Implémentation PyTorch + PyTorch Lightning d'une librairie de Machine Learning inspirée de Scikit-Learn

PFE M2 IIS

Responsable: Pierre Jacob (pierre.jacob@u-bordeaux.fr)

Mot-clés: génie logiciel, programmation Python, PyTorch, apprentissage automatique.

1 Introduction

Scikit-learn (`https://scikit-learn.org/stable/`) est une bibliothèque open-source développée en Python regroupant des outils simples de l'apprentissage statistique et automatique. On retrouve les outils classiques de l'IA sur une large variété de tâches, telles que la classification (machines à vecteurs supports, k plus proches voisins, random forest, LDA/QDA, ...), la régression (moindre carré, Lasso, ...), réduction de dimension (PCA, LLE, ...), ainsi que des méthodes génériques comme les méthode d'ensembles (bagging, forests, boosting), de sélection de features, calibration, de visualisation, *etc.*

Scikit-learn ne nécessite que peu de dépendances (seules les librairies NumPy, SciPy et matplotlib sont requires) afin de de la rendre accessible et réutilisable dans une grande variété de contexte et d'applications.

Son principal défaut cependant repose sur le fait qu'elle n'est réellement utilisable que sur dees architectures CPUs, du fait de sa construction autour des librairies SciPy et NumPy. L'objectif de ce projet est de re-développer une librairie d'algorithme de machine learning inspirée de Scikit-learn, mais en remplaçant les frameworks NumPy et SciPy par **PyTorch** (`https://pytorch.org/`) et **PyTorch Lightning** (`https://www.pytorchlightning.ai/`).

2 PyTorch

PyTorch est un framework optimisé pour le prototypage d'architecture de machine learning. Historiquement, il a principalement été optimisé pour supporter le calcul GPU autour d'outils de différentiation automatique afin de profiter des gains de performance par rapport à des architectures CPUs pour de l'entraînement de modèles de deep learning, tels que les réseaux de neurones. En développant une librairie de machine learning classique basée sur PyTorch, il sera possible de bénéficier de l'environnement de gestion natif de l'architecture

hardware (même code que l'on l'exécute en CPU ou GPU), des outils inhérents au machine/deep learning (gestion de données basée datasets, meilleure modularité que Scikit-learn, doc fournie, large communauté, framework supporté par Facebook), tout en étant capable d'exploiter des algorithmes classiques du machine learning, qui restent très compétitifs (autant computationnellement qu'en terme de performances pures) dans une large variété d'applications et de tâches (ex: quasi-totalité des données tabulées, quelques tâches en images).

3 PyTorch Lightning

PyTorch Lightning est un framework complémentaire à PyTorch, pensé pour optimiser l'environnement autour de PyTorch. Typiquement, PyTorch Lightning permet de :

- Gérer automatiquement la répartition du code ou des données dans un environnement multi-GPU ou hybride
- Profiler l'applications en analysant les performances et les goulots d'étranglement (accès à la données, temps de calculs des modèles, ...)
- Gérer quasi-automatiquement la reprise de l'entraînement de modèles (checkpoint)
- D'utiliser des modèles avec des précisions réduites, voire quantifiées pour accélérer le temps de traitement
- Déployer son code sur des environnements distribués
- Gérer l'environnement autour du modèle et de son entraînement (gestion des logs, métriques, visualisation, early stopping, ...)
- Gérer efficacement le cycle de vie d'un modèle de machine learning (pipeline des données, entraînement et sélection de modèles, *etc.*)

4 Objectifs

Au terme du projet, le groupe d'étudiant devra:

- Développer une librairie Python inspirée des APIs de Scikit-learn et PyTorch, tout en exploitant le framework de ce dernier,
- Fournir des Notebook Jupyter exploitant la librairie développée, par exemple dans le cadre de compétitions Kaggle (`https://www.kaggle.com/competitions`) sur des données au choix (tabulées, images, son, texte, *etc.*) pour une tâche libre (classification, régression, visualisation, ...)
- Réaliser des benchmarks des algorithmes proposés (comparaison en performance entre Scikit-learn, l'implémentation qui tourne sur CPU et l'implémentation qui tourne sur GPU)