des unités de soins intensifs, et a plus de 2 millions d'observations au total.

Le nombre d'attributs dépasse 300, incluant des informations démographiques (âge, sexe), des signes vitaux (fréquence cardiaque, pression artérielle, température), des résultats de laboratoire (taux de lactate, créatinine, bilirubine, etc.), des diagnostics codés (ICD-9), des traitements administrés, et des scores de gravité (SOFA, SAPS-II). Son utilisation dans le federated learning permet de préserver la confidentialité des données, un aspect crucial dans le domaine médical, ce qui était la raison la plus importante de notre choix.

➔ Algorithme de Federated learning utilisé

L'algorithme choisi est FedAvg (Federated Averaging), dont nous avons parlé dans la partie algorithmes du FL. Comme cité précédemment, c'est l'algorithme de base du federated learning. Il est simple à implémenter, efficace et offre une bonne convergence pour des tâches de classifications comme la nôtre.

➔ Choix de la classification

Nous avons choisi la prédiction de la mortalité, un problème binaire où le modèle doit prédire si un patient décédera ou survivra pendant son séjour à l'hôpital. C'est un problème important car il permet d'identifier les patients en état critique et d'adapter les traitements en conséquence. Le modèle utilisé est un réseau de neurones simple avec une couche cachée, une fonction d'activation ReLU, et une couche de sortie avec une sigmoïde pour produire une probabilité de mortalité. La fonction perte utilisée pour évaluer notre modèle est BCEWithLogitsLoss, et la bibliothèque que nous avons utilisée pour concevoir notre implémentation est PyTorch de Python.

6. Implémentation du FL

Lien vers le notebook :

7. Conclusion

Le Federated Learning représente une avancée majeure dans l'apprentissage distribué, qui offre une solution efficace pour préserver la confidentialité des données tout en permettant un entraînement collaboratif de modèles d'intelligence artificielle. Grâce à ce projet, nous avons observé les avantages de cette approche, notamment sa capacité à traiter des données décentralisées et à exploiter la parallélisation pour accélérer l'apprentissage.

Cependant, plusieurs défis demeurent. Parmi eux, la communication entre les clients et le serveur central, qui peut entraîner une latence et une consommation élevée de bande passante. De plus, la non-uniformité des données distribuées (hétérogénéité des clients) peut impacter la convergence du modèle et nécessiter des stratégies d'équilibrage et d'agrégation adaptées. Enfin, la sécurité et l'intégrité des mises à jour envoyées par les clients restent des préoccupations majeures, car des attaques comme les model poisoning peuvent compromettre l'ensemble du système.

Malgré ces défis, le Federated Learning ouvre de nouvelles perspectives pour l'apprentissage collaboratif sécurisé et reste l'une des solutions les plus efficaces dans ce domaine.

Références :

https://arxiv.org/pdf/2403.17287

https://flower.ai/docs/framework/tutorial-series-what-is-federated-learning.html

https://manta-tech.io/articles/FL/

https://en.wikipedia.org/wiki/Federated_learning

https://www.amazon.science/publications/towards-multi-objective-statistically-fair-federated-learning

https://fr.wikipedia.org/wiki/Apprentissage_f%C3%A9d%C3%A9r%C3%A9

https://www.innovatiana.com/post/federated-learning

https://arxiv.org/abs/2310.09866

https://flower.ai/docs/framework/main/fr/tutorial-series-what-is-federated-learning.html

https://france-science.com/le-federated-learning-un-paradigme-dapprentissage-en-plein-essor-aux-etats-unis/

https://cs.uwaterloo.ca/~z97hu/presentation/FedMGDA_seminar_talk_website.pdf

https://www.sfds.asso.fr/sdoc-6887-32bb0c39cff6f70a92b7621b470bf181-sl_drt_20_0661_v2.pdf

https://allonia.com/federated-learning-revolutionner-lia-en-preservant-la-confidentialite-des-donnees-2/

https://ieeexplore.ieee.org/document/9573567/

https://konfuzio.com/fr/apprentissage-federe/

https://frq.gouv.qc.ca/projet/apprentissage-par-renforcement-pour-lapprentissage-federe/

https://arxiv.org/abs/2501.03392

https://www.scaleoutsystems.com/what-is-federated-learning

https://www.nature.com/articles/s41591-021-01506-3

https://research.ibm.com/blog/what-is-federated-learning

https://www.deepchecks.com/glossary/federated-learning/