TP : Découverte de l'Apprentissage Fédéré avec Fluke

Objectifs du TP

Ce TP a pour but de découvrir et manipuler la librairie **Fluke**, un framework dédié à l'apprentissage fédéré (FL). Fluke permet de configurer, exécuter et analyser des scénarios FL de manière flexible et modulaire.
Répo officiel : https://github.com/flukefr/fluke

À la fin du TP, vous serez capables de :

- Installer Fluke dans un environnement Python compatible ;
- Lancer une première expérience FL avec MNIST (scenario IID) ;
- Utiliser les fonctionnalités de log de Fluke puis sauvegarder les métriques globales et locales (par client) ;
- Tracer et analyser les performances des modèles entraînés ;
- Configurer et évaluer un scénario FL non-IID ;
- Appliquer une méthode de mitigation de l'hétérogénéité des données (ex : SCAFFOLD) et analyser ses performances par rapport à FedAVG;
- Tester un autre dataset inclus dans Fluke ;
- Intégrer un nouveau dataset externe et un nouveau modèle ;
- Développer un mini-projet complet d'apprentissage fédéré.

Installation de Fluke

L'objectif de cette section est d'obtenir un environnement FL fonctionnel.

▷ Créer un environnement Python >= **3.10** avec `conda` pour assurer la compatibilité avec Fluke.

▷ Installer Fluke : deux approches sont possibles selon l'objectif.

Installation rapide via PyPI (pour exécuter des expériences)

```
conda create -n fluke310 python=3.10
conda activate fluke310
pip install fluke-fl                          % installation du framework
```

Installation en mode développement (pour modifier Fluke)

```
conda create -n fluke310 python=3.10
conda activate fluke310
git clone https://github.com/flukefr/fluke.git
cd fluke
pip install -e .                  % installation editable (mode développement)
```

▷ Une fois l'installation terminée, vérifier que Fluke est accessible via la ligne de commande en exécutant :

```
fluke --help
```

Premier entraînement fédéré avec MNIST (scenario IID)

Cette section vous fera exécuter votre première expérience FL.
À faire :

▷ charger la config MNIST fourni par Fluke ;

```
fluke-get config exp
fluke-get config fedavg
```

Cette commande crée automatiquement un dossier `config/` contenant au moins deux fichiers YAML :

- `exp.yaml` — configuration de l'expérience MNIST
- `fedavg.yaml` — configuration de l'algorithme FedAVG

▷ lancer un entraînement FL avec l'algorithme FedAVG ;

```
fluke federation config/exp.yaml config/fedavg.yaml
```

▷ visualiser l'évolution des performances au fil des rondes d'entraînement ;

Ensuite :

▷ lancer une exécution **centralisée** (entraînement classique non-fédéré) via Fluke ;

```
fluke centralized config/exp.yaml config/fedavg.yaml
```

▷ lancer une exécution **d'apprentissage fédéré décentralisé** ;

```
fluke clients-only config/exp.yaml config/fedavg.yaml
```

▷ comparer les courbes et discuter les différences entre FL, centralisé et client-only.

Sauvegarder et analyser les traces d'entraînement

Fluke fournit un logger configurable via la section `logger` dans le fichier de configuration d'expérience. Les loggers standards sont implémentés dans le module `fluke/utils/log.py`.
Les principales classes de logger sont :

- `Log` : logger de base. Il affiche dans la console les performances du modèle global et les informations de communication pendant l'entraînement fédéré.
- `DebugLog` : variante plus verbeuse de `Log`, utile pour le débogage.
- `WandBLog` : envoie les métriques vers Weights & Biases.
- `TensorboardLog` : envoie les métriques vers TensorBoard.
- `ClearMLLog` : logger pour ClearML, qui s'appuie sur le logger TensorBoard.

La configuration minimale du logger dans le fichier d'expérience ressemble à :

```
logger:
    name: Log
```

Pour plus de détails sur les loggers et leurs options, voir la documentation officielle de Fluke (section *fluke.utils.log*) : https://makgyver.github.io/fluke/fluke.utils.log.html.

À faire :

▷ modifier la seuvegarde des métrique pour générer deux fichiers CSV : un pour les métriques globales, un pour celles des clients ;

▷ tracer l'accuracy globale au fil des rounds ;

▷ tracer l'accuracy par client et comparer les trajectoires.

Contexte de données non-IID

L'objectif ici est de tester un scénario non-IID afin d'observer l'impact d'une répartition hétérogène des données entre les clients dans un apprentissage fédéré. Fluke permet de choisir différents schémas de distribution des données via le paramètre `distribution` dans le fichier d'expérience. La liste des distributions prises en charge (IID, Dirichlet, etc.) et leurs paramètres est décrite dans la documentation officielle : https://makgyver.github.io/fluke/fluke.data.html.

À faire :

▷ la documentation décrit les distributions possibles, l'une d'elles est la distribution `dirichlet` paramétrée par un coefficient `beta`.

▷ remplacer la distribution IID par :

```
distribution:
    name: dir
    beta: 0.02
```

- ▷ entraîner, sauvegarder les traces (globales et locales) et tracer les métriques ;
- ▷ comparer avec le scénario IID.

Méthode de mitigation de l'hétérogénéité : SCAFFOLD

Fluke inclut plusieurs algorithmes d'apprentissage fédéré pouvant traiter l'hétérogénéité des données. Parmi eux : **SCAFFOLD**. La liste complète des algorithmes FL disponibles (FedAVG, FedProx, SCAFFOLD, etc.) est consultable dans la documentation officielle : https://makgyver.github.io/fluke/fluke.algorithms.html.

À faire :

- ▷ récupérer la config de SCAFFOLD ;

```
fluke get scaffold
```

- ▷ relancer l'expérience en non-IID avec SCAFFOLD au lieu de FedAVG;
- ▷ tracer les performances et comparer à FedAVG non-IID.

Question de réflexion :

- SCAFFOLD améliore-t-il l'apprentissage dans votre configuration ? Pourquoi ?

Tester un autre dataset inclus dans Fluke

Fluke intègre plusieurs datasets utilisables directement (images, texte, capteurs, tabulaires, etc.). La liste complète des datasets inclus dans Fluke, ainsi que leurs caractéristiques et options de préchargement, est disponible dans la documentation officielle : https://makgyver.github.io/fluke/fluke.data.datasets.html.

À faire :

- ▷ choisir un dataset inclus (par ex. CIFAR-10, Shakespeare, FEMNIST...) ;
- ▷ adapter le modèle si nécessaire (CNN pour CIFAR-10, LSTM pour Shakespeare...) ;
- ▷ exécuter un scénario FL ;
- ▷ tracer les résultats.

Intégration d'un nouveau dataset et d'un nouveau modèle

Cette partie du TP consiste à étendre Fluke en allant au-delà des datasets et modèles déjà inclus dans la bibliothèque. L'objectif est d'importer un dataset externe, de définir un modèle de votre choix (CNN, RNN, Transformer, MLP, etc.) et de rendre l'ensemble compatible avec l'entraînement fédéré en construisant un `DataContainer` conforme aux spécifications de Fluke.

Indication :

- ▷ le tutoriel officiel pour ajouter un dataset dans Fluke se trouve ici : https://makgyver.github.io/fluke/examples/tutorials/fluke$_c ustom_d ataset.html$
- ▷ le tutoriel pour ajouter un nouveau modèle est disponible ici : https://makgyver.github.io/fluke/examples/tutorials/fluke$_c ustom_n n.html$
- ▷ la structure et les champs d'un `DataContainer` sont illustrés dans les exemples fournis dans la documentation.
- ▷ une fois le dataset et le modèle intégrés dans Fluke, développer un **scénario fédéré complet** : définir la distribution des données entre les clients, choisir un algorithme d'agrégation, lancer la fédération et analyser les performances globales et locales.