PdfToTxt

Romain Le Talour, Corentin Leconte, Thibaut Quilleré

Avril 2021

Abstract

PdfToTxt est l'issue d'un sujet commun donné à l'ensemble des groupes de l'UE Projet de développement et en voici notre approche. Le projet que nous menons a pour but de développer un logiciel permettant de convertir un document PDF scientifique. Les formats de conversion supportés sont le texte ainsi que le XML. Ce projet à été réalisé dans un but scolaire.

1 Méthodes

Le parseur d'articles scientifiques en format texte et format XML repose sur le language Python ainsi que du logiciel de conversion de fichiers PDF en fichiers texte nommé << pdftotext >>.

Le script demande à l'utilisateur le chemin absolu ou relatif du répertoire contenant les fichiers PDF voulant être converties par l'utilisateur, le logiciel récupère ensuite touts les fichiers dont le nom est terminé par le suffixe << .pdf >> qui sont les suffixes utilisés par les fichiers PDF, le script, avec les fichiers PDF repérés dans le dossier construira un dictionnaire qui contiendra le nom du PDF, et a chaque PDF, un identifiant entier unique. L'utilisateur pourra donc choisir de convertir seulement une partie des fichiers ou la totalité en utilisant le caractère "*". Enfin, le script demandera un dernier argument pour connaître le format de sortie des fichiers, soit en format texte, soit en format XML.

Lorsque toutes les arguments seront donnés par l'utilisateur, le script se comportera de la façon suivante :

- Il crée, dans le répertoire donné par l'utilisateur, un sous-répertoire nommé "parsers" qui contiendra les fichiers PDF convertits par pdftotext, il crée ensuite un autre sous-répertoire nommé "ParserOutput" qui contiendra quand à lui, les fichiers de sortie dans le format demandé par l'utilisateur, il faut noter que si ces 2 dossiers sont déja présents dans le dossier donné par l'utilisateur, ces 2 répertoires seront effacés puis recrées.

- Ensuite, pour chaque fichier ciblé par l'utilisateur, une fonction construirera un dictionnaire, étant une structure de données du language Python, afin de stocker les informations suivantes :

  – Le nom du fichier PDF

- Le titre de l'article
- Les auteurs
- l'affiliation des auteurs
- l'abstract de l'article
- l'introduction de l'article
- le corps de l'article
- la conclusion de l'article
- la discussion de l'article
- les références de l'article

Ces informations sont récupérées du fichier PDF grâce à deux manières :

- La première étant l'extraction des méta-données présentes dans le fichier PDF, la méthode étant jugée plus fiable, plus rapide et plus efficace que la seconde méthode, n'est néanmoins beaucoup plus limitée que la seconde méthode, car les métadonnées ne concerneront qu'une partie des informations recherchées, comme le titre, les auteurs, l'abstract et si seulement ces informations ont étées enregistrées dans les méta-données.

La seconde méthode est la conversion du fichier PDF en fichier texte et d'effectuer du traitement de texte sur le dit fichier afin de récupérer les informations souhaitées. Lors de la réflexion sur la création et des outils à utiliser pour le parseur d'article scientifiques, 2 choix avaient étés proposés: pdf2txt et pdftotext, 2 logiciels de conversion de fichiers PDF en fichiers texte disponibles sur Linux. Les premières versions du script utilisaient le logiciel pdf2txt, mais le logiciel pdf2txt fut rapidement remplacé par pdftotext, qui proposait de meilleurs options et réglages pour la conversion et permettait une apparition moindre d'artefacts venant troubler le script sur la récupération des informations.

Suite à la conversion du fichier PDF en fichier texte, la récupération des informations se fait grâce au << RegEx >> ou nommé << Regular Expression >> RegEx, étant une suite de caractères typographiques permet de rechercher, de modifier et de manier du texte, un outil très puissant utilisé dans de nombreuses applications dans le domaine informatique. Ici, seul les fonctions de recherche sont utilisées. Le language Python permet d'importer un package nommé << re >>, un package présent dans la libraire standard Python qui contient le code nécessaire pour utiliser du RegEx au sein du parseur. Ainsi, pour chaque élément recherché afin de construire le fichier de sortie, une fonction fût créée pour récupérer l'élément, la regex recherchera un certain pattern, si ce pattern n'est pas retrouvé dans le fichier texte, la fonction retournera un String qui notifira que l'élément n'as pas été trouvé, ce cas peut être rencontré lorsque l'article scientifique ne suis pas un modèle standard, ou lorsque le convertissement du fichier PDF en fichier texte a créé des artefacts

empêchant la bonne lecture du fichier texte. Des solutions à ces problèmes peuvent être utilisées (cf. Conclusion ou Discussion).

Enfin, le script suivera la logique suivante : Pour chaque fichier PDF donné par l'utilisateur, le parseur appellera pdftotext afin de convertir le fichier PDF en fichier texte qui le mettra dans le répertoire "parsers". Le parseur ouvrira le fichier PDF afin de récupérer les méta-données Ensuite, pour chaque élément recherché, le parseur cherchera d'abord a récupérer l'élément dans les méta-données, si l'élement n'est pas trouvé, le parseur appellera la fonction correspondante à l'élément pour le rechercher avec une expression régulière dans le fichier texte. Si l'élément n'est toujours pas trouvé, le parseur stocke un String qui notifira l'échec du repérage de l'élément.

Lorsque touts les éléments ont étés recherchés, le parseur appellera une fonction pour créer le fichier de sortie, cette fonction nommée "writeFile" écrirera, selon le paramètre de sortie donnés par l'utilisateur un fichier texte ou un fichier XML, et mettrera toutes les informations nécessaire au parsage du fichier PDF. Le parseur fermera ensuite son accès au fichier PDF, au fichier convertit en txt et au fichier de sortie et continuera jusqu'à ce que tous les fichiers donnés par l'utilisateur ont été parsés. Ensuite le parseur s'arrêtera, n'ayant plus de traitement à faire.

2 Résulats

Pour mesurer l'efficacité de notre parseur, nous nous sommes appuyés sur un corpus de test. En effet, nous avons eu 10 PDF rassemblés dans un corpus que nous avons dû découper manuellement dans un fichier XML. Ensuite nous avons converti les PDF avec notre logiciel et avons comparé notre propre découpage avec celui du parseur. Pour mesurer les résultats, c'est-à-dire le calcul de la précision, nous avons utilisé une formule qui est :

```
Precision = SectionsCorrectesTrouvéesParLeSystème / SectionsVéritables
```

Voici donc nos résultats :

acl2012 :
Titre : 84.8% 0points
Emails : 100% 1points
Auteurs : 63.15%+54.54%+38%+53.84+55.55% = 53.016%
Abstract : 98.88% 1points
Introduction : 91.05% 1points
Corps : 95.61% 1points
Conclusion : 97% 1points
Discussion : 92.36% 1points
Biblio : 64% 0points
6/13 0.46

b0e5 :
Titre : 100% 1points
Auteurs : 100%, 98, 98, 3points
Emails : 100% 1points
Abstract : 100% 1points
Introduction :93% 1points
Corps : 24% 0points
Conclusion : 27% 0points
Biblio : 94.7% 1points
8/10 0.8

BLESS :
Titre : 100% 1points
Auteurs : 100% 100% 2points
Emails : 100% 100% 2points
Abstract : 99% 1points
Introduction : 92% 1points
Corps : 93.9% 1points
Conclusion : 99% 1points
Biblio : 19% 0points
9/10 0.9

C14-1212 :
Titre : 100% 1points
Auteurs : 100% 100% 100% 100% 100% 5points
Emails : 100% 1points
Abstract : 93.82% 1points
Introduction : 58% 0points
Corps : 92.83% 1points
Conclusion : 99.35% 1points
Biblio : 99.88% 1points
11/12 0.91

Guy :
Titre : 97.87% 1points
Auteurs : 40% 40% 0% 0points
Emails : 100% 100% 100% 3points
Abstract : 76% 0points
Introduction : 98% 1points
Corps : 88% 0points
Conclusion : 99% 1points
Biblio : 90% 1points
7/12 0.58

infoEmbeddings :
Titre : 98% 1points
Auteurs : 0% 0points
Emails : none
Abstract : 72% 0points
Introduction : 74% 0points
Corps : 91% 1points
Conclusion : 98% 1points
Biblio :97.03% 1points
4/7 0.57

IPM1481 :
Titre : 0% 0points
Auteurs : 100% 100% 100% 0% 100% 4points
Emails : 100% 100% 40% 100% 3points
Abstract : 2% 0points
Introduction :48% 0points
Corps : 0% 0points
Conclusion : 19% 0points
Biblio : 93% 1points
8/13 0.6

L18-1504 :
Titre : 97% 1points
Auteurs : 100% 0% 0% 0% 1points
Emails : 100% 0% 0% 0% 1points
Abstract : 99.75% 1points
Introduction : 97% 1points
Corps : 0% 0points
Conclusion : 95% 1points
Biblio : 22% 0points
6/14 0.4

On The Morality of Artificial Intelligence
Titre : 66% 0points
Auteurs : 0% 0% 0points
Emails : None
Abstract : 40% 0points
Introduction : None
Corps : 87% 0points
Conclusion : 71% 0points
Biblio : 96% 1points
1/6 0.16

SurveyTermExtraction
Titre : 100% 1points
Auteurs : 0% 0% 0% 0points
Emails : 100% 100% 100% 3points
Abstract : 100% 1points
Introduction : 15% 0points Corps : 34% 0points
Conclusion : 4% 0points
Biblio : 97% 1points
6/12 0.5

Moyenne générale : (0.46+0.8+0.9+0.91+0.58+0.57+0.6+0.4+0.16+0.5)/10= 0.548 = **54.8% de précision.**

3 Conclusion

Pour conclure ce projet, nous pensons que nous avons mené à bien notre développement et que le cahier des charges est respecté. La moyenne que nous obtenu sur le calcul de la précision est tout à fait correct et nous nous en satisfaisons. Il faut noter cependant des améliorations possibles, au départ de la création du parser, seul 2 logiciels de convertisseur PDF étaient proposés, pdf2txt et pdftotext, suite a l'apparition de nombreux problèmes comme l'apparition d'artefacts ou de difficultés a trouver une expression régulière qui permettrait de trouver un pattern exact, il peut être intéressant de se tourner vers des convertisseurs PDF plus complets et plus performants comme PDFMiner disponible sous le langage Python, qui permettrait une meilleure précision grâce à l'utilisation de la taille des caractères ou la présence de lettres en gras et aussi éviter l'apparitions d'artefacts pouvant être dommagables pour la recherche avec les expressions régulières.

Il peut être aussi possible d'utiliser l'intelligence artificielle pour récupérer les bons éléments, on peut par exemple, utiliser l'intelligence artificielle afin de récupérer les noms et prénoms des auteurs sans avoir de faux positifs, il faut cependant tester et vérifier si l'apport de cette méthode peut être intéressante comparé au défi technique que cette méthode apporte.