Les algorithmes de Federated Learning

karim houidi

November 2023

Contents

List of Tables

1 Définitions de FL

Références	Définitions
[9]	L'apprentissage fédéré (Federated Learning) propose d'avoir un ensemble de périphériques extrêmes pour effectuer des tâches d'apprenant localement et de ne communiquer qu'un modèle mis à jour à un serveur de coordination.Un serveur apprend un modèle global partagé en regroupant des modèles formés localement à partir d'un nombre potentiellement très important de clients.Les clients ont généralement des données déséquilibrées et non-i.i.d. (indépendant et distribué à l'identique) ainsi que des capacités de transfert de données limitées.
[6]	Federated Learning implique l'apprentissage collaboratrive des modèles DNN (Deep Neural Network) sur les terminaux.Il y a, en général, deux étapes dans le processus de FL,l'apprentissage des modèles locaux sur les périphériques finaux et l'agrégation globale des paramètres mis à jour dans le serveur FL. FL permet aux utilisateurs de former en collaboration un modèle partagé tout en conservant des données personnelles sur leurs appareils, soulageant ainsi leurs préoccupations en matière de confidentialité.
[7]	Federated Learning, c'est une approche décentralisée qui plaidons pour une alternative qui laisse les données d'entraînement distribuées sur les appareils mobiles, et apprend un modèle partagé en agrégant les mises à jour informatisées localement.

Table 1: Définitions FL

2 Les Algorithmes de FL

2.1 Federated Averaging (FedAvg)

2.1.1 Motivations

[9], [2]: Motivations

- Un serveur apprend un modèle global partagé en regroupant des modèles formés localement à partir d'un nombre éventuellement très important de clients.
- Les clients ont généralement des relations déséquilibrées et non i.i.d (indépendant et distribué à l'identique).
- Capacités limitées de transfert de données.

2.1.2 Problèmes

Problème: trouver le w qui minimise la perte moyenne sur les n exemples d'entraînement.
Dans un contexte big data: nombre d'exemples est trop important pour être stocké sur un seul ordinateur.

- Repartir le calcul sur plusieurs ordinateurs
- Le nombre d'exemples de formation détenus par le client $k; n_k = |P_k|$

$f(w) = \sum_{k=1}^{K} \frac{n_k}{n} F_k(w)$ avec $F_k(w) := \frac{1}{n_k} \sum_{i \in P_k} f_i(w)$
K: client sur lesquels les données et les calculs sont distribués
P_k: chaque client détient une partie P_k tous les exemples de formation et calcule $F(w)$: la perte moyenne

du client k.
Problème d'optimisation : $\min_{w \in \mathbb{R}^d} f(w), f(w) := \frac{1}{n} \sum_{i=1}^{n} f_i(w)$
w: vecteur contient d paramètre de modèle Pour l'apprentissage supervisé $f_i(w)$, une fonction de perte (loss function); $f_i(w) = l(x_i; y_i; w)$.
(x_i, y_i): l'un de n exemple de données étiquetés.

Algorithm 1: Apprentissage fédéré FedAvg

Data: Initialisation: w_0
for $t \leftarrow 0$ **to** ... **do**
  $m := \max\{C \cdot K, 1\}$;
  $S_t :=$ ensemble aléatoire de m clients;
  for *chaque* $k \in S_t$ *en parallèle* **do**
    $w_{t+1}^k = \text{ClientUpdate}(k, w_t)$;
  $w_{t+1} = \sum_{k \in S_t} \frac{n_k}{n_\sigma} w_{t+1}^k$, où $n_\sigma = \sum_{k \in S_t} n_k$
Result: w_{t+1}

[9], [2], [8]: FedAvg orchestre la formation via un serveur central qui héberge le modèle global partagé w_t ou t est le cycle de communication. L'optimisation effectuée localement sur les clients (utilisant par exemple, Stochastic Gradient Decent)SGD. FedAvg a cinq hyperparamètres:

1. la fraction de client C à sélectionner pour la formation
2. la taille du mini-lot local B
3. le nombre d'époques locales E
4. un taux d'apprentissage α (learning rate)
5. décroissance du taux d'apprentissage λ

2.2 Federated Stochastic Variance Reduced Gradient FSVRG

L'idée derrière FSVRG [9] est d'effectuer un calcul de coût d'apprentissage de gradient complet de manière centralisée,suivi de nombreuses mises à jour stochastiques distribuées sur chaque client.Le FSVRG standard n'a qu'un seul hyperparamètre : h. Le client k a une taille locale des étapes h_k qui est inversement proportionnelle à n_k, $h_k = \frac{h}{n_k}$

Algorithm 2: Federated Stochastic Variance Reduced Gradient FSVRG

Data: Initialisation: w_0
$h \leftarrow$ stepsize (Taille de pas)
$\{P_k\}_{k=1}^K =$ partition de données
for $t \leftarrow 0$ **to** ... **do**
  $\nabla f(w_t) = \frac{1}{n} \sum_{i=1}^{n} \nabla f_i(w_t)$;
  for *Tous les clients* k *en parallèle* **do**
    initialisation : $w_{t+1}^k \leftarrow w_t$ et $h_k = \frac{h}{n_k}$;
    $\{i_s\}_{s=1}^{n_k}$ permutation de P_k;
    for $s \leftarrow 1$ **to** n_k **do**
      $\Theta \leftarrow \nabla f_{i_s}(w_{t+1}^k) - \nabla f_{i_s}(w_t) + \nabla f_{i_s}(w_t)$;
      $w_{t+1}^k \leftarrow w_{t+1}^k - h_k \Theta$;
  $w_{t+1} = \sum_{k=1}^{K} \frac{n_k}{n} w_{t+1}^k$;
Result: w_{t+1}

2.3 Proposed Framework: FedProx

2.3.1 Motivations

Les différents appareils dans les réseaux fédérés ont souvent des contraintes de ressources différentes en termes de matériel informatique, de connexions réseau et de niveaux de batterie. Par conséquent, il est irréaliste de forcer chaque appareil à effectuer une quantité uniforme de travail (c'est-à-dire exécuter le même nombre d'époques locales,E).

2.3.2 Principe

FedProx [4], est similaire à FedAvg dans le fait qu'un sous-ensemble de périphériques est sélectionné à chaque tour, des mises à jour locales sont effectuées, et ces dernières sont alors moyennées pour former une mise à jour globale. Toutefois, FedProx effectue les modifications simples mais critiques suivantes, qui donnent lieu à des améliorations empiriques importantes et nous permettent également de fournir des garanties de convergence. Dans FedProx :

- Généralisons FedAvg en permettant à des volumes de travail variables d'être effectués localement sur l'ensemble des appareils en fonction de leurs sources de ressources disponibles,
- Agrégant les solutions partielles envoyées par les retardataires(les appareils qui n'appartient pas ou sous ensemble sélectionné d'elle départ pour chaque tour).

En d'autres termes, au lieu d'assumer un γ uniforme pour tous les appareils tout au long du processus d'apprentissage, FedProx accueille implicitement des γ variables pour différents appareils et à différentes itérations.

Définition de γ-inexact solution

Pour une fonction $h(w, w_0) = F(w) + \frac{\mu}{2}\|w - w_0\|^2$ et $\gamma \in [0, 1]$, avec w_* est un γ-inexact solution de $\min_w h(w, w_0)$ si $\|\nabla h(w^*, w_0)\| \leq \gamma \|\nabla h(w_0, w_0)\|$ ou $\nabla h(w, w_0) = \nabla F(w) + \mu(w - w_0)$.

Définition de γ_k^t-inexact solution

γ_k^t-inexact solution, pour l'appareil k à l'itération t, pour une fonction $h_k(w, w_t) = F_k(w) + \frac{\mu}{2}\|w - w_t\|^2$ et $\gamma \in [0, 1]$, avec w_* est un γ_k^t-inexact solution de $\min_w h_k(w, w_t)$ si $\|\nabla h_k(w^*, w_t)\| \leq \gamma_k^t \|\nabla h_k(w_t, w_t)\|$ ou $\nabla h_k(w, w_t) = \nabla F_k(w) + \mu(w - w_t)$.

Algorithm 3: FedProx (Proposed Framework)

Data: Paramètres: $K, T, \mu, \gamma, w_0, N, p_k$, $k = 1, \ldots, N$

for $t = 0$ **to** $T - 1$ **do**

Server sélectionne un sous-ensemble S_t de K appareils de manière aléatoire (chaque appareil k est choisi avec une probabilité p_k);

Le serveur envoie w_t à tous les appareils choisis;

Chaque appareil choisi $k \in S_t$ trouve un w_{t+1}^k qui est un minimiseur inexact de γ_t^k tel que :

$$w_{t+1}^k \approx \arg\min_w h_k(w; w_t) = F_k(w) + \frac{\mu}{2}\|w - w_t\|^2$$

Chaque appareil $k \in S_t$ envoie w_{t+1}^k de nouveau au serveur;

Le serveur agrège les w comme $w_{t+1} = \frac{1}{K}\sum_{k \in S_t} w_{t+1}^k$;

Result: w_{t+1}

3 Comparaissions entre les algorithmes de FL

Voici un tableau qui représente les algorithmes les plus utilisés pour l'apprentissage fédéré:

Référence	Algorithme	Description
[9]	FedAvg (Federated Averaging)	Moyenne des mises à jour des modèles locaux.$w_{t+1} = \sum_{k \in S_t} \frac{n_k}{n_\sigma} w_{t+1}^k$, où $n_\sigma = \sum_{k \in S_t} n_k$
[4]	FedProx (Federated Proximal)	Régularisation proximale γ_k^t-inexact solution pour encourager la similitude entre les modèles locaux et globaux.
[10]	FedOpt (Federated Optimization)	FedOpt vise à résoudre les défis liés à la distribution inégale des données, aux variations de la qualité des connexions et à d'autres problèmes spécifiques à l'apprentissage fédéré. Utilise des techniques d'optimisation avancées. CLIENTOPT et SERVEROPT sont des optimisateurs basés sur des gradients avec des taux d'apprentissage α_l et α respectivement. CLIENTOPT vise à minimiser en fonction des données locales de chaque client tandis que SERVEROPT optimise dans une perspective globale. FedOpt permet naturellement l'utilisation d'optimisateurs adaptatifs (par exemple, ADAM, YOGI, etc.), ainsi que de techniques telles que le dynamique côté serveur (FEDAVGM)
[3]	FedSGD (Federated SGD)	Utilise un algorithme de descente de gradient stochastique pour optimiser les modèles locaux.
[5]	q-FedAvg (Quantized Federated Averaging)	Version quantifiée de FedAvg, où les mises à jour du modèle sont quantifiées avant l'agrégation, q est un paramètre qui ajuste la quantité d'équité que nous voulons imposer.$\min_w f_q(w) = \frac{1}{m} \sum_{k=1}^m \frac{p_k}{q+1} F_k^{q+1}(w)$.
[1]	FedPer (Federated Personalization)	L'objectif principal de FedPer est de permettre aux modèles d'apprentissage d'être personnalisés pour chaque utilisateur tout en conservant la nature décentralisée de l'apprentissage fédéré. Chaque appareil ou client peut personnaliser son modèle en fonction de ses propres données locales. Intègre des mécanismes de personnalisation pour s'adapter aux préférences individuelles des appareils.

Table 2: Algorithmes de Federated Learning

La comparaison entre les algorithmes de Federated Learning se fait au niveau de calcule des fonction de précision (Accuracy), qui mesure la proportion de prédictions correctes par rapport au nombre total d'échantillons, la perte (Loss), la fonction de perte utilisée pendant l'apprentissage peut être évaluée pour mesurer la qualité des prédictions. Pour être la comparaison plus dure et correcte en peut ajouter le temps de convergence, c'est le nombre d'itérations nécessaires pour atteindre une certaine précision ou pour que l'algorithme converge, en plus la communication entre le serveur et les clients, c'est le coût de la communication entre le serveur central et les clients peut être évalué, par exemple, en mesurant la quantité de données échangées ou la fréquence des communications.Pour faire tout les comparaisons nécessaire il faut bien implémenté les différent algorithme et le test sur des donnée réel comme MINIST ou CIFAR-10.

Caractéristique	**FedAvg**	**FedSGD**	**FSVRC**
Optimisation	Minimiser la pert moyenne sur n exemple, utilise une approche basée sur la moyenne des mises à jour des modèles des clients. $\min_{w \in \mathbb{R}^d} f(w), f(w) := \frac{1}{n} \sum_{i=1}^{n} f_i(w)$	Utilise une approche de descente de gradient stochastique fédérée.	Calcule de coût d'apprentissage de gradient complet de manière centralisé, suivi de nombreuses mis à jours stochastiques distribuées sur chaque client.
Communication	Nécessite des communications entre les clients et le serveur central pour partager les mises à jour de modèle.	Implique également des communications entre les clients et le serveur central.	Communications avec les clients pour la mis à jours stochastique distribuées de modèle.
Agrégation	Utilise une moyenne pondérée des mises à jour des clients pour mettre à jour le modèle global.	Agrège généralement les mises à jour locales en utilisant des méthodes de moyenne ou d'autres mécanismes d'agrégation.	Agrégation avec réduction de gradient.
Complexité de l'algorithme	Relativement simple et facile à comprendre, en particulier grâce à son approche de moyenne pondérée.	Peut être plus complexe à mettre en œuvre et à ajuster en raison de la nature stochastique de la descente de gradient.	Introduit de l'hyperparamètre h, ajoutant une complexité.

Table 3: Comparaison entre FedAvg, FedSGD, et FSVRC

References

[1] Manoj Ghuhan Arivazhagan, Vinay Aggarwal, Aaditya Kumar Singh, and Sunav Choudhary. Federated learning with personalization layers.

[2] Peter Kairouz, H. Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Kallista Bonawitz, Zachary Charles, Graham Cormode, Rachel Cummings, Rafael G. L. D'Oliveira, Hubert Eichner, Salim El Rouayheb, David Evans, Josh Gardner, Zachary Garrett, Adrià Gascón, Badih Ghazi, Phillip B. Gibbons, Marco Gruteser, Zaid Harchaoui, Chaoyang He, Lie He, Zhouyuan Huo, Ben Hutchinson, Justin Hsu, Martin Jaggi, Tara Javidi, Gauri Joshi, Mikhail Khodak, Jakub Konečný, Aleksandra Korolova, Farinaz Koushanfar, Sanmi Koyejo, Tancrède Lepoint, Yang Liu, Prateek Mittal, Mehryar Mohri, Richard Nock, Ayfer Özgür, Rasmus Pagh, Mariana Raykova, Hang Qi, Daniel Ramage, Ramesh Raskar, Dawn Song, Weikang Song, Sebastian U. Stich, Ziteng Sun, Ananda Theertha Suresh, Florian Tramèr, Praneeth Vepakomma, Jianyu Wang, Li Xiong, Zheng Xu, Qiang Yang, Felix X. Yu, Han Yu, and Sen Zhao. Advances and open problems in federated learning.

[3] Anran Li, Hongyi Peng, Lan Zhang, Jiahui Huang, Qing Guo, Han Yu, and Yang Liu. FedSDG-FS: Efficient and secure feature selection for vertical federated learning.

[4] Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. Federated optimization in heterogeneous networks.

[5] Tian Li, Maziar Sanjabi, Ahmad Beirami, and Virginia Smith. Fair resource allocation in federated learning.

[6] Wei Yang Bryan Lim, Nguyen Cong Luong, Dinh Thai Hoang, Yutao Jiao, Ying-Chang Liang, Qiang Yang, Dusit Niyato, and Chunyan Miao. Federated learning in mobile edge networks: A comprehensive survey.

[7] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, pages 1273–1282. PMLR. ISSN: 2640-3498.

[8] Jed Mills, Jia Hu, and Geyong Min. Communication-efficient federated learning for wireless edge intelligence in IoT. 7(7):5986–5994.

[9] Adrian Nilsson, Simon Smith, Gregor Ulm, Emil Gustavsson, and Mats Jirstrand. A performance evaluation of federated learning algorithms. In *Proceedings of the Second Workshop on Distributed Infrastructures for Deep Learning*, pages 1–8. ACM.

[10] Sashank Reddi, Zachary Charles, Manzil Zaheer, Zachary Garrett, Keith Rush, Jakub Konečný, Sanjiv Kumar, and H. Brendan McMahan. Adaptive federated optimization.