Conception et Implémentation d'un Système de Questions-Réponses basé sur l'Architecture RAG : Cas de l'IFRI

Arix Alimagnidokpo[1], Obed Eguedji[1], Rosas Behoundja[1], Gédéon Guedje[1]

Superviseur : Dr Ing. (MC) Vinasétan Ratheil Esse Houndji[1]

[1]Institut de Formation et de Recherche en Informatique (IFRI), Université d'Abomey-Calavi, Bénin

Résumé

La Génération Augmentée par Récupération (RAG) combine recherche sémantique et génération de texte pour produire des réponses fondées sur des documents. Ce projet consiste à concevoir et déployer un agent conversationnel destiné à répondre aux questions liées à l'IFRI en s'appuyant uniquement sur les documents officiels de l'institution. L'implémentation repose sur ChromaDB pour le stockage vectoriel, sur le modèle all-MiniLM-L12-v2 pour l'embedding, et sur Gemini 1.5 Pro pour la génération. Le système montre comment une architecture RAG peut faciliter l'accès aux connaissances institutionnelles dans le contexte universitaire béninois.

Introduction

Dans de nombreuses universités africaines, les informations académiques et administratives sont éparpillées entre PDF, site web et archives papier, rendant leur consultation lente et complexe.
L'objectif du projet est de développer un système capable d'extraire automatiquement les informations pertinentes et de les restituer sous forme conversationnelle en langage naturel.
Le travail s'appuie sur plusieurs documents officiels de l'IFRI : Arrêté d'Ouverture de Formation, conditions de délivrance des actes académiques, offres de formation Licence et Master, règlement intérieur, ainsi qu'une extraction du site web institutionnel.

Collecte et Traitement des Données

Le corpus est composé de PDF officiels fournis par l'administration. Le texte extrait est segmenté en blocs de 1600 caractères avec un chevauchement de 200 caractères pour préserver le contexte.
Chaque segment est transformé en vecteur à l'aide du modèle HuggingFace all-MiniLM-L12-v2. Au total, 847 segments ont été générés puis stockés dans ChromaDB, permettant des recherches rapides par similarité sémantique.

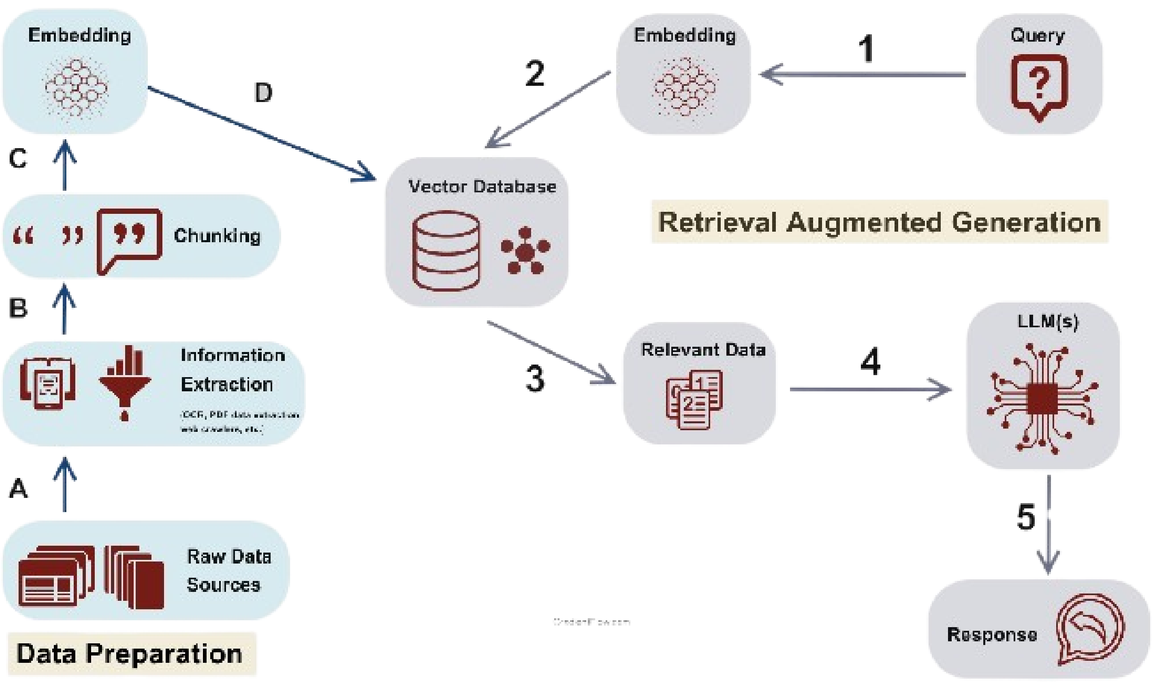
Architecture du Système

L'architecture suit trois étapes principales :

Extraction et segmentation du contenu PDF en segments cohérents.

Vectorisation des segments (384 dimensions) via all-MiniLM-L12-v2 afin de capturer les liens sémantiques.

Indexation et recherche dans ChromaDB. Lorsqu'une question est posée, elle est vectorisée puis comparée aux segments du corpus pour récupérer les cinq plus pertinents qui serviront de contexte.



Mécanisme de Génération de Réponses

Le système répond en deux phases :

1. Reformulation contextuelle
Pour les questions dépendantes du contexte (ex. « Quels sont ses prérequis ? »), Gemini reformule la requête de manière explicite et autonome, en intégrant la mémoire de la conversation.

2. Recherche et génération
La question reformulée est vectorisée puis comparée aux segments stockés. Les cinq plus proches sont intégrés dans un prompt système qui oblige Gemini à répondre uniquement sur la base de ces documents et en français.

La conversation complète est mémorisée pour garantir cohérence et continuité sur plusieurs échanges.

Implémentation Technique

Le système est développé en Python et présenté via une interface Streamlit de type chat.

Les composants principaux sont :

- Gemini 1.5 Pro (température 0.1 pour la reformulation, 0.5 pour la réponse),
- HuggingFace Inference pour la génération des embeddings,
- LangChain pour orchestrer les différentes étapes,
- ChromaDB pour la recherche vectorielle en temps quasi-réel.

Discussion

Points forts

- Fidélité aux documents officiels.
- Gestion naturelle du contexte conversationnel grâce à la mémoire intégrée.
- Segmentation avec chevauchement préservant le sens des passages complexes.

Limitations

- Qualité d'extraction variable selon les PDF.
- Latence due aux deux appels successifs au modèle.
- Portée limitée aux documents de l'IFRI.
- Coûts liés à l'utilisation de modèles de grande taille.

Perspectives

- Intégration d'outils d'explicabilité pour visualiser les passages utilisés.
- Complément de recherche lexicale pour améliorer le traitement des acronymes.
- Test du système sur d'autres institutions.
- Développement d'une version mobile pour les usages administratifs sur le terrain.

Conclusion

Ce projet démontre l'efficacité d'une architecture RAG pour l'accès aux connaissances institutionnelles, en combinant recherche sémantique et génération contrainte.
L'interface Streamlit offre un outil immédiatement exploitable pour les acteurs de l'IFRI, améliorant considérablement la consultation des informations réglementaires et académiques.