Fiche Documentaire du Code : *Collecte_data*

Titre du projet :

Analyse personnalisée des articles scientifiques (à l'aide des LLM et d'un paradigme RAG)

Auteurs :

- **Bachou Brahim** : Responsable de la collecte des données et de l'automatisation du téléchargement des articles scientifiques.

Description du code :

Ce code a pour objectif de collecter des articles scientifiques depuis la plateforme *arXiv* et de les traiter pour extraire leur contenu. Il utilise l'API de *Google Drive* pour automatiser le téléchargement des fichiers au format texte, puis stocke ces articles dans un fichier Excel structuré. Le code extrait les titres, auteurs, résumés et contenus des PDF des articles récupérés, et permet de les enregistrer sous forme de fichiers texte.

Structure du code :

- **Collecte_data.ipynb** : Ce notebook est le point d'entrée du processus de collecte et de traitement des articles scientifiques. Il contient toutes les étapes nécessaires pour installer les bibliothèques, configurer l'environnement, interagir avec l'API Google Drive et exécuter le téléchargement et le traitement des fichiers.

Structure des fichiers générés :

- **dossier_Echantillons/** : Répertoire contenant les fichiers texte de chaque article récupéré.
- **arxiv_data.xlsx** : Fichier Excel contenant les informations collectées sur chaque article (titre, auteurs, lien, contenu).

Instructions d'installation :

1. **Installer les bibliothèques nécessaires :**
    - Les bibliothèques requises sont installées via pip, notamment selenium, requests, PyPDF2 et PyMuPDF.

```
pip install selenium requests PyPDF2 PyMuPDF
```

2. **Configurer l'environnement :**

- Vous devez monter Google Drive dans l'environnement Colab pour accéder aux fichiers.
- Ajouter les clés d'API pour utiliser l'API Google Drive.

Prérequis :

- **Python 3.10**
- **Google Colab** pour exécuter le code.
- **Clé API Google Drive** pour interagir avec Google Drive et télécharger les fichiers.

Fonctionnement du code :

1. **Installation des dépendances** : Le code commence par installer les bibliothèques nécessaires, notamment selenium, PyPDF2 et PyMuPDF, pour l'extraction de contenu des fichiers PDF.
2. **Automatisation du téléchargement des articles** :
   - Le module *Selenium* est utilisé pour automatiser la navigation et la collecte des articles sur la plateforme *arXiv*.
   - Les titres, auteurs, résumés et liens vers les PDF sont extraits et sauvegardés.
3. **Extraction de contenu des PDF** :
   - La fonction extract_pdf_content_from_url() télécharge et extrait le contenu textuel de chaque article au format PDF, en utilisant *PyMuPDF*.
4. **Enregistrement des données** :
   - Les informations sur chaque article sont stockées dans un fichier Excel et les contenus sont sauvegardés dans des fichiers texte individuels dans un répertoire.

Entrées et sorties :

Entrées :

- URL des articles scientifiques récupérés sur *arXiv*.
- Fichiers PDF des articles scientifiques.

Sorties :

- Fichiers texte contenant le contenu des articles scientifiques.
- Fichier Excel structuré contenant les informations des articles : titre, auteurs, lien, résumé, lien PDF.

Exemples d’utilisation :

Pour exécuter le code et commencer la collecte des articles, il suffit d'exécuter les cellules du notebook *Collecte_data.ipynb*. Le processus automatisera la collecte, le téléchargement des PDF et l'extraction du contenu :

```
driver = web_driver()
driver.get('https://arxiv.org/search/?searchtype=all&query=ai&abstracts=show&size=200&order=-announced_date_first&start=0')
```

Tests et validation :

Le code a été validé en testant l'extraction de plusieurs articles et leur stockage dans des fichiers texte et Excel. Le contenu des PDF a été correctement extrait, même pour des fichiers volumineux.

Problèmes connus :

- Certains fichiers PDF peuvent contenir des erreurs de syntaxe liées à leur format, ce qui peut limiter l'extraction correcte du texte pour ces documents spécifiques.