Projet de Module Machine Learning 1, IDSCC 3

Système Interactif de Questions-Réponses pour la Constitution

Préambule

Ce guide vise à vous accompagner dans le développement du projet. Il présente une approche structurée, mais vous êtes encouragés à explorer des solutions alternatives et à résoudre les problèmes selon votre propre vision. L'innovation et la créativité sont les bienvenues tant que les objectifs principaux du projet sont atteints.

1 Introduction

Ce projet vise à développer un système de questions-réponses interactif utilisant la Génération Augmentée par Récupération (RAG) pour répondre aux questions des utilisateurs concernant la Constitution. Ce chatbot permettra aux utilisateurs de poser des questions en langage naturel et récupérera les sections pertinentes du document pour générer des réponses précises et contextuelles.

2 Étapes du Projet

2.1 Semaine 1-2 : Analyse et Préparation des Données

1. **Extraction du texte de la Constitution**
   - Utiliser PyPDF2, pdfminer.six ou d'autres outils d'extraction de texte
   - Évaluer la qualité de l'extraction et résoudre les problèmes de formatage
   - Explorer des alternatives comme les API OCR pour les documents de mauvaise qualité
2. **Prétraitement et segmentation du texte**
   - Nettoyer le texte (supprimer les en-têtes, les notes de bas de page, etc.)
   - Segmenter en unités logiques (articles, sections, clauses)

— Conserver les métadonnées importantes (numéros d'articles, titres de sections)

3. **Analyse de la structure du document**
   — Identifier la hiérarchie du document (titres, sous-titres, articles)
   — Développer des règles pour traiter les références croisées
   — Créer un schéma de métadonnées pour faciliter la récupération contextuelle

2.2 Semaine 3-4 : Développement du Système de Récupération

1. **Création d'embeddings vectoriels**
   — Utiliser des modèles de Sentence Transformers pour créer des représentations vectorielles
   — Évaluer différents modèles pré-entraînés (BERT, MPNet, LegalBERT)
   — Optimiser la taille des chunks pour la représentation vectorielle
2. **Construction d'une base de données vectorielle**
   — Implémenter FAISS, Pinecone, ou Chroma pour le stockage et la recherche de vecteurs
   — Tester la performance de récupération avec différentes métriques de similarité
   — Optimiser les index pour l'efficacité de la recherche
3. **Développer un moteur de recherche sémantique**
   — Implémenter des algorithmes de recherche par similarité
   — Intégrer des techniques de ré-ordonnancement pour améliorer la pertinence
   — Développer des mécanismes de filtrage basés sur les métadonnées

2.3 Semaine 5-6 : Implémentation du Système de Génération

1. **Intégration d'un modèle de langage**
   — Sélectionner un modèle approprié (OpenAI API, Llama, Mistral, etc.)
   — Configurer le modèle pour la génération de réponses
   — Implémenter un wrapper pour standardiser les interactions avec l'API
2. **Développement du système RAG**
   — Concevoir l'architecture pour combiner récupération et génération
   — Implémenter des techniques de prompt engineering pour guider le LLM
   — Développer des mécanismes pour citer les sources dans les réponses

3. **Optimisation de la pertinence des réponses**
   - Expérimenter avec différentes stratégies de prompt
   - Ajuster la température et d'autres paramètres de génération
   - Implémenter des techniques de vérification post-génération

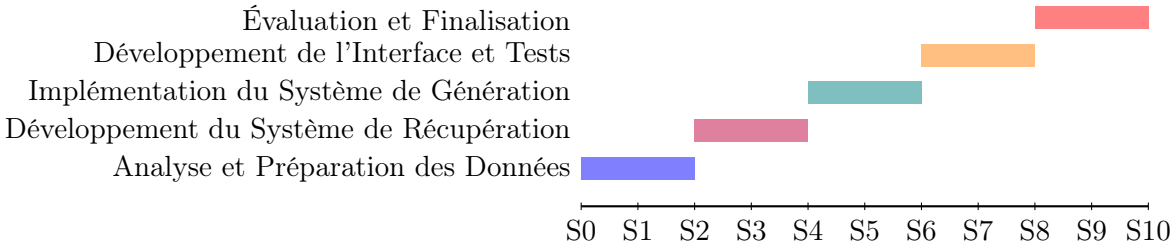
2.4 Semaine 7-8 : Développement de l'Interface et Tests

1. **Création de l'interface utilisateur**
   - Développer une interface web simple (Flask, Streamlit, Gradio)
   - Implémenter l'historique des conversations et la gestion du contexte
   - Ajouter des fonctionnalités pour afficher les citations et références
2. **Tests initiaux du système**
   - Créer un ensemble de questions de test couvrant différents aspects de la Constitution
   - Évaluer la précision et la pertinence des réponses
   - Identifier les cas difficiles et les problèmes communs
3. **Optimisation des performances**
   - Améliorer la vitesse de récupération
   - Ajuster les paramètres du système en fonction des résultats des tests
   - Implémenter des techniques de mise en cache pour les requêtes fréquentes

2.5 Semaine 9-10 : Évaluation et Finalisation

1. **Évaluation complète du système**
   - Mener des tests utilisateurs
   - Évaluer selon des métriques établies (précision, pertinence, vitesse, etc.)
   - Comparer avec d'autres approches ou systèmes existants
2. **Finalisation et documentation**
   - Résoudre les problèmes identifiés pendant l'évaluation
   - Documenter l'architecture du système et le code
   - Préparer un guide d'utilisation et de maintenance
3. **Explorer les extensions possibles**
   - Support multilingue
   - Intégration d'autres documents juridiques
   - Visualisation des relations entre différentes parties de la Constitution

3 Diagramme de Gantt



4 Technologies et Outils Recommandés

4.1 Extraction et Prétraitement des Données

- PyPDF2, pdfminer.six, ou Tesseract OCR (pour l'extraction de texte)
- NLTK ou spaCy (pour le traitement du langage naturel)
- Regular expressions (pour le nettoyage et la segmentation)
- langchain-text-splitters (pour la segmentation avancée)

4.2 Vectorisation et Récupération

- Sentence-Transformers (pour la création d'embeddings)
- FAISS, Pinecone, Chroma, ou Weaviate (pour la base de données vectorielle)
- LangChain ou LlamaIndex (frameworks pour faciliter le RAG)
- BM25 ou ElasticSearch (pour la récupération hybride)

4.3 Génération et Interface

- OpenAI API, Hugging Face, Llama, Mistral (pour le modèle de langage)
- Flask, FastAPI, Streamlit, ou Gradio (pour l'interface utilisateur)
- Redis ou SQLite (pour la mise en cache et la gestion de session)
- pytest (pour les tests automatisés)

5 Considérations Techniques Importantes

5.1 Défis Liés à l'Extraction et au Prétraitement

- **Qualité de l'extraction** : Les documents PDF juridiques peuvent présenter des défis spécifiques (colonnes, notes de bas de page, etc.)

— **Segmentation précise** : Trouver l'équilibre optimal entre des segments trop courts ou trop longs
— **Maintien des références** : Conserver la traçabilité vers le document original

5.2 Défis Liés à la Récupération et à la Génération

— **Langage juridique spécialisé** : Les termes juridiques peuvent nécessiter des modèles spécialisés
— **Ambiguïté des questions** : Les utilisateurs peuvent poser des questions ambiguës ou mal formulées
— **Hallucinations du LLM** : Risque que le modèle génère des informations incorrectes
— **Latence du système** : Équilibrer la rapidité et la qualité des réponses

5.3 Considérations Éthiques et Pratiques

— **Transparence** : Indiquer clairement que le système n'est pas un conseil juridique
— **Attribution des sources** : Citer précisément les parties de la Constitution utilisées
— **Gestion des limites** : Reconnaître quand une question dépasse les capacités du système

6 Extensions Possibles

— Intégration d'autres textes juridiques (lois, jurisprudence, etc.)
— Ajout de fonctionnalités d'explication ou de simplification du langage juridique
— Développement d'une analyse comparative entre différentes constitutions
— Création d'un système de recommandation de lectures supplémentaires
— Implémentation d'un mécanisme de feedback pour l'amélioration continue

Note Finale

Ce guide présente une structure recommandée pour le projet, mais n'hésitez pas à l'adapter selon vos idées et votre vision. L'objectif principal est de développer un système de questions-réponses efficace qui fournit des informations précises et pertinentes sur la Constitution. Vos approches innovantes pour résoudre les défis rencontrés sont encouragées et valorisées.