Rapport Sprint 2

Changement de la librairie: de pdfMiner à pyMuPDF :

Au cours de ce sprint, nous avons pris la décision de remplacer la librairie pdfMiner par pyMuPDF. Cette décision a été motivée par plusieurs facteurs clés :

1. Performance: pyMuPDF offre une vitesse de traitement plus élevée que pdfMiner, ce qui est crucial pour l'analyse et l'extraction de données à partir de documents PDF volumineux ou complexes.

2. Facilité d'utilisation: L'API de pyMuPDF est plus simple et plus intuitive, ce qui réduit le temps de développement et facilite la maintenance du code.

3. Fonctionnalités avancées: pyMuPDF possède des fonctionnalités supplémentaires qui n'étaient pas disponibles ou étaient moins efficaces dans pdfMiner.

Etape de réalisation du Parseur d'articles scientifiques en format texte :

Titre :

Pour récupérer le titre du PDF on utilise la fonction : find_title(path). Elle détermine la taille de police cible et extrait le texte correspondant à cette taille de police comme titre potentiel. Cette fonction utilise les fonctionnalités de pyMuPDF pour accéder au contenu du premier bloc de texte de la première page, ce qui est souvent où les titres sont situés.

Ensuite, avec la fonction get_size(path) on parcourt les blocs de texte et compare leurs tailles de police pour trouver la plus grande. Cela est essentielle pour identifier le titre dans la fonction find_title(path), en supposant que le titre est en plus grande taille de police par rapport au reste du texte.

Auteurs :

Pour récupérer les auteurs on cherche dans les blocks compris entre le résumé et le titre et en prenant tout ce qui ressemble à un nom

-On vérifie les métadonnées du document PDF à la recherche des auteurs. Si les auteurs y sont clairement indiqués, cette information est extraite et utilisée directement.

-Si les métadonnées ne fournissent pas d'informations sur les auteurs, l'étape suivante consiste à rechercher des adresses email dans le document.

-En l'absence d'informations claires dans les métadonnées et d'adresses email, l'analyse se concentre sur le contenu textuel situé entre l'abstract et le titre du document.

-Une fois la zone cible identifiée, les fonctions recognize_name et recognize_author sont appliquées pour détecter les chaînes de caractères qui correspondent à des formats de noms

-Les noms identifiés sont ensuite formatés et préparés pour la présentation finale. Les fonctions make_abr et make_name contribuent à transformer les adresses email et les noms détectés en une liste structurée d'auteurs.

Rapport Sprint 2

Résumé :

Pour récupérer le résumé :

-On détermine les positions de l'abstract et de l'introduction

- Si l'abstract est clairement délimité (identifié avant l'introduction et séparé par le mot "Abstract"), elle procède à l'extraction du texte.

-Si l'abstract se trouve réparti sur plusieurs blocs de texte, la fonction tente de combiner ces blocs pour former le texte complet de l'abstract. Cela est fait en utilisant biggestBlock pour trouver le bloc principal contenant l'abstract et, si nécessaire, smallestBlock pour ajouter du texte supplémentaire manquant.

-Enfin, si l'abstract n'est pas explicitement identifié, la fonction essaie d'extraire le texte précédant immédiatement l'introduction comme étant potentiellement l'abstract.

Conclusion :

En conclusion, le script utilise pyMuPDF pour améliorer l'extraction de texte souchaité du PDF en exploitant ses capacités de traitement de texte et d'image. L'approche combinée d'analyse des métadonnées et du contenu textuel vise à augmenter la précision de l'extraction des textes.

Problèmes non résolus et Plans pour le prochain sprint :

Malgré les progrès réalisés dans ce sprint, certains objectifs n'ont pas été atteints. Nous prévoyons d'aborder ces problèmes au prochain sprint en explorant des approches plus avancées d'analyse de texte et de reconnaissance de motifs.