INFO-F105 – Langages de programmation I
Projet - Simulation d'une Intelligence Artificielle Générative
Phase I

Youcef Bouharaoua
youcef.bouharaoua@ulb.be

Année académique 2023-2024

Introduction

Dans le vaste monde culinaire de l'intelligence artificielle générative, nous avons la chance de jouer avec des ingrédients qui peuvent transformer des idées en délices numériques. Comme un chef cuisinier explore de nouvelles recettes pour éblouir les papilles.

L' **IA générative** nous permet de créer du contenu innovant : des textes captivants, des images à couper le souffle, et bien plus encore.
Parmi les chefs de notre cuisine numérique, ChatGPT brille (ou pas) par son talent, capable de concocter des plats textuels avec une aisance déconcertante. À la base de cette prouesse se trouve un modèle sophistiqué connu sous le nom de GPT (Generative Pre-trained Transformer), une sorte de livre de recettes secret qui guide ChatGPT dans la préparation de ses créations.

C'est d'ailleurs ChatGPT qui nous a suggéré cette métaphore culinaire.[1]

Mais avant de pouvoir régaler nos invités avec des plats complexes, tout chef digne de ce nom sait qu'il doit d'abord maîtriser l'art de préparer et d'organiser ses ingrédients. C'est là que commence notre aventure : un voyage à travers la sélection, la préparation et l'organisation méticuleuse de données variées pour nourrir notre modèle que nous appellerons TPG. Ce projet vous invite à enfiler votre tablier, à aiguiser vos couteaux de programmation et à plonger dans la cuisine de l'**IA générative**, où votre mission sera d'organiser un festin numérique en utilisant Python et C++ comme vos ustensiles de prédilection.

[1] via le prompt : *propose moi une métaphore afin d'expliquer d'une manière marante et intuitive l'IA générative.*

Préparez-vous à explorer le processus derrière la création d'un LLM (Large Language Model), en commençant par la tâche fondamentale: **la collecte et l'organisation des données**.

Détails de soumission et critères d'évaluation

Durant cette première phase du projet, les notions suivantes sont abordées :

- Les variables et la gestion de la mémoire.
- Les instructions basiques en C++ dont les conditionnelles et les boucles.
- Les types de données dont les pointeurs et les tableaux.
- Les fonctions dont la transmission par valeur et référence.
- Les distinctions entre C++ et Python, en particulier concernant les opérations d'entrée et de sortie (E/S).

Il n'est pas nécessaire que vous maîtrisiez ces concepts dans les détails pour pouvoir faire ce projet. L'idée est de vous y familiariser de manière pratique. Ne vous inquiétez pas si le travail que vous rendez ne fonctionne pas parfaitement (tant que vous rendez votre travail), aucune pénalité ne sera appliquée. Pour rappel, votre travail pour la première phase doit être rendu au plus tard le 18 mars 2024 à 23h59. La remise est constitué d'un zip portant comme nom *phase1.zip* et contenant : *phase1.cpp* et *phase1.py*. En cas de non remise à temps, vous ne serez pas autorisés à rendre votre travail pour la deuxième et troisième phase.

Le Projet

Objectif

Dans notre cuisine numérique, votre mission en tant que chef développeur est de concocter un système efficace pour traiter un assortiment diversifié de données. À l'aide des outils de programmation Python et C++, vous allez préparer les données — nos ingrédients — en les décompressant, classifiant et organisant soigneusement dans des répertoires structurés, tout comme un chef range ses ingrédients par catégorie pour faciliter la préparation de ses plats.

Cette première étape est cruciale : elle pose les bases nécessaires au développement d'applications d'intelligence artificielle, telles que les modèles de langage génératifs. En traitant et en organisant les données, vous apprendrez non seulement à manipuler des fichiers et des dossiers à travers des scripts automatisés, mais aussi à aborder des problématiques de confidentialité et d'accès aux données, un aspect fondamental dans le domaine de l'IA.

Contexte

Vous recevez un fichier ZIP contenant une collection mixte de fichiers dans différents formats (documents HTML, fichiers texte (.txt), fichiers audio (.mp3, .wav), images (.jpg, .png), et vidéos (.mp4, .avi)). Votre tâche consistera à développer un programme qui, en deux parties, traitera ces données :

1. La première partie, réalisée avec Python, se concentrera sur la décompression de l'archive, l'identification du type de chaque fichier et leur organisation dans des dossiers appropriés.

2. La deuxième partie, mise en œuvre avec C++, approfondira l'analyse en affichant un rapport détaillé sur le contenu de ces dossiers, incluant une vérification spécifique des fichiers texte pour identifier d'éventuels problèmes de confidentialité liés à leur licence d'utilisation.

L'objectif est de vous familiariser avec des opérations fondamentales de gestion de fichiers et de répertoires tout en abordant des considérations pratiques et éthiques importantes dans le traitement des données.

Partie 1 : Décompression et classification des données avec Python

Tâches

1. **Décompression de l'archive :** Écrivez une fonction Python nommée `decompress_archive` qui prend en entrée le chemin de l'archive ZIP et le répertoire de destination pour les fichiers extraits. Afin de correctement réaliser cette tâche, vous devez utiliser le module *zipfile*.

2. **Identification et organisation des fichiers :** Implémentez une fonction Python `organize_files` pour classifier les fichiers selon leur type (HTML, Textes, audio, images), le type du fichier est déterminé via son extension uniquement. La fonction devra ensuite ranger les fichiers dans des dossiers correspondants. Voici une liste exhaustive des extensions qui doit être gérées:

   - textuelles (html) : *.html*.
   - textuelles (textes) : *.txt*.
   - binaires (audio) : *.mp3*, *.wav*.
   - binaires (images) : *.jpg*, *.png*.

Livraisons

- Un script Python accomplissant les tâches de décompression et d'organisation des fichiers (*phase1.py*).

Listing 1: Signatures des fonctions Python

```
 import zipfile

def decompress_archive(zip_path: str, extract_to: str) -> None:
    """
    Decompresse l'archive ZIP specifiee dans le chemin zip_path
        vers le repertoire extract_to.
    :param zip_path: Chemin vers le fichier ZIP a decompresser
    :param extract_to: Repertoire de destination pour les fichiers
        extraits.
    """
    # Votre code ici

def organize_files(directory: str) -> None:
    """
    Organise les fichiers dans le repertoire specifie en les
        classant dans des sous-dossiers selon leur type.
    :param directory: Chemin du repertoire contenant les fichiers a
         organiser.
    """
    # Votre code ici
```

Listing 2: Structure du repertoire avant l'execution de organize_files

```
/mon_dossier
    document1.txt
    image1.jpg
    musique1.mp3
    script1.html
```

Listing 3: Structure du repertoire après l'execution de organize_files

```
/mon_dossier
    /Textes
        document1.txt
    /Images
        image1.jpg
    /Audio
        musique1.mp3
    /HTML
        script1.html
```

Partie 2 : Traitement des données et rapport avec C++

Tâches

1. **Analyse des dossiers :** Créez une fonction C++ `analyze_folders` pour parcourir les répertoires organisés et identifier le nombre de fichiers par catégorie.

2. **Vérification des problèmes de confidentialité :** Intégrez dans l'analyse une vérification des fichiers texte (qui ont comme extension *.html* ou *.txt* pour s'assurer qu'ils contiennent la mention "Licence: free access" au début, signalant leur utilisation comme libre.

Vous devez utiliser la librairie <*filesystem*> afin d'effectuer les opérations nécessaires à l'accomplissement des tâches ci-dessus (parcours d'un répertoire, opérations sur fichiers,...etc.).

Livraisons

- Le code source C++ pour l'analyse des données et la vérification des licences (*phase1.cpp*).

Listing 4: Signatures des fonctions C++

```
#include <string>
#include <filesystem>

namespace fs = std::filesystem;

void analyze_folders(const std::string& directory_path) {
    /**
     * Parcourt les repertoires organises pour identifier le nombre
          de fichiers par categorie
     * ainsi que leur taille totale. Inclut une verification pour
         les fichiers texte contenant
     * la mention "Licence: free access" au debut du fichier.
     * Affiche un rapport detaille sur le contenu des dossiers
         analyses, incluant le nombre de fichiers
     * par categorie et les resultats de la verification de
         confidentialite.
     * @param directory_path Chemin du repertoire contenant les
         fichiers organises.
     */
}
```

Listing 5: Affichage du rapport d'Analyse

```
    Resume par Categorie

    Textes:
    - Nombre de fichiers: 15
    - Fichiers avec problemes de confidentialite: 2

    HTML:
    - Nombre de fichiers: 8
    - Fichiers avec problemes de confidentialite: 0

    Audio:
    - Nombre de fichiers: 20
    - Fichiers avec problemes de confidentialite: N/A

    Images:
    - Nombre de fichiers: 30
    - Fichiers avec problemes de confidentialite: N/A

    Videos:
    - Nombre de fichiers: 5
    - Fichiers avec problemes de confidentialite: N/A
```