Sécurité Appliquée, PVP
Devoir maison à rendre avant le 21/01 à 23h45

Jean-François COUCHOT
couchot[arobase]femto-st[point]fr

1 Introduction

Ce sujet contient 3 sections dont la difficulté (indiquée par un nombre d'étoiles) est décroissante. Il est demandé aux étudiant·e·s de se mettre en binôme/trinôme et de choisir un des sujets correspondant à une des sections. Vous devez négocier entre vous le choix des sujets car je ne souhaite qu'il y ait au plus 2 groupes par sujet.

Tout le matériel me permettant d'évaluer votre travail doit m'être retourné par mail avant le 21/01 à 21h45.

2 Apprentissage fédéré(***)

Selon la CNIL [1], « l'apprentissage fédéré est un paradigme d'apprentissage dans lequel plusieurs entités entraînent collaborativement un modèle d'IA sans mise en commun de leurs données respectives. En pratique, les entités impliquées dans l'apprentissage envoient les modèles appris sur leurs données locales à un centre orchestrateur afin de consolider le modèle global. Ce paradigme s'oppose à l'apprentissage centralisé dans lequel toutes les données sont transmises à un serveur central chargé d'exécuter l'apprentissage du modèle ». Cet apprentissage permet de facto de garantir en partie la confidentialité des données. On commencera par déployer une implantation éprouvée, Flower (section 2.1), puis on s'intéressera à une version plus robuste vis à vis de la PVP et qui garantit la DP (section 2.2).

2.1 Apprentissage fédéré avec Flower

Réaliser le tutoriel d'apprentissage fédéré initial [2] avec Pytorch et Flower. On réalisera ceci dans un notebook jupyter que l'on partagera. Rentrer dans les détails du code pour le comprendre et l'expliquer.

2.2 Apprentissage fédéré DP

L'article de référence est récent [YHY+23]. Le travailler, expliquer les points clés. Évaluer l'implantation github [3] citée dans l'article.

3 Gradient Boosting (**)

Le gradient boosting est une technique d'apprentissage automatique où plusieurs modèles faibles, généralement des arbres de décision simples, sont combinés pour former un modèle plus puissant. Il fonctionne en corrigeant les erreurs du modèle existant à chaque étape, en utilisant un processus itératif. L'idée est de construire des modèles successifs qui se concentrent sur les erreurs résiduelles du modèle précédent. Cela améliore la précision globale du modèle. Des bibliothèques populaires comme XGBoost, LightGBM... implantent le gradient boosting.

On commencera par déployer une implantation éprouvée (section 3.1). On poursuivra en étudiant la théorie du gradient boosting (section 3.2). On terminera en étudiant à une version qui garantit la DP (section 3.3).

3.1 Déploiement d'implantations de Gradient Boosting

Réaliser un tutoriel pour prendre en main pratiquement une bibliothèque de Gradient Boosting. On pourra prendre celle de classification de sklearn [4], ou une autre (XGBoost, LightGBM...). Appliquer ceci sur un dataset connu et partager. Expliquer votre code.

1. https://www.cnil.fr/fr/definition/apprentissage-federe
2. https://flower.dev/docs/framework/tutorial-series-get-started-with-flower-pytorch.html
3. https://github.com/BHui97/PrivateFL
4. https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.GradientBoostingClassifier.html

3.2 Théorie du gradient boosting

On creusera la page anglaise wikipedia [5] sur le sujet pour comprendre comment fonctionne le classifieur. On restituera ceci dans un document.

3.3 Gradient boosting DP

On creusera l'article [LWWH20] montrant comment introduire de la DP dans un algorithme de Gradient Boosting. On restituera les points clés notamment. L'implantation [6] sera déployée et évaluée.

4 Regroupement k-moyennes efficace (*)

On a vu en cours que l'algorithme de k-moyennes cherche à à partitionner un ensemble de points en k ensembles en minimisant la distance entre les points à l'intérieur de chaque partition. On a vu une version qui garantit de plus la DP et dont une implantation est présente dans diffprivlib.

Une version plus récente a été présentée dans [BDL+17]. Selon un blog [7] d'une chercheuse chez Google, cette méthode est plus performante en termes de regroupement pour un budget défini que celle implanté dans diffprivlib. On s'intéressera à déployer tout d'abord cette implantation (section 4.1). On se concentrera ensuite sur la théorie de cet algorithme (section 4.2). On déploiera éventuellement une version qui paraît encore plus prometteuse (section 4.3).

4.1 Déploiement de l'implantation k-means d'ICML17

Le code correspondant à l'article est présent sur github [8]. Le déployer et réaliser une série d'expérimentations comme celles du blog présentées ci dessus.

4.2 Théorie de l'article [BDL+17]

Dans cette partie, on creusera l'algorithme DP décrit dans [BDL+17]. On explicitera les points clé et on expliquera pourquoi il vérifie la DP (sans refaire la preuve).

4.3 Implantation encore plus récente

Le blog annoncé ci-dessus propose une implantation d'un k-means vérifiant la DP dont le bruit se fait au moyen d'une fonction de hachage localement sensible [9]. Aucune publication n'est parue à ce sujet. Il s'agira d'évaluer donc pratiquement cette implémentation. A partir de la bibliothèque differential-privacy [10], réaliser un évaluation de cette implantation.

Références

[BDL+17] Maria-Florina Balcan, Travis Dick, Yingyu Liang, Wenlong Mou, and Hongyang Zhang. Differentially private clustering in high-dimensional euclidean spaces. In *International Conference on Machine Learning*, pages 322–331. PMLR, 2017. available at http://proceedings.mlr.press/v70/balcan17a/balcan17a.pdf.

[LWWH20] Qinbin Li, Zhaomin Wu, Zeyi Wen, and Bingsheng He. Privacy-preserving gradient boosting decision trees. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 784–791, 2020. available at https://ojs.aaai.org/index.php/AAAI/article/view/5422/5278.

[YHY+23] Yuchen Yang, Bo Hui, Haolin Yuan, Neil Gong, and Yinzhi Cao. {PrivateFL} : Accurate, differentially private federated learning via personalized data transformation. In *32nd USENIX Security Symposium (USENIX Security 23)*, pages 1595–1612, 2023. available at https://www.usenix.org/system/files/sec23fall-prepub-427-yang-yuchen.pdf.

5. https://en.wikipedia.org/wiki/Gradient_boosting
6. https://github.com/QinbinLi/DPBoost
7. https://blog.research.google/2021/10/practical-differentially-private.html
8. https://github.com/mouwenlong/dp-clustering-icml17
9. https://fr.wikipedia.org/wiki/Locality_sensitive_hashing
10. https://github.com/google/differential-privacy/tree/main/learning/clustering