TP #2 – ESTIMATEUR DE SIMILARITES

8PRO408 – Outils de programmation pour la science des données

Dans le cadre du cours, nous avons vu l'utilisation de NumPy, de Pandas, de Jupyter. Afin d'appliquer les concepts vus en cours et d'avoir un point de vue pratique dans le cours, il vous est demandé de transformer une phrase en représentation vectorielle et de trouver l'élément qui représente le mieux cette représentation.

Pour ce faire, vous devez utiliser ce jeu de données : https://dl.fbaipublicfiles.com/fasttext/vectors-crawl/cc.fr.300.vec.gz **(ATTENTION : FICHIER DE PLUSIEURS GIGA-OCTETS),** qui représente chaque mot avec sa position vectorielle dans un espace à 300 dimensions. Bien que le concept puisse paraitre abstrait, on peut utiliser le jeu de données avec chaque mot étant associé à un tableau de taille 300. Ce fichier est au format « CSV », mais le délimiteur n'est pas une virgule, mais un espace. La première colonne correspond au mot, et les 300 suivantes aux coordonnées.

Exemple :

```
deux -0.0227 0.1099 -0.0036 0.0586 … -0.0041 0.0753 0.0509 0.0709
faire 0.0476 -0.0587 -0.0058 0.0405 … -0.0208 0.1458 0.0167 0.0408
```

Ce TP doit être réalisé seul.

Travail à effectuer

1. En négligeant le poids en octet du mot, déterminer le nombre d'octets moyen pour stocker la représentation vectorielle en 300 dimensions d'un seul mot.
2. Télécharger le jeu de données et créer une fonction avec les conditions suivantes :
   - Prend en paramètre un mot à chercher dans le jeu de données
   - Parcoure le jeu de données et retourne la représentation vectorielle de ce mot, ou « None » le cas échéant.
   - Ne doit jamais prendre en mémoire plus que 100Mo pour la gestion des données du jeu de données. Pour ce faire, vous pouvez prendre votre calcul défini en 1. et retrouver le nombre de ligne « maximal » à récupérer à la fois.
3. Faire un programme prenant dans la toute première cellule du notebook une phrase, par exemple « Je suis tombé du toit de l'usine ».

- Celui-ci devra lire la phrase et prendre tous les éléments de la phrase de manière séparée (mots, signes de ponctuation, etc.).
- Pour chaque élément, utiliser la fonction créée en 2. pour retrouver la position vectorielle de chaque mot.
- Faire la moyenne de ces éléments.

4. Implémentez une fonction pour calculer la distance cosinus entre deux éléments avec l'équation suivante :
   - $\frac{A \cdot B}{\|A\| \cdot \|B\|}$ sachant que vous pouvez utiliser les fonctions de Numpy DOT et NORM.
5. *Utilisez la fonction de distance développée pour comparer chaque mot du jeu de données avec celui-ci. Retourner le « mot » dont la représentation vectorielle est la plus proche de celle retrouvée en 3.*
6. ***BONUS : 20 points bonus peuvent être accordés pour le défi suivant :***
   - Le programme doit gérer une phrase avec une extension, par exemple « FR: Je suis tombé » ou « EN: I fell from the roof ». (PREFIX: PHRASE).
   - Le préfixe détermine la langue dans laquelle il faut récupérer et comparer les représentations vectorielles. Les tests ne seront effectués qu'avec anglais et francais (EN / FR).
     - FR : https://dl.fbaipublicfiles.com/fasttext/vectors-crawl/cc.fr.300.vec.gz
     - EN : https://dl.fbaipublicfiles.com/fasttext/vectors-crawl/cc.en.300.vec.gz
   - Si le texte est envoyé sans préfixe, il faut assumer que le préfixe est FR.

Format de rendu

Le projet rendu DOIT respecter les consignes suivantes :

- Fonctionner avec la dernière version de Python 3 (3.12.3 à l'heure actuelle)
- Fonctionner avec les librairies NumPy et Pandas (elles seront les seules installées sur l'environnement de test)
- Votre rendu doit être un unique fichier Jupyter Notebook, correctement rédigé avec des informations sur vos tests, sur la manière de générer vos données, etc.
- Celui-ci doit être capable de gérer des architectures comme celle-ci :
  - Votre fichier notebook
  - Le dossier « data »
    - « vector-en.txt » (représentant le dataset demandé en anglais)
    - « vector-fr.txt » (représentant le dataset demandé en français)
- Votre code sera lu par le correcteur au format Notebook
- Le fichier Notebook doit être nommé de la manière suivante : « NOM_PRENOM.ipynb »

Tout non-respect de ces consignes de rendu peut se voir sanctionné par des pénalités.

Date de rendu

Le projet devra être remis avec **un seul fichier .zip au plus tard le dimanche 16 juin 2023 à 23h59** contenant UNIQUEMENT le fichier .ipynb du notebook. Le rendu se fera sur Moodle via l'option créée à cet effet.

ATTENTION : Aucun retard n'est toléré. Si vous n'avez pas rendu le fichier zip, vous aurez automatiquement la note 0.

Répartition de la notation

- 5% : Nomenclature respectée pour l'ensemble des fichiers (donnés et générés)
- 5% : Le code fonctionne directement sans intervention annexe (avec une phrase d'exemple)
- 30% : La fonction du 2. retourne bien la représentation vectorielle d'un mot
- 25% : La fonction du 3. retourne bien la représentation vectorielle d'une phrase.
- 25% : La fonction du 4. retourne bien la bonne distance cosinus entre deux vecteurs
- 10% : Programme complet respecté.
- *Le bonus de 20% sera accordé sur un autre TP si l'étudiant a déjà la note maximale à celui-ci.*