1. **Introduction**

One of the most popular and commonly used file formats is Portable Document Format, or PDF, which has the capability to store a plethora of crucial information for organizations, businesses, and institutions. In today's rapidly evolving digital economy, PDFs are becoming a sustainable alternative to paper, allowing users to share, print, and view files across different platforms. Despite their reliability for storing and formatting data, it remains challenging to scrape, parse, and extract data from PDF files as they are versatile. In our project, we are dealing with semi-structured data that doesn't transform accurately into columns or rows, due to the nature of the pdf data files.

This project aims to analyze the Emergency Food Security Fund (EFSF) data made available during the pandemic by the Canadian Federal Government. The EFSF was announced in 2020 to aid local food organizations with nearly $100 million in funds to deal with the food crisis that arose amidst the COVID-19 pandemic. Food banks were unable to deal with the huge demand that arose leading to food insecurity for numerous Canadians to eat nutritious food. Agriculture and Agri-Food Canada partnered with 6 national organizations namely Salvation Army, Breakfast Club of Canada, Second Harvest, la Tablee des Chefs, Community Food Centres Canada and Food Banks Canada. These organizations funded a total of 3038 projects including purchase of fridges, food storage units, delivery services partnering with community centres, local churches, food banks and other charitable organizations. 20% of local partners were also surveyed to understand the perspective of the underlying organization. The shared information was analyzed to create performance reports and program results to report to the Parliament.

The key objective of the project is to extract these valuable data points from the available unstructured PDFs and give the researchers at University of Toronto, the required data in an efficient and reliable manner into Comma separated values (CSV) spreadsheets. Our goal is to leverage state-of-the-art artificial intelligence (AI) technologies and data science solutions to develop solutions that can mine valuable insights from unstructured sources like PDFs and scanned images.  In addition to saving time in capturing data, these techniques make it possible to generate high-quality and insightful statistics in a timely manner enabling researchers to spend more time performing meaningful analyses.

An issue, however, that was highlighted as the pandemic progressed was the theft of funds across the world. The funds raised were alleged to have been stolen by various political bureaucrats, non-profit organizations, and private individuals across North America**[1, 2].** The analysis of food security data will provide insight into the responses by recipients of COVID funds. This will provide insight into the type of investment, the number of people served, the funding received by each recipient across different regions. By using this information, organizations can prevent fraud and determine how funds can be allocated in the future in the best manner.

At the outset of our report, we perform literature survey on the associated topics to parse PDFs. The Literature review section discusses how versatile PDF formats are and how they match different standards. In the methodology section, an overview of the data and PDFs we are dealing with has been provided along with the challenges associated with extracting data from variable PDF formats. The model development section details the mode of our approach in tackling the extraction process of unstructured data from PDFs and storing them into CSV files. The last section of the report details the peer reviewed articles, papers and GitHub repositories referred to develop the solution to parse the PDFs.

Terminology and key model parameters are document parsing, optical character recognition conversion, unstructured data, non-relational databases, sentiment analysis, text mining, data visualization, and financial analytics. The data present is in both structured and unstructured formats. Currently, to deal with unstructured data in the survey response, the best approach is extracting the textual information using Microsoft Excel software with its ‘Data-from-Picture’ option into a CSV file.  For the structured data, the information will be parsed using easyOCR and Tesseract to convert the information into raw text. The parsing code will vary for each page of the PDF with different column attributes associated. We have identified 4 different types of page formats for extraction. The parsing code for each format of page will be scaled to run on all pages. A higher computing resource is required and data science services by cloud providers such as Google, Oracle, Azure, and Amazon were explored before settling on using Google Collaboratory. Once this has been accomplished, the structured data is stored into two csv files containing structured data of each PDF after a writeback via python. The files are validated using random sampling measures, manually verifying key data points for ApprovedFunding\_AAFC and PeopleServed columns and by using the PageIndex column as a primary key or Index to verify that all pages have been traversed in the extraction process.

The unstructured data will be similarly stored into a custom spreadsheet with data about the extracted text and corresponding sentiment score for the feedback provided in the Interim file. The information extracted is analyzed quantitatively to understand how the funds have been used by different leads, descriptive analysis of survey data, and sentiment analysis on the feedback shared by participants. Exploratory Data Analysis (EDA) is carried out to find any discrepancies with data that can be shared for the welfare of the Emergency Food Security Fund Program.

1. **Literature Review**

The project involves the conversion of food security funds data from PDF files via OCR and storing them in databases for further analysis. Even though PDF is one of the most used formats for document storage, it is not standardized. There are many types of PDFs, from computer-generated articles to scanned copies of old documents, all of which affect how well a set of instructions from software can "read" the text embedded in a PDF.

There is a constant evolution and change in the technical specifications and requirements of different PDF formats. These include a range of Adobe-owned intellectual property as well as those standardized by the International Organization for Standardization (ISO**)[3].** Depending on how one plans to use, share, and keep your document, the best suitable standard will vary. Eight PDF standards exist, six of which are ISO Standards (PDF, PDF/A, PDF/E, PDF/X, PDF/VT, PDF/UA) and two are from different organisations (PAdES, PDF Healthcare) **[4].** Because each standard has a distinct function, saving a file to the wrong standard can cause issues and alter the desired functionality of usage.

Based on their structure, Statistics Canada acknowledges three most common types of PDFs as: Structured PDFs, Text-based unstructured PDFs, and Scanned Unstructured PDFs **[5].** The basic structure and arrangement of the documents in structured PDFs remain constant throughout the dataset. To automate the extraction of data in structured PDFs and its organization into tabular form, segments must be created and assigned labels **[6].** A PDF document is text-based if you can click and drag to retrieve text in a PDF viewer. While layout-free text can be parsed easily from these documents, it can be quite difficult to do so in a layout or context-based approach.

Pages are sometimes scanned as images and converted to PDFs, and these have their own peculiarities. Scanned PDF documents contain information in multiple shapes and sizes. Additional steps help to localize text components and perform optical character recognition to extract textual tokens **[7].** Once the PDF is converted to text and the location for these tokens are identified, you can deploy similar methods used for text-based PDFs to extract information. There is no common solution that is applicable for all scenarios.

Occasionally, PDF files have font data but not encoded data. This is often due to a particular version of Ghost script **[8]**, but this also applies to other PDF files, and you can do this on purpose to prevent users from reading private material. The problem with these PDF files is that they do not contain text; instead, they are lists of glyphs **[9]**, each with instructions on how to view it and the characters it represents which is its encoding. Generally, the encoding is a built-in pattern (e.g., MAC encoding or Windows encoding) **[10,11]**. One of the critical challenges in the project is to decode and transform data points that are Win ANSI encoded and extract textual information.

There are many packages that can extract text from PDF files, including Tabula, PyPDF2, PDFPlumber, and Textract etc. which have their specific advantages and disadvantages depending on the use-case. Some packages are designed to work with table specific formats, while others are better for parsing text. When dealing with real-world data, the data is often not organized and lacks a particular format. We must transform unstructured data into a structured format before it can be reviewed. As the shared project data files are semi-structured, the aforementioned Python libraries cannot be used in our situation due to their inconsistent outcomes. Also, it is significant that the extracted text must also be as accurate as possible, and as there is a need to handle hundreds of pages, it's vital that our procedure be expeditious which can be efficiently handled by Optical Character Recognition (OCR) to an extent. Optical Character Recognition (OCR) **[12]** has been one of the most popular tasks in Computer Vision. OCR is used by nearly all applications that need to extract text from images, which include data entry for business, automated passport and licence plate recognition **[13]**, rapid document verification, Internet of Things (IoT) applications, task automation, and others, demonstrating its popularity.

OCR works by converting information present in images to a text format that can be used to automate data processing in files with images **[10]**. The images are cleaned by aligning the document appropriately and de-speckling the image while correcting the shapes in the image. There are mainly two types of OCR processes namely pattern matching and feature extraction. During pattern matching, a character image called Glyph is compared with an existing image of a glyph **[10].** The limitation of this technique is that if the font and scale are different between the new Glyph and the existing glyph then the pattern matching will fail. Feature extraction **[10]** works by deconstructing the glyph and breaking it down into lines, intersections, and loops and finding the closest match among the stored glyphs. The image is later converted into a textual format using the above processes.

During the postprocessing phase, the extracted textual information is stored in a format like PDF. Modern OCR systems use Intelligent character recognition technology using neural networks to process the images faster. Jain et Al. **[14]** explore different OCR toolsets and concluded that ABBYY is the best for high-quality images while Tesseract 4.0 is the most popular openly available software for an end-to-end lifecycle of data conversion. It was also noted that Rossetta is popular when dealing with a large volume of images. Moritz et Al**. [15]** discuss methods like OCRmyPDF and ImageMagick which can be used to merge PDF files and how one can perform text mining using topic modeling to analyze the corpus with the open-source ILO repository index. This will help determine word probability for the associated topic and export them for further analysis.

After performing OCR conversion and topic modeling, sentiment analysis can be performed on the data to understand the Lead sentiment behind the Covid19 food program and gain insights based on their opinions. Yu et Al. **[16]** discuss the use of descriptive analytics to understand the sentiment during the pandemic using Twitter. They analyze the tweet's percentage, and keywords, and calculate a weighted average sentiment score using the VADER lexicon across the timeline of collected tweets. It was noted that the analysis was limited in the context of real-time data and further measures to improve the architecture were required to get a more accurate representation of public sentiment. Thus, the overall scope of the project currently is to use the recommended OCR toolkits to convert the PDF information into textual information and analyze them for insights.

1. **Methodology**
2. **Dataset Overview**

The two unstructured PDFs given to us are: ATIP Final Interim package and Interim release package. They consist of 2354 and 669 pages respectively with a total memory size of PDFs coming close to 70MB. The PDFs were obtained from the Canadian government and shared by researchers from the University of Toronto collaborating with Trent University’s AMOD department to extract information for further analysis. These PDFs demonstrate the allocation of funds and their respective management by local food banks and other food relief organizations as a part of the Emergency Food Security fund (EFSF) program, a federal investment fund to support Canadians suffering from food insecurity as a result of the COVID-19 pandemic. The attributes of the data include the recipient’s name and type, their community, town/city, and province indexed by a project number and lead. It also presents the approved funding amount, date of approval/rejection, date paid and type of the investment with estimated number of people served. The data also contains a ‘Notes’ column for the recipients to add any feedback and comments. The unstructured data contains responses with checkboxes in bilingual form (English and French), survey response text and bar graphs.

Upon minor inspection of data in the PDF, one could find an important pattern: text at the intersection of column names is stacked and shifted so that it could hardly be recognized as the additional feature of the same column name. Nonetheless, any data that does not fit rightly into a column or a row is widely considered unstructured, we can categorize data from these given PDFs as semi-structured.

Some key challenges and limitations in the dataset noted from the PDF are highlighted below:

* Format in which the attributes are stored in the page varies from page 1544 of the first PDF – ATIP Final Interim package – Figure 1&2.
* Some Personal Identifiable Information (PII) details are not redacted and are treated as null.
* Interim release package PDF – The first 45 pages consist of reviews and results as shown in figure 3 below, data points from page 46 onward are in the semi-structured format.

Graphical user interface, table

Description automatically generated

***Figure 1: Format A of PDF’s attributes***

Table

Description automatically generated with medium confidence

**Figure 2: Format B of PDF’s attributes**

**Text

Description automatically generated**

**Figure 3: A sample from PDF showing unstructured data**

1. **Model Development**

To parse the PDFs, we use open-source OCR (Optical Character Recognition) engines and python as the programming language of choice. After exhaustive testing, optimal OCR engines were determined as EasyOCR and Tesseract with OpenCV aiding for image processing techniques. The other packages that were considered were PaddleOCR, pyPD2, PDFQuery, Excalibur, PDFminer which were not able to parse the text due to the nature of the data file as the document is image-based rather than text-based for parsing. Tabula, another pdf parser package that is used in python was able to parse some of the data using inbuilt functions, however due to the image-based nature of the file, the tabula functions were unable to parse some of the records. The tabula functions were modified using custom code to run in batches of code, however, it was unable to parse all the information.

Tesseract OCR **[17]** was developed by Hewlett Packard and was originally written in C. Tesseract was declared as an open-source software in the year 2005 following which Google funded its further development until 2018. The latest version of Tesseract is based on Long Short-Term Memory based neural networks and can detect more than 100 languages without external model training. It has UTF-8 support including various formats such as PNG, JPEG, PDF, hOCR. Tesseract was found to parse text in PDFs better than other open-source libraries tested, however it lacked the customizability of arguments such as modifying bounding boxes for python as it was mainly built for use with C++. EasyOCR **[18]** was developed by Jaded AI as their first open-source project in the year 2020. It is a python module based on deep learning which can detect dense and natural scene text in the document.

Diagram

Description automatically generated

**Figure 4: EasyOCR Framework**

Easy OCR **[19]** makes use of the CRAFT algorithm and pre-trained model created by Baek et Al. **[20]** and convolutional recurrent neural networks for recognizing features in the text and extracting them. It also uses the Connectionist Temporal Classification algorithm **[21]** by Harald Scheidal and Eduoard Belval. It also supports more than 80 languages like Tesseract and has a simple API built for the python Ecosystem. There are some cases where it was observed that certain characters in PDF like ‘0’ are recognized better by Tesseract than easyOCR. The output of the easy OCR with bounding boxes looks like ([685, 273], [935, 273], [935, 309], [685, 309]],'School/Educational',0.7561491418543352). The first parameter of the API response provides the coordinates of bounding boxes as shown in figure 5 below. The second parameter returns the character recognized inside the bounding box. The final parameter returns the confidence behind the detected text. Thus, enabling us to specify a threshold for character detection to detect information that has not been parsed accurately for data validation **[22]**.

Graphical user interface, application, table

Description automatically generated

**Figure 5: Bounding boxes over the text using easyOCR**

1. **Data Extraction**

During the data extraction process, the files were divided into three main subcategories due to the same column headers on alternative pages. For File A, the data for pages until 1544 and data for pages from 1545 until the end of the file are segmented into two subsets as they follow a particular pattern of odd and even pages making up a single whole page. For File B, data for first 45 pages and data for pages from 46 until the end of the file are segmented. Each page of the PDF file was converted into an image using PDF2Image library. We can manually specify the dpi of each image, and also pad each image with a number sequentially. To ensure sharper text for processing, OpenCV was used to enhance the quality of the individual images by pre-processing them to grayscale, and all images were normalized to a constant aspect ratio.

The widths of the column headers remained the same across each segment and were computed for extraction. The heights, however, varied as the number of data records vary across each page. The varying heights are computed by taking the height of the first row and the difference in height between each row until it reaches the end of the page. While Pytesseract was used to get the data from the image and convert it to an editable text format, EasyOCR was found to be the best tool for reading the data. The masked Personal Identifiable Information(PII) data and empty rows are stored as null values. The page numbers are also added at the end to indicate the page of the PDF file from which it was extracted and the file.

Chart

Description automatically generated with low confidence

***Figure 6: Sample Snippet of Extracted CSV data of File A***

The final output after parsing all the raw text is stored onto an array using high performance cloud computing resources. A CSV writeback of the array is performed to store the information into spreadsheets that allow the researchers to use the data immediately using Ms Excel without the need of installation any external tools or setting up databases. The spreadsheets are uploaded to perform text mining operations, exploratory data analysis, descriptive analysis, and sentiment analysis to gain data driven insights for the researchers.

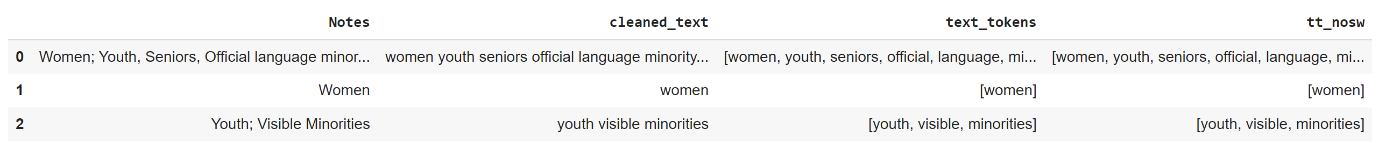
1. **Results**
2. **Data Pre-Processing**

The data is preprocessed before further analysis using the following steps:

* The PII data columns are removed.
* The text type columns such as description, and notes are removed.
* Non-numeric values from Measure columns such as ApprovedFunding\_AAFC and NumberPeopleServed are removed.
* NaN and blank rows are removed from columns with numeric attributes.
* Irregular extracted data formats such as ’/20/2022’ are replaced with neighboring values.
* The data format for dateapproved\_rejected and date\_paid are converted into DD-MM-YYYY format.

The Feedback Text of File B (Interim Package) and File A Notes column is used to perform word frequency and sentiment analysis and is cleaned using the following steps:

* The null rows are dropped. Empty rows are replaced with Nan.
* Numbers, punctuation, and double spaces are dropped.
* The text is tokenized using NLTK.
* The stopwords such as ‘in’, and ‘yours’ are removed as they do not add unique information to the text.



***Figure 7: Cleaned and tokenized data of File A***

1. **Analysis**

The data was visualized using Oracle Data Visualization and a new column called funding efficiency was created which was NumberPeopleServed/ApprovedFunding\_AAFC. Descriptive Analysis of AAFC Approved Funding Amount and the number of people served by Lead and Indigenous Groups. Funding Efficiency is computed as People served by Approved Funding Amount to understand the efficiency in distribution of funds. Example If a Lead serves 10 people with 100$ and another serves 20 with 100$ then FE would be 0.1 and 0.2. This shows that latter has utilized the funds effectively. It is important to note that this is a suggested metric as there were incomplete data for some rows of AAFC Approved Funding and Number of People Served as well as some unknown lead names. These metrics are plotted below in Figure 7 and a treemap visual consisting of the top 5 Indigenous Groups that received AAFC Approved Funding are also shown.

Chart

Description automatically generated

***Figure 8: Data Visualization of File A data***

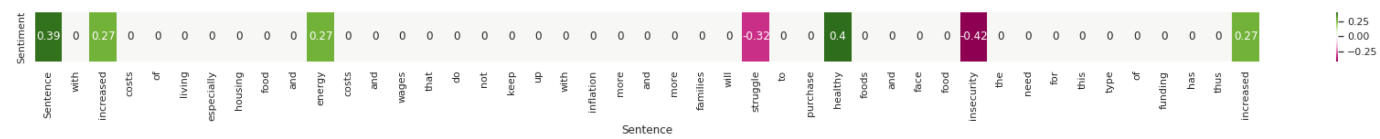
The word cloud highlights the key words used in the PDFs, the largest and most prominent words in the cloud are “youth”, “women”, “seniors” and “income” which reflects the means in which funds are distributed.

Text

Description automatically generated

***Figure 9: Word Cloud of File A Notes data***

Vader based Sentiment Analysis [23] of feedback form with English and French text to understand opinion of Leads for the Emergency Fund Security Project. Vader-fr, a French version of the was used to analyze French text.



***Figure 10: Sentiment Scoring Sample Sentence Snippet***

The sentiment score corresponds to the compound Vader score, a measure that is computed as an aggregated weighted average of the positive, negative, and neutral scores for the question numbers from the feedback obtained from the leads in the Interim file. Data across 5 different leads and the type of funds namely Local Infrastructure Fund and Emergency Funds.

Chart, bar chart

Description automatically generated

***Figure 11: Sentiment Score across LeadName,FundType based on Question of Feedback Form***

The Approved Funding Status in Interim File B is classified using a Logistic Regression Algorithm. The Leadname, Town\_City\_Community Name, AAFC Approved Funding and Investment Type are used as input parameters with the funding status acting as the prediction variable. As the data is imbalanced the samples of ‘Approved’ Funding status are downsampled to ensure accurate model training. The resulting model classification accuracy was 83.72.

**Table

Description automatically generated with medium confidence**

***Figure 12: Sentiment Score across LeadName, FundType based on Question of Feedback Form***

1. **Conclusion**

Brief a little more about analysis: Why are the results important?

The key takeaways from the analysis are that the funding efficiency can be used to understand utilization of funds by leads and recipients, sentiment score can be used to understand feedback and the classification algorithm can be employed to understand the chances of an Investment Type being approved. Investment Type are used as input parameters with the funding status acting as the prediction variable. As the data is imbalanced the samples of ‘Approved’ Funding status are down sampled to ensure accurate model training. The resulting model classification accuracy was 83.72.

In conclusion, the successful extraction of tabular data from unstructured big data PDFs is a challenging task, but it can be accomplished with the right combination of tools and techniques. In this project, we utilized EasyOCR and Pytesseract for performing OCR, PDF2Image for PDF page to image conversion, and OpenCV for image pre-processing. We also implemented a method to combine odd and even pages to form a single page of data. The segmented large PDFs were processed, and the extracted data was stored in CSV format. This approach provides a way to efficiently extract tabular data from large unstructured PDFs, which can save time and improve productivity in data analysis and research. Overall, this project demonstrates the potential of using OCR and image processing techniques to extract structured data from unstructured PDFs.

1. **References**
2. CBC/Radio Canada. (2021, September 22). *3rd person charged in alleged $11M scheme to defraud ontario covid-19 relief fund | CBC News*. CBCnews. Retrieved February 7, 2023, from https://www.cbc.ca/news/canada/toronto/toronto-covid-relief-fund-1.6185794
3. Romm, T. (2022, September 21). *U.S. charges 'brazen' theft of $250 million from Pandemic Food Program*. The Washington Post. Retrieved February 7, 2023, from <https://www.washingtonpost.com/business/2022/09/20/pandemic-fraud-food-children-doj/>
4. *ISO 32000-2:2020*. ISO. (2020, December 11). Retrieved February 7, 2023, from https://www.iso.org/standard/75839.html
5. Wyatt, P. (2005, August 25). *ISO 19005 (PDF/A)*. PDF Association. Retrieved February 7, 2023, from https://www.pdfa.org/resource/iso-19005-pdfa
6. Government of Canada, S. C. (2022, January 21). *Document intelligence: The art of PDF information extraction*. Government of Canada, Statistics Canada. Retrieved February 7, 2023, from https://www.statcan.gc.ca/en/data-science/network/pdf-extraction
7. Rajan, K., Brinkhaus, H. O., Sorokina, M., Zielesny, A., & Steinbeck, C. (2021). Decimer-segmentation: Automated extraction of chemical structure depictions from scientific literature. *Journal of Cheminformatics*, *13*(1). https://doi.org/10.1186/s13321-021-00496-1
8. Government of Canada, S. C. (2023, February 10). *Data Science Network for the Federal Public Service (DSNFPS)*. Government of Canada, Statistics Canada. Retrieved February 7, 2023, from https://www.statcan.gc.ca/en/data-science/network
9. How to use ghostscript. (n.d.). Retrieved February 8, 2023, from https://ghostscript.com/docs/9.54.0/Use.htm
10. *Unicode character encoding model*. UTR#17: Unicode Character Encoding Model. (n.d.). Retrieved February 8, 2023, from http://unicode.org/reports/tr17/#CharactersVsGlyphs
11. *Text*. (n.d.). Retrieved February 8, 2023, from https://developer.apple.com/library/archive/documentation/mac/pdf/Text.pdf
12. *PDF File settings*. IBM. (n.d.). Retrieved February 8, 2023, from https://www.ibm.com/docs/en/cognos-analytics/10.2.2?topic=SSEP7J\_10.2.2%2Fcom.ibm.swg.ba.cognos.ug\_cra.10.2.2.doc%2Fc\_pdffilesettings.htm
13. *What is OCR*. Amazon. (2010). Retrieved February 8, 2023, from https://aws.amazon.com/what-is/ocr/
14. Awalgaonkar, N., Bartakke, P., & Chaugule, R. (2021). Automatic License Plate Recognition System using SSD. *2021 International Symposium of Asian Control Association on Intelligent Robotics and Industrial Automation (IRIA)*. https://doi.org/10.1109/iria53009.2021.9588707
15. Jain, Taneja, K., & Taneja, H. (2021). Which OCR toolset is good and why? A comparative study. Kuwait Journal of Science, 48(2), 1–12. <https://doi.org/10.48129/KJS.V48I2.9589>
16. Mähr, M. (2020). Working with batches of PDF Files. *Programming Historian*, (9). https://doi.org/10.46430/phen0088
17. Yu, Ferreira, M. D., & Paulovich, F. V. (2021). Senti-COVID19: An Interactive Visual Analytics System for Detecting Public Sentiment and Insights Regarding COVID-19 From Social Media. IEEE Access, 9, 126684–126697. <https://doi.org/10.1109/ACCESS.2021.3111833>
18. Tesseract-Ocr. (n.d.). *Tesseract-OCR/tesseract: Tesseract open source OCR engine (main repository)*. GitHub. Retrieved February 8, 2023, from https://github.com/tesseract-ocr/tesseract
19. *Demo*. Jaided AI: EasyOCR demo. (n.d.). Retrieved February 9, 2023, from https://www.jaided.ai/easyocr/
20. JaidedAI. (n.d.). *Jaidedai/EasyOCR: Ready-to-use OCR with 80+ supported languages and all popular writing scripts including Latin, Chinese, Arabic, Devanagari, cyrillic and etc..* GitHub. Retrieved February 9, 2023, from https://github.com/JaidedAI/EasyOCR
21. Baek, Y., Lee, B., Han, D., Yun, S., & Lee, H. (n.d.). Character Region Awareness for Text Detection. <https://doi.org/10.48550/arXiv.1904.01941>
22. Shi, B., Bai, X., & Yao, C. (n.d.). An End-to-End Trainable Neural Network for Image-based Sequence Recognition and Its Application to Scene Text Recognition. https://doi.org/10.48550/arXiv.1507.05717
23. Belval. (n.d.). *Belval/textrecognitiondatagenerator: A Synthetic Data Generator for text recognition*. GitHub. Retrieved February 9, 2023, from https://github.com/Belval/TextRecognitionDataGenerator
24. Hutto, C.J. (2014). vaderSentiment: VADER Sentiment Analysis. GitHub Repository. Retrieved from <https://github.com/cjhutto/vaderSentiment>