Document Parsing Tool for Language Translation and Web Crawling using Django REST Framework

Kruthika Alnavar¹, R Uday Kumar² and C Narendra Babu³
alnavar8@gmail.com¹, udaykumar4578@gmail.com², narendrababu.cs.et@msruas.ac.in³
¹,² M. Tech Student, Department of Computer Science, M S Ramaiah University of Applied Sciences, Bangalore, India.
³ Professor, Department of Computer Science, M S Ramaiah University of Applied Sciences, Bangalore, India.

Abstract. There are 7.5 billion inhabitants and over 7,117 languages existing around the world, but only 20% of the people speak English. To understand the wisdom and knowledge of other cultures language translation becomes a basic need. In this paper, a computer-assisted document parsing tool is investigated. The proposed approach uses a language translator that performs translation from images eliminating the need of a human translator for images avoiding the scope for misinterpretation and misunderstanding among people of different ethnic groups. The proposed tool is also capable of performing web crawling using Django Representational State Transfer framework. Further, the proposed approach employs Python packages such as pytesseract, textblob and beautifulsoup to perform Optical Character Recognition, Translation and Extraction of Hypertext Markup Language data respectively. Experimental results of translation on four different categories of images such as Maps, Comics, Newspapers and Magazines, Scientific Publications demonstrate an accuracy of 97.2%, 93.3%, 95.82% and 98.27% respectively. By considering websites like E-commerce, Magazines, Blogs, Social Media, News and Educational sites average precision of 5.4, recall of 7.45 and F-score of 6.24 is achieved. The results reveal that the proposed system can be used as an improvement over a human translator and a data entry operator.

Keywords — API, Django, Language Translation, OCR, REST framework, Web Application, Web Crawling.

1. Introduction
Transferring of one kind of information into another is called parsing of a document. Generally, the data in formats such as Portable Document Format, email, websites and images are being transferred into readable and understandable format by businesses to use and analyse it. For instance, the large number of resumes and job notifications from different emails and websites can be parsed into one single readable document. This process is tedious and requires a lot of manual work as each of the email or website is different from the other. A study conducted in 2008 found that the probability of human error was 18% to 40% while performing simple tasks such as entering data into sheets [1]. Here the need for a document parsing tool comes into play. A tool like this can help a company by eliminating the requirement for manual data entry and improving overall accuracy, time and expenditure. It removes the duplication error and also makes the data searchable.

The proposed system is a document parsing approach to perform parsing and comprises of features such as language translation and web crawling in a single platform. We know that there are 7.5 billion inhabitants and over 7,117 languages around the world, but only 20% of the people speak English [2]. It appears that English is the most used language over internet, but until January 2020, only 25.9% of...
internet users throughout the world represent English. To prevent the loss of wisdom, knowledge and culture of the remaining 74.1% of internet users it is essential to perform language translation from one language to another so that the rest of the world can be benefitted. The other feature that we have provided is spiderbot also known as a web crawler which provides automated script. It can be described as a program which crosses the web to download web files in a methodological and automated way.

Web Search Engines face new problems due to the existence of a wide range of web documents. A web search engine has gained considerable importance in modern days and comprises of two main constituents namely, the web crawler which downloads and parses the material on the world-wide network and the data miner which extracts keywords from pages. The extracted keywords can be converted in to a corpus of words followed by pre-processing and cleaning of data [3].

In this context, a document parser system is proposed in this paper which is capable of performing web crawling and language translation of images submitted by the client using a Django Rest Framework based web application. This paper unfolds into four more sections. Section II discusses the related work. The methodology and system design are discussed in section III. The fourth section describes the results. Ultimately, this paper is concluded in the fifth section.

2. Related Work
This section discusses some of the contemporary tools and research works with respect to document parsing, language translation and web crawling.

2.1. Document Parsing
There are a few document parsing tools such as DocInfusion, Docparser and Automatio. The web application DocInfusion stores data on their secure servers, and provide options to either download or access it via web service. PDFs, Excel files, text and word documents are the main focus of DocInfusion. Docparser on the other hand provides Quick Response code Detection and custom parsing rules. Automatio is another parsing tool which has an excellent user interface with a Data panel and deals with data formats such as Application Programming Interface, JavaScript Object Notation, Rich Site Summary and Comma Separated Values.

A previous study proposed an end-to-end architecture based on a deep neural network which extracts data on the basis of a core assumption that every document has the same structured information [4]. Another study highlights the information extraction from table fields such as purchase orders using recurrent neural network wherein an external commercial Optical Character Recognition system is used for data retrieval [5]. During document parsing, the functioning of a certain topic-specific web crawler can also be improved by considering the structure of documents that are downloaded by the website crawler [6].

2.2. Language Translation
An application is developed for Health Information System for digitization of documents using Python library [7]. Various libraries such as PyOCR, TesserOCR, and PyTesseract are compared with respect to the precision and execution speed. The study showed that TesserOCR was the fastest Python Library whereas PyTesseract was the most precise OCR library. PyOCR showed better performance when larger area is considered for analysis. Hence, PyTesseract can be used in cases that allow the tradeoff of execution time and requires precise output, PyOCR can be ideally used for applications such as scanning and TesserOCR for fast applications.

2.3. Web Crawling
Web-crawlers are described as a vital source of data retrieval [8]. A two-stage crawler system is proposed for effective crawling of web pages which executes a backward search and gradually optimizes to balance the type and web document query contents [9].

Distinct crawling techniques can be used for crawling hidden web files in various ways [10]. The main crawling manager posts a fragment of code from the crawler to the server for fetching hidden information. The crawl manager schedules the crawlers so as to prevent two crawlers from accessing the identical address [11]. A study in paper [12] includes a design of the focused webpage crawler
employing genetic algorithm. It has a detection policy to identify the changes in webpage. Data function and jaccard method can be used to identify connected web pages [13]. The main objective is the selection of the best links for maximum relevancy of latest and not visited Uniform Resource Locator via the application of genetic algorithm.

Two methods of crawler are supervised learning and another determines advantage of the link after moving them, and using those links to evaluate next [14]. URLs are used to analyse and filter the web pages. Later the hyperlinks from this webpage is obtained and the text content is extracted. Similarly the crawler extracts the image from this hyperlinks and analyses its size and format [15].

3. Methodology
In the proposed system the user has to register and login to the fully featured web document parsing system to access the features, namely language translation and web crawling. The output is saved as a .txt file or .csv in the user’s device. Input such as images and URLs are accepted and a user-friendly interface is created using Cascading Style Sheets and Bootstrap. The images are uploaded to the database. Extra feature is provided for creating, updating and deleting of user reviews on the home page.

3.1. Django Rest Framework
As shown in figure 1 Django Rest Framework is used to create a functional web application to create an application programming interface. REST uses a Representational state transfer type of architectural pattern for creating web services to provide interoperability between two or more systems on the internet. The API helps to separate the Django templates on the client side and the database on the server side. The communication happens via .json file.

The models are python classes, where each of them is mapped to a separate database table. Views on the other hand are python functions that accept a web request and return web response based on the logic. This framework takes away the complexity of interaction with database and prevents duplication of data by connecting to the same API.

3.2. Fully Featured Web App
The Document Parsing Tool as shown in figure 2, is a full featured web application. It includes separate apps or sub-models called Reviews, Language Translation, Web Crawling and Users. The Reviews app helps in creating, updating and deleting the user reviews on the home page. The Users app helps in registration of users, login, and logout to access features. Also, the user information such as email id, password and profile picture are saved in the database. Web crawling app includes a POST request to fetch the URL and the output is saved in the form of .csv. Each of these apps have models that create a table in the database.
3.3. Pytesseract
Python tesseract wrapper uses Google’s Tesseract-OCR engine which is used to recognize the text embedded in images. A wrapper helps in converting the data into a suitable format. Here the Pytesseract wrapper is used to hide the intricacy of the underlying entity. It also reads images of tiff, bmp, jpeg, png, gif and other formats, supported by Leptonica and Pillow libraries.

3.4. TextBlob
It is a python library mainly used for processing of text and can perform Natural Language Processing tasks like POS tagging. Noun phrase extraction, sentimental analysis, classification using DT, tokenization parsing, spelling correction and many other features are provided for Natural Language Tool Kit applications.
3.5. Language Translation

The figure 3 shows the detailed block diagram of the Document Parsing Tool. On logging in to the Document Parsing Tool options such as Language Translation and Web Crawling become accessible as shown in Fig 4. The steps for Language Translation are as follows:

1. Accept an input image
2. Detect and OCR text from images
3. Translate the OCR’d text
4. Save the result as a .txt file.

Here the python function, accepts three inputs from the user and passes these values to the other function. The three inputs are the language of the input image, the image itself, and the output translation language. A list of languages is displayed to be selected by the user.

A dictionary consisting of key value pairs is created, where in the input language is mapped to the key word which is given to the selected input and output languages. OpenCV package cv2 is used to read the image and BGR image is converted to RGB. Using: pytesseract.image_to_string, the text from the image is recognized and converted into a string and this string is printed in the terminal. TextBlob() performs various natural language processing tasks such as tagging parts of speech. We will use it to translate OCR’d text. The command ‘translate’ will provide translation of text into major languages as already listed. Once translated, the text is saved in the desired location in the form of a text file.

3.6. Web Crawling

Web crawling involves navigating through the list of pages on the internet and connecting with them automatically. It rapidly and effectively collects as many helpful web pages as feasible, along with the link structure linking them. The Web crawling process is seen in Figure 3. The block diagram consists of the mechanism for crawling from the URL. These spiders are primarily used for make a backup of the accessed pages required for subsequent processing through the crawler that indexes the pulled down webpages to ensure quick searches.

The framework is designed to terminate any redundant details for the user. The process is structured in such a way as to reiterate which helps to update and view new pages, eliminating repetition and replication of pages. The Depth First Search Algorithm is implemented to help the system view one web-related hyperlinks before switching to the next one. It then backtracks and switches to the adjacent hyperlink sites. Backtracking is used to find the next unvisited connection, and this process is repeated in a similar manner to all hyperlinks. During this process, the links passed through the crawler are placed in a stack that can be used for backtracking. This helps to confidently conclude that all hyperlinks are accessed at least once during the whole process. There should be at least one connection in the waiting set for the hyperlinks to be indexed and the algorithm to start crawling.
The crawler also offers an ability to determine the maximum width of the connection that can be
reached by the crawler in case of narrowing down the search for particular purposes. In case of absence
of any other link path, the algorithm shall terminate, thus providing the analyser with the necessary
results under the defined conditions. Later, the text data that is crawled from the webpage can be stored
in either text or csv format for additional applications. Transformation process for adding several pre-
processing measures to the data is carried out. The preprocessing stages depend on the scope as well as
the problem, so that we don't have to implement all the actions to every problem. The stored text or csv
file after performing web crawling can be used as data for the transformation part.

3.7. Document Parser
The Document Parser Web Application is shown in the figure 4. Start page includes navigation panel
with options such as Home, About, Profile, Web Crawling, Language Translation, Write a review and
Logout. The users can post their reviews after logging into
the website.

The website is created using Python, HTML, CSS, Javascript, Bootstrap, SQL database and Django
Rest framework. Rest framework provides easy creation, updation and deletion of user reviews using
GET, POST and DELETE methods. The user profile
can also be created.

Language translation is shown in figure 5 where the ImageField and TextField inputs are provided.
The uploaded image is displayed and on clicking run, a .txt file containing the output is saved on the
device. Figure 6 shows web crawling which accepts a URLField and the output is saved. Once the URL
is given the user is given a choice to crawl text or images and output can be saved as .txt or .csv file.

4. Results and Discussion
Python libraries that perform language translation are pytranslate, TextBlob, googletrans, goslate and
translate. Pytesseract is used to perform OCR followed by TextBlob for translation. The proposed
system includes 111 languages using Pytesseract and performed translation on 7 different varieties of
images, namely Maps, Comics, Legal Documents, Sign boards, Manuscripts, Magazines and
Newspapers and Scientific Publications. The results of translation are as shown in Table 1. Sign boards,
Artifacts and Legal Documents achieved an accuracy of 90%, 85.4% and 98.3% respectively. Accuracy
Translation might lengthen or shrink the output depending on the language. This is due to the fact that languages differ with respect to their word usage, grammar and syntax. The linguistics of document also affects the word count. For example, if we consider an advertisement as input then it is likely to expand because additional specifics and describing words are required to convey the message. Whereas when we consider a scientific publication the output language is likely to be shorter and summarized.
The high accuracy of word count does not imply that the translated content is accurate. In fact, it depends on the OCR performed by PyTesseract. We have compared the similarity of the translated text between TextBlob and Pytranslate and found a 98% of similarity for 50 samples as shown in Table 2.

The accuracy of OCR relies on the factors such as the properties of the image, the size of text, and illumination. Also, the image should not contain overlap of text and watermarks as it makes the text difficult to be recognized. If any words are missed during OCR then it is not possible to translate them, this could change the entire meaning of the sentence. The designed web crawler eliminates all unnecessary information from the user and analyser. The framework only takes the stated URL from the user as input. The average efficiency of the tool shall be determined using below equations:

\[
Precision = \frac{\text{No. of relevant pages retrieved}}{\text{Total no. of pages in website}}
\]  

(1)

\[
Recall = \frac{\text{No. of relevant pages retrieved}}{\text{Total no. of relevant pages retrieved}}
\]  

(2)

\[
F - \text{score} = \frac{2 \times Precision \times Recall}{Precision + Recall}
\]  

(3)

Table 2. Similarity of text

| Translation (100 words) | Pytranslate | Googletrans |
|-------------------------|-------------|-------------|
| Similarity with TextBlob| 98%         | 99%         |

Table 3. Percentage of extraction for different websites

| Website Type    | Percentage of extraction | Reason for failure of extraction                                                                 |
|-----------------|--------------------------|---------------------------------------------------------------------------------------------------|
| E-commerce      | 60%                      | Due to the change in the format of script in different e-commerce websites it becomes difficult to apply a generalized method. |
| E.g.: Flipkart, Amazon                                      |                                                                                                  |
| Magazine        | 75 – 85%                 | Error due to the presence of different tag formats and many images.                               |
| E.g.: Abstract Elephant, Vogue Blogs                        |                                                                                                  |
| Blogs           | 87%                      | Page does not get indexed.                                                                       |
| E.g.: Medium                                             |                                                                                                  |
| Social Media Sites | 62%                      | Due to protected tags, unauthorized person cannot access the pages                               |
| E.g.: Facebook, Twitter                                    |                                                                                                  |
| News            | 86%                      | Connectivity and DNS issues cause error                                                           |
| E.g.: India Today                                         |                                                                                                  |
| Educational sites | 92-95+%                  | Timeout or server error due to free access                                                       |
| E.g.: Wikipedia, IEEE                                      |                                                                                                  |

Level of extraction is different depending on the type of website. We have considered website types like E-commerce, Magazines, Blogs, Social Media, News and Educational sites for extraction.
percentage. The extraction of webpages depends on many factors as shown in Table 3. Most of the webpages don’t have standard data formats and tags, hence we cannot generalize extraction of webpages on a large scale. In Blog and News like webpages the data and features are updated on daily or hourly basis, making it difficult for extraction due to the internet traffic. Also, when web crawler tries to connect to the server again and again it leads to DDOS (Distributed Denial of Service) error to website. Some tools like scrape shield and scrapesentry distinguish between a human and a bot and prevent the scraping of data. There are also certain webpages with protected tags that cannot be extracted.

Figure 7. Precision Recall and F-score of different webpages

By using Depth First Search algorithm, this model performs retrieval of necessary data with an average precision of 5.4, recall of 7.45 and F-score 6.24. The figure 7 shows the Precision Recall and F-score of 60 different webpages. The API in Django helps to separate the Django templates on the client side and the database on the server side. The Django REST framework provides Database Migrations, URL routing in Python making it writable compared to PHP. It also provides security by hiding the website’s source code. XSS and CSRF attacks can be prevented along with protection against SQL injections and Clickjacking.

5. Conclusion

The paper presents a fully functional web application for the purpose of document parsing using PyTesseract, TextBlob and BeautifulSoup. Features like language translation and web crawling are provided under a single platform using Django framework. This application provides an effective solution to automate data entry and language translation by eliminating human error. It has a number of advantages over manual methods. Since hundreds of new web pages are added every day to the web directories. There is a need for an effective web crawler to deal with most of the online pages. Several web crawlers don’t have the flexibility to go and parse sites using URLs. Here, a new web crawler formula is built using the priority queue where URLs in web pages have been split into inter-domain and intra-domain connections. The developed algorithm weights these hyperlinks according to the type of link and preserves these links in the priority queue. Experimental findings clearly reveal that the algorithm develops a well-worn output against unreached, crawled web sites. In addition, the built algorithm has a strong ability to delete duplicate URLs. This paper can be further extended to perform many translation operations such as advanced pre-processing operations, detection of layout, routing and QR code detection.

6. References

[1] XCD. *The cost, and likelihood, of inaccuracy in manual data handling* | XCD. [Accessed 24 December. 2020]. Available at: https://www.peoplexcd.com/insights/the-cost-and-likelihood-of-inaccuracy-in-manual-data-handling/.
[2] Statista. (2020). *Internet: most common languages online 2020* | Statista. [Accessed 02 January. 2021]. Available at: https://www.statista.com/statistics/262946/share-of-the-most-common-languages-on-the-internet/.

[3] Haripriya A, Kumari S and Babu C N. 2018, September. Location based real-time sentiment analysis of top trending event using hybrid approach. *International Conference on Advances in Computing, Communications and Informatics (ICACCI)*. pp 1052-57. IEEE.

[4] Palm R B, Laws F and Winther O. 2019, September. Attend, copy, parse end-to-end information extraction from documents. *International Conference on Document Analysis and Recognition (ICDAR)*. pp 329-336. IEEE.

[5] Sage C, Aussem A, Elghazel H, Eglin V and Espinas J. 2019, September. Recurrent neural network approach for table field extraction in business documents. *International Conference on Document Analysis and Recognition (ICDAR)*. pp 1308-13. IEEE.

[6] Patel A and Schmidt N. 2011. Application of structured document parsing to focused web crawling. *Computer Standards & Interfaces*. 33. pp 325-331.

[7] Ribeiro M R, Júlio D, Abélha V, Abélha A and Machado J. 2019, July. A comparative study of optical character recognition in health information system. *International Conference in Engineering Applications (ICEA)*. pp 1-5. IEEE.

[8] Udapure T V, Kale R D and Dharmik R C. 2014. Study of web crawler and its different types. *The International Organization of Scientific Research-Journal of Computer Engineering (IOSR-JCE)*. 16. pp 01-05.

[9] Nakashe S M and Kolhe K R. 2018, August. Smart approach to crawl web interfaces using a two stage framework of crawler. *Fourth International Conference on Computing Communication Control and Automation (ICCUBEA)*. pp 1-6. IEEE.

[10] Mahar B and Jha C K. 2015. A comparative study on web crawling for searching hidden web. *International Journal of Computer Science and Information Technologies (IJCSIT)*. 6.

[11] Kumar M and Bhatia R. 2016, March. Design of a mobile web crawler for hidden Web. *3rd International Conference on Recent Advances in Information Technology (RAIT)*. pp 186-190. IEEE.

[12] Trivedi H P, Daxini G N, Oswal J A, Gor V D and Mali S. 2014. An approach to design personalized focused crawler. *International Journal of Computer Science and Engineering*, 2. pp 144-147.

[13] Kale P and Dahiwale P. 2015. Design and implementation of focused web crawler using genetic algorithm. *International Journal of Advanced Engineering and Global Technology*. 3. pp 1392-98.

[14] Sahu M B and Bharne S. 2016, March. A survey on various kinds of web crawlers and intelligent crawler. *International Journal of Scientific Engineering and Applied Science (IJSEAS)*. 2.

[15] Kalmukov Y and Valova I. 2019, May. Design and development of an automated web crawler used for building image databases. *42nd International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*. pp 1553-58. IEEE.