Projet 2: Reconfiguration en temps réel d'un réseau sans fils

A rendre avant le 20 décembre 2023 minuit

December 1, 2023

1 L'objectif

L'objectif de ce projet est d'implémenter des algorithmes d'apprentissage par renforcement sur des données en temps réel pour simuler une intéraction directe entre un agent et un environnement. Dans ce projet on vous propose de travailler sur des réseaux sans fils pour mesurer la qualité des transmissions entre un terminal et une antennes. Vous avez le choix de choisir un projet parmi deux:

- Projet envoyé via teams
- https://github.com/stefanbschneider/mobile-env.git

Les deux projets ont pour but d'analyser l'état du lien et converger vers un état optimal qui permet des maximiser le débit des données envoyées par les terminaux.

2 Les livrables

Vous devez produire du code Python, structuré en plusieurs fichiers, permettant d'exécuter un dashboard dans un navigateur standard. Le dashboard contiendra a minima 3 figures (pour 3 scénarios: exemple: 4 antennes vs 500 terminaux) avec 5 algorithmes d'apprentissage par renforcement (ANN, Policy Iteration, Thompson, EXP3, UCB). Le/Les réseaux de neurones doivent contenir au moins 4 neurones en entrée, 4 neurones cachés, et 1 ou plusieurs neurones en sortie. Vous n'avez pas le droit d'importer des librairies qui executent ces algorithmes, vous devez les coder par vous même.

Le code devra recueillir et nettoyer les données, les sélectionner et les organiser pour les représentations graphiques interactives illustrant votre point de vue sur le sujet.

3 Les données

Les données doivent être générées en temps réel par le simulateur. Elles doivent contenir un ensemble des parametres de configuration (BW, puissance de transmission, CR, etc) et un ensemble de métrique (débit (DR), ToA, SNR, RSSI).

4 Contexte du développement

Le dashboard sera développé avec les versions les plus récentes disponibles au 1er septembre de l'année universitaire :

- du langage Python ;
- de pandas, dash et plotly ;
- et plus généralement des autres packages utilisés dans le mini projet.

C'est dans ces conditions que votre travail sera évalué.

5 Remise du travail

Gardez à l'esprit que votre travail sera déployé dans un contexte différent de celui de votre machine de développement.

Le dépôt Git doit être public et contenir :

- l'ensemble du code strictement nécessaire pour exécuter le projet ;
- un fichier main.py à la racine du dépôt tel que l'exécution de ce seul fichier construit le dashboard ;
- le fichier README formatté en Markdown renseigné avec :
  - un "User Guide" qui permet de déployer et d'utiliser votre dashboard sur une autre machine ;
  - un rapport d'analyse qui met en avant les principales conclusions extraites des données ;
  - un "Developper Guide" qui permet de comprendre l'architecture du code et de modifier ou d'étendre celui ci.

Le fichier requirements.txt doit contenir la liste des seuls packages additionnels requis tel que la commande qui installe l'ensemble des dépendances nécessaires (et uniquement celles là).

> python -m pip install -r requirements.txt

Après clonage du dépôt et à l'exécution de main.py :

- le code ne doit pas produire de warnings dans la console ;
- le dashboard ne doit pas comporter pas d'erreur de call back.

6 Déploiement du travail

Le projet sera copié sur la machine d'évaluation avec la commande:
> git clone adresse_publique_de_votre_projet
Le dashboard sera lancé avec la commande:
> python main.py
Il sera prudent de reproduire les 2 étapes ci dessus sur une autre machine avant de soumettre votre travail.

7 Evaluation du travail

Le dashboard doit être fonctionnel à la première tentative. Chaque interaction supplémentaire entre l'évaluateur et le binôme sera assortie d'un point de pénalité. Les causes les plus fréquentes

- référence à un fichier local à la machine de développement ;
- utilisation d'un package obsolète ;
- utilisation d'un package non recensé dans requirements.txt.

Votre code doit être structuré en fonctions, classes, et modules.
Il doit être documenté :

- par des noms de variables explicites
- des commentaires précisant pourquoi on exécute les principales instructions
- des docstrings, voire du typing pour les fonctions/classes utilisées
- Les structures de données doivent être adaptées.
- Les "bonnes pratiques" du langage seront valorisées.

8 Grille évaluation

Ci dessous, une liste non limitative des critères pouvant être utilisés pour l'évaluation. Ces critères ne sont pas d'égale importance.

- Le dépôt
  - montre une utilisation régulière de git ;
  - le README est renseigné ;
  - contient les principales conclusions de l'étude ;
  - contient des informations sur les données utilisées ;
  - contient un User Guide correctement renseigné ;

  - contient un Developer Guide correctement renseigné.

- Le dashboard
  - l'instruction pour le produire est présenté et il se lance sans erreur ;
  - contient les représentations graphiques demandées ;
  - les graphiques sont correctement documentés (titre, label des axes, etc.) ;
  - est dynamique et fluide ;
  - est de façon générale de bonne qualité.

- Le code
  - est structuré en modules, classes, fonctions ;
  - est lisible et commenté ;
  - met en œuvre les bonnes pratiques du langage ;
  - les structures de données et les modules sont adaptés

- Les données
  - les données sont générées en temps réel par le simulateur.

A chaque item, un grade est attribué:

- 4 : tout à fait d'accord
- 3 : plutôt d'accord
- 2 : ni vraiment d'accord, ni vraiment en désaccord
- 1 : plutôt en désaccord
- 0 : tout à fait en désaccord

Ces critères garantissent un projet complet, structuré et développé selon les bonnes pratiques.