JPMorgan Chase & Co.

Columbia University Data Science Institute

Spring 2022 Capstone Project
Table Extraction via Eye Gaze Tracking

Final Report

Yibai Liu (yl4616), Yijia Jin (yj2682), Yeqi Zhang (yz3975),
Shihang Wang (sw3275), Yinqiu Feng (yf2579)

May 13, 2022

Abstract

We have designed an end-to-end table extraction pipeline for JP Morgan & Chase that automatically detects tables within an image, utilizes eye tracking to locate the Area of Interest, and returns cropped table of attention as well as texts inside. Related table extraction solutions cannot perceive people's attention, and we extended the task to distinguish and extract the table of interest. The resulting solution pygazeTE combines eye gaze tracking with a YOLOv5 object detection model and a Tesseract text recognition model to perform the tasks stated above. Models were trained and validated on 2,000 images from IBM's financial document dataset FinTabNet, and the solution was wrapped to a Python package `pygazeTE`.

1 Introduction

This Capstone project was sponsored by the AI Research team of JP Morgan & Chase. The amount of data being collected is drastically increasing day-by-day with growing numbers of applications, software, and online platforms. Fast and accurate extraction of tables and figures is critical to enable more efficient workflow and more smooth knowledge sharing for numerous teams throughout the business. Table extraction from images is a more complex task than from PDF files, and Computer Vision (CV) performs a crucial role in this object detection related work. A use case scenario during a business meeting can be that an employer wishes to obtain a table from a comprehensive document projected on the screen, and this brought in the Eye Gaze Tracking technology, which distinguishes people's attention via fixations.

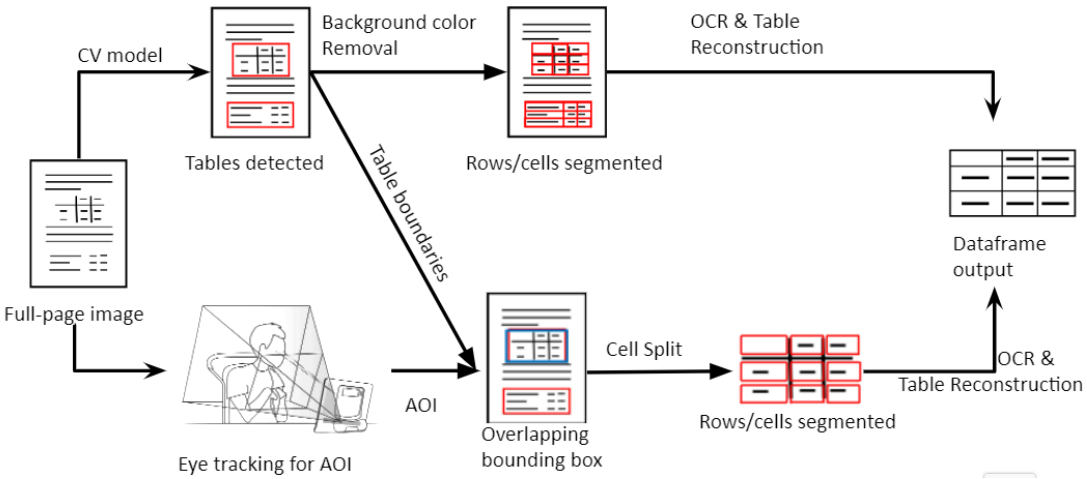


Figure 1: A flow chart of pygazeTE pipeline

Taking a full-page image input as shown in Figure 1, the objective of this Capstone Project was to utilize Eye Gaze technology and design a system to automatically extract tables from an image-form document leveraging CV and Optical Character Recognition (OCR). Automating this process allows an employee to extract the data in the table(s) of interest from a report by simply looking at the tables, and it will be more efficient compared to screenshot capturing and manual data entry.

2 Related Work

2.1 Table Extraction

Previous works on table extraction falls into two subtasks, (1) table detection [1]; (2) table structure decomposition [2]. The table detection task is a ground application of object detection, which mainly focuses on identifying the table borderlines from a given document. Then the table structure decomposition task identifies the components of the original tables and reconstructs the tables by stacking those components. The decomposition process includes the accurate identification of rows and columns, as well as the correct allocation of the table cells.

However, table structure decomposition still remains a difficult problem to solve, not only because of the changes of table layout and style, but also because of the changes of page layout and noise pollution [3]. At present, a lot of research has been done on table structure recognition, most of which are based on heuristic technology combined with OCR to figure out the layout features of the table [4]. OCR, defined as the electronic or mechanical conversion of images of typed, handwritten, or printed text into machine-encoded text [5], on the other hand, also suffers from the failure of generalization because of the absence of meta-features or errors made by the OCR when there is a significant variance in table layouts and text organization [6].

Therefore, this project propose a two-step framework that leverages both table detection model (yolov5) and OCR techniques altogether to address the issues of OCR, which has significantly improve the model performance and generalizability.

2.2 Eye Tracking

More often than not, the extraction process cannot be fully automated, and there is instead an important amount of manual intervention [7]. In the meantime, eye movements have been shown to reflect a combination of influences of low level image properties, the observer's task, interest, and goals [8] On top of that, researchers proposed that the gaze of human annotators during a manual information extraction process could be exploited towards reducing the manual effort and automating the process, and it was also found that relevant areas in the document can be identified reliably through automatic fixation classification, and the obtained models generalize well to new subjects [9]. For instance, Beymer et al. [10] captures and analyes web reading behavior according to eye gaze tracking, Shanmuga et al. [11] extracted attentionally important objects from videos with the assistance of eye gaze tracking, Rigaud et al. [12] utilizaed eye gaze data for manga content recognition, etc.

Therefore, it is in principle possible to integrate the human eye gaze information in the table extraction analysis loop, making use of the scan path tracking to automatically extract the table content of the user's interest.

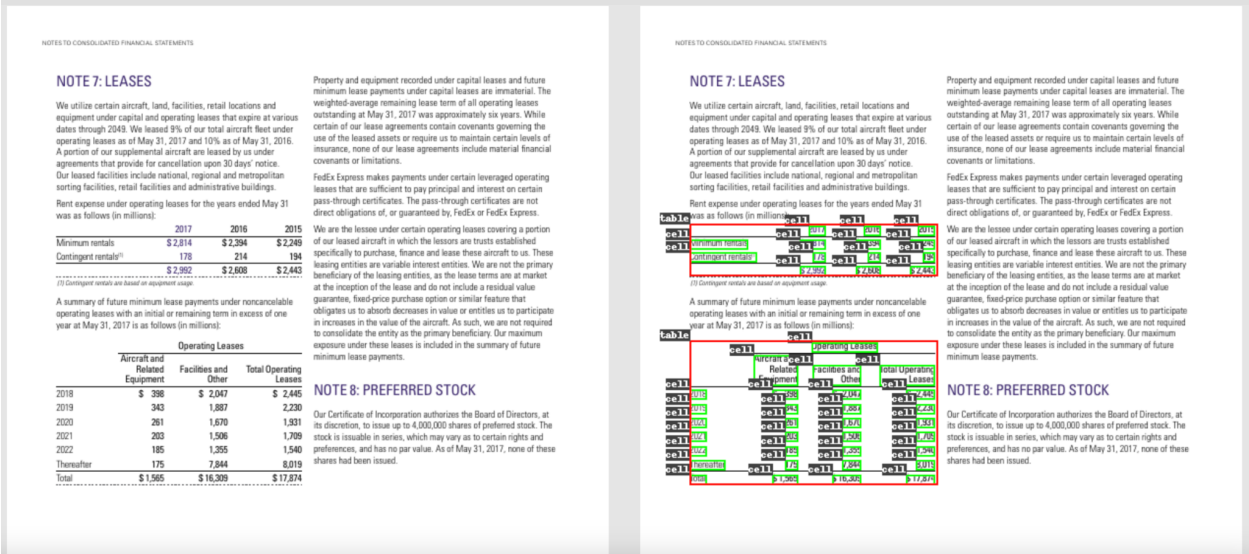
3 Datasets and Exploratory Data Analysis

3.1 Table Document Dataset

This project utilized IBM's open source dataset FinTabNet which includes 89,646 pages of financial document comprising 112,887 tables (some pages may contain more than one tables) [13].

This dataset covers complex tables from the annual earnings reports of S&P 500 companies. Like financial documents in practice, the financial tables in this dataset have diverse structures and color variations, with fewer graphical gridlines, tighter rows, and larger column gaps within each table, different from those more "standard" well-structured tables in scientific and government files. These variations brought in more challenges to OCR and Table Structure Reconstruction tasks.

All the documents are in PDF format, and annotations of cell structures and content are stored in a separate JSON file. We selected 2,000 documents from the huge dataset in our study and converted the PDF documents into images.



NOTES TO CONSOLIDATED FINANCIAL STATEMENTS

NOTE 7: LEASES

We utilize certain aircraft, land, facilities, retail locations and equipment under capital and operating leases that expire at various dates through 2049. We leased 9% of our total aircraft fleet under operating leases as of May 31, 2017 and 10% as of May 31, 2016. A portion of our supplemental aircraft are leased by us under agreements that provide for cancellation upon 30 days' notice. Our leased facilities include national, regional and metropolitan sorting facilities, retail facilities and administrative buildings.

Rent expense under operating leases for the years ended May 31 was as follows (in millions):

	2017	2016	2015
Minimum rentals	$ 2,814	$ 2,394	$ 2,249
Contingent rentals[1]	178	214	194
	$ 2,992	$ 2,608	$ 2,443

(1) Contingent rentals are based on equipment usage.

A summary of future minimum lease payments under noncancelable operating leases with an initial or remaining term in excess of one year at May 31, 2017 is as follows (in millions):

	Operating Leases		
	Aircraft and Related Equipment	Facilities and Other	Total Operating Leases
2018	$ 398	$ 2,047	$ 2,445
2019	343	1,887	2,230
2020	261	1,670	1,931
2021	203	1,506	1,709
2022	185	1,355	1,540
Thereafter	175	7,844	8,019
Total	$ 1,565	$ 16,309	$ 17,874

Property and equipment recorded under capital leases and future minimum lease payments under capital leases are immaterial. The weighted-average remaining lease term of all operating leases outstanding at May 31, 2017 was approximately six years. While certain of our lease agreements contain covenants governing the use of the leased assets or require us to maintain certain levels of insurance, none of our lease agreements include material financial covenants or limitations.

FedEx Express makes payments under certain leveraged operating leases that are sufficient to pay principal and interest on certain pass-through certificates. The pass-through certificates are not direct obligations of, or guaranteed by, FedEx or FedEx Express.

We are the lessee under certain operating leases covering a portion of our leased aircraft in which the lessors are trusts established specifically to purchase, finance and lease these aircraft to us. These leasing entities are variable interest entities. We are not the primary beneficiary of the leasing entities, as the lease terms are at market at the inception of the lease and do not include a residual value guarantee, fixed-price purchase option or similar feature that obligates us to absorb decreases in value or entities us to participate in increases in the value of the aircraft. As such, we are not required to consolidate the entity as the primary beneficiary. Our maximum exposure under these leases is included in the summary of future minimum lease payments.

NOTE 8: PREFERRED STOCK

Our Certificate of Incorporation authorizes the Board of Directors, at its discretion, to issue up to 4,000,000 shares of preferred stock. The stock is issuable in series, which may vary as to certain rights and preferences, and has no par value. As of May 31, 2017, none of these shares had been issued.

Figure 2: A preview of the dataset, the picture on the left shows the original PDF file, and the picture on the right illustrates cell annotations (actual annotations stored in JSON format)

3.2 Eye Gaze Data

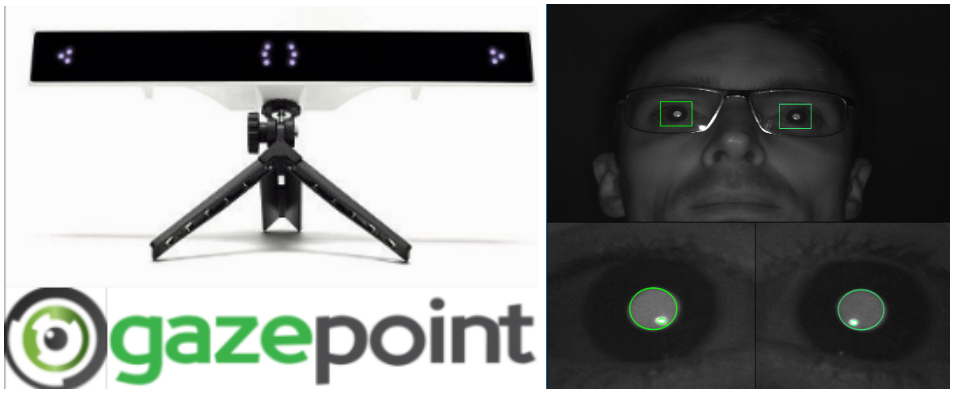


Figure 3: Illustrations of Gazepoint's GP3 Eye Tracker

In lack of eye tracking data with display of table documents, we designed a webcam-based eye tracking experiment in Python using Gazepoint's eye tracker GP3 and the open-source toolbox PyGaze, a Python wrapper for Gazepoint's OpenGaze API, to establish connection with the Gazepoint equipment. In an experiment trial, a participant was asked to complete device calibrations, and then a number of image-form documents were displayed on the screen. The participant was instructed to look for table(s) in the image and gaze at one table of interest for 10 seconds per image. During the display, the Gazepoint eye tracker recorded the exact positions of the pupils, the fixation duration and coordinates, the gaze direction, etc., which provides information about eye movements and areas of interest (AOIs). At the same time, a log file was generated recording the calibration results, timestamps for each image, and other events during the experiment.

3.3 Exploratory Data Analysis

To better understand the data, we carried out exploratory data analysis on the raw eye gaze data.

3.3.1 Missing Value Patterns

There was a consistency between columns FPOGID (the fixation point of gaze id number) and the record index, as shown in Figure 4 below. Since fixation was calculated based on both eyes, the missingness of these columns could be attributed to blinking as the equipment was unable to trace participants' eyes while they were closed. Based on FPOGID, it is possible to compute the frequency of blinking via regular intervals as follows:

Frequency of Blinking = Total Time / The Count of Blinking

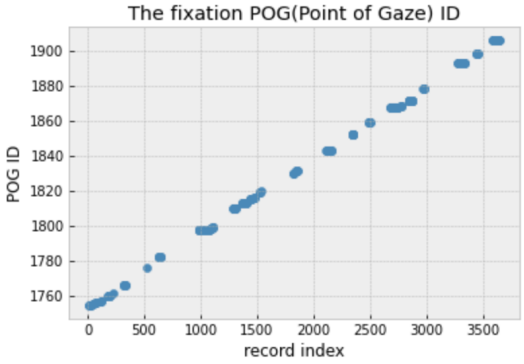


Figure 4: The Fixation POG ID Distribution

By experiment time interval and count of unique FPOGID, the average frequency of blinking, as calculated, was about 2.02 (averagely a participant blinks every 2 seconds).

3.3.2 Correlation Between Features

By plotting distributions of features, we observed that some features share a similar distribution, therefore we used heatmaps to explore correlations between all features and between some pairs of features.

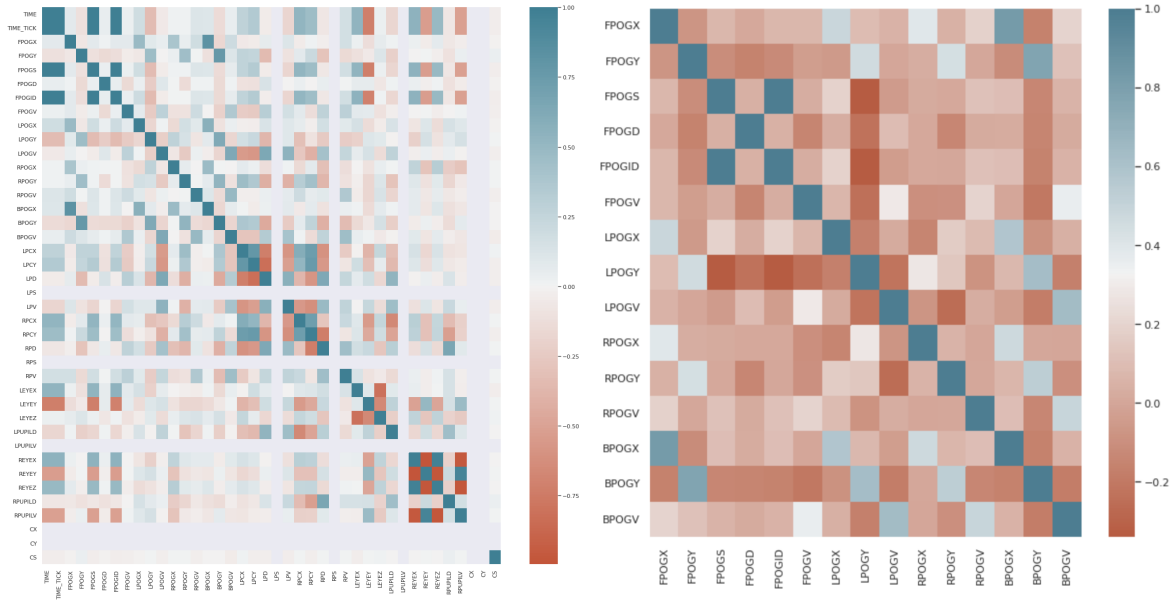


Figure 5: Heatmaps of correlations between all features (left) and between pairs of features (right)

From heatmaps (Figure 5), we observed that among all possible pairs, the highest positive correlations are found between FPOGX and BPOGX (the fixation and the best of x coordinates), and FPOGY and BPOGY(the fixation of and the best of y coordinate value), FPOGV and BPOGV (the fixation and the best validity flag). This shows that the best POG points are probably filtered from the fixation points (left-right average).

We also observed high correlations between left eye and right eye data, but from the data we collected, POG coordinates had a slightly higher correlation with left eye's data (LPOGX/Y) than with right eye's data (RPOGX/Y). Our guess was that the fixation calculation took account of the 3D data of eyes, and two eyes may trivially differ in their distance to the device.

3.3.3 Patterns of reading habits

As shown in Figure 6, heatmaps and scanpaths of fixation points were plotted. These visualization allowed us to quickly evaluate the gaze data quality and to see how eye movements vary with respect to time.

In the eye gaze experiments we collected data from different participants, and by visualizing scanpaths, we noticed the reading patterns vary from person to person, e.g. from top to bottom, from left to right, from numbers to texts, from results to details, or vice versa.

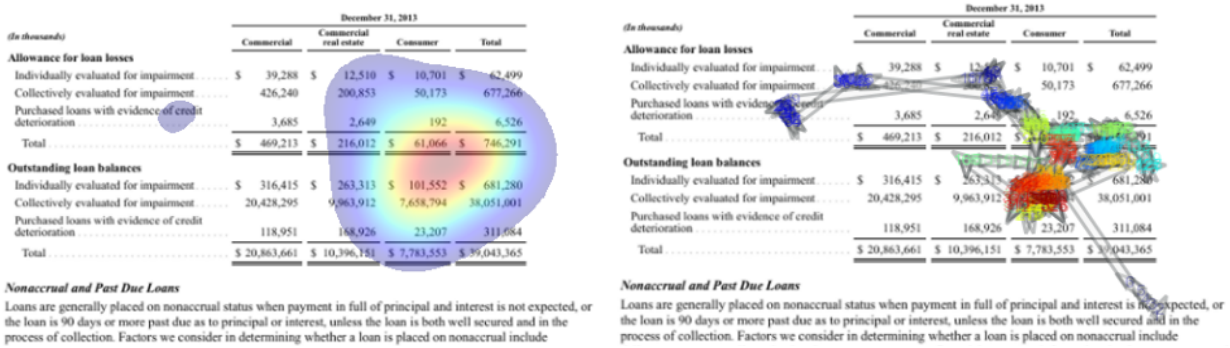


Figure 6: Heatmap and Scanpath of fixation points

3.3.4 Time Series Analysis

Since the movements of eye gazing were time dependent, which the position we are looking at was related to where we looked at around the last timestamp. This was more obvious when the participant was provided with clear targets, such as reading a paragraph or looking at figures and tables within the document. Therefore, the eye movements were hypothetically predictable given the history data reflecting where the eyes were gazing at.

To do so, we transformed the X, Y coordinates into [0,1] ratios, where the original X and Y coordinates were divided by the image width and image height, respectively. Then, we conducted time series analysis, taking the first image as an example, the eye movement sequences with respect to time were drawn below.

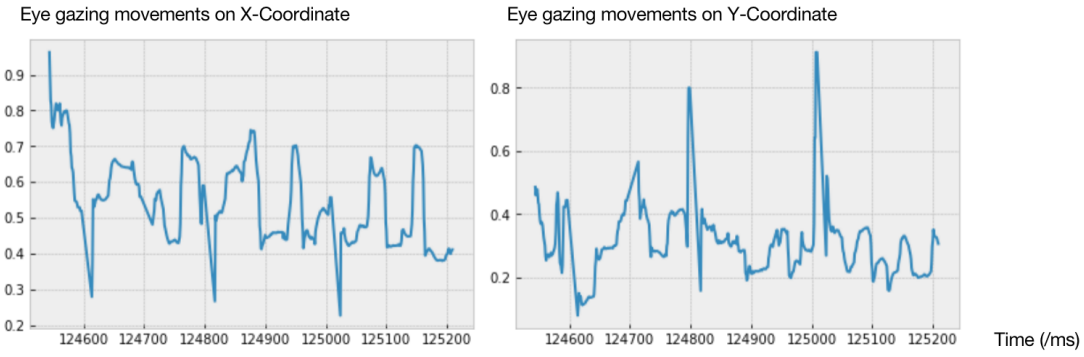


Figure 7: Eye gazing movements on X- and Y-coordinates

Then, we implemented the first-order difference on each of the sequences and found that the fluctuations were reduced a lot, as shown below:

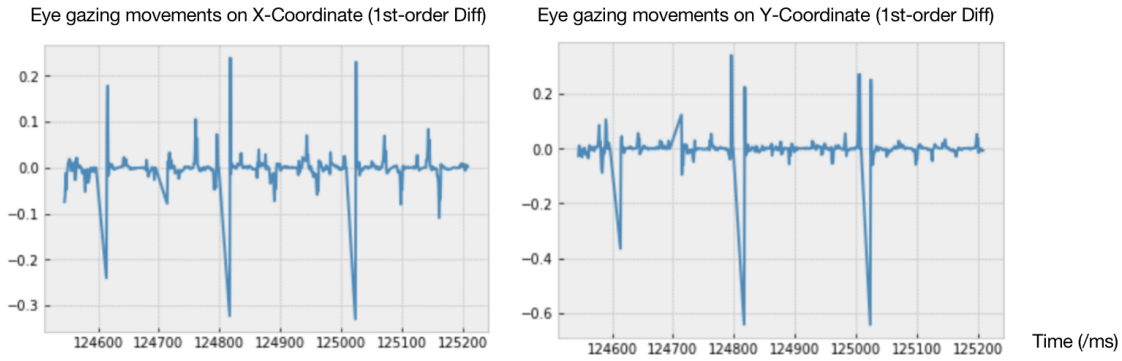


Figure 8: First-order derivatives of eye gazing movements on X- and Y-coordinates

After analyzing the autocorrelations, partial autocorrelations, as well as the ADF test and white noise test, we supposed that the ARIMA(0,1,1) model was appropriate for the eye gaze movements prediction on the first graph. We divided both sequences into two parts, the first 80% of the sequence were used as historical data (or training data), and the last 20% were used as testing data.

The ARIMA(0,1,1) model achieved BIC value of -2443.146 and RMSE value of 0.155 for X-coordinates, and BIC value of -1892.588 and RMSE value of 0.120 for Y-coordinates, which indicated that the ARIMA(0,1,1) modeled the sequences well and the eye gazing movements was actually predictable for our first image.

Similar calculations were also implemented on other images. To obtain more robust results, we also allowed the program to conduct a grid search on the ARIMA(p,1,q) parameters p and q, and the best model was then selected to report the evaluation metrics.

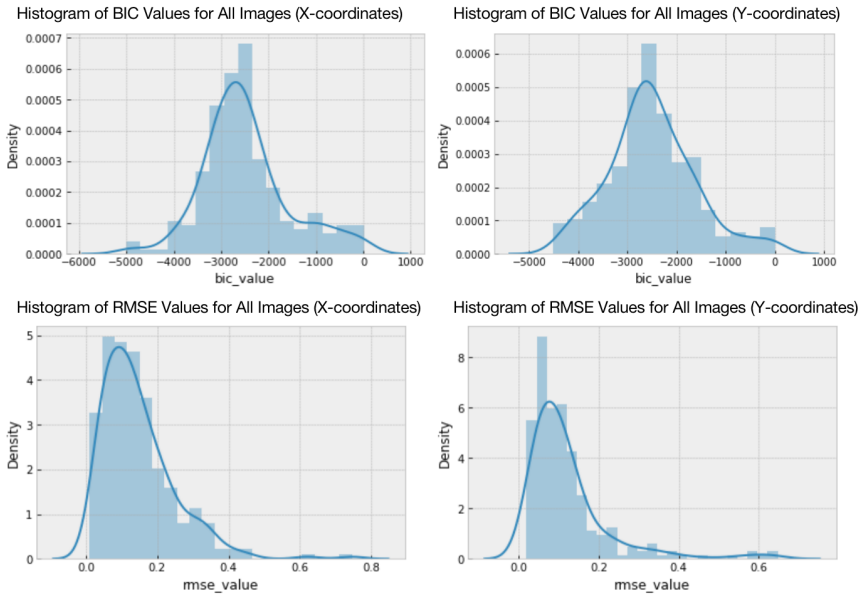


Figure 9: Histograms of BIC and RMSE values on X- and Y-coordinates, respectively

The results above indicated that the eye gazing movements in most of the images were predictable when the individuals are looking through their tables.

3.3.5 Dimensionality Reduction

To decrease dimensionality of the data, we tried methods including Linear Discriminant Analysis (LDA), Principal Component Analysis (PCA) , and t-SNE. LDA is a supervised technique that also achieves classification of the data simultaneously, whereas PCA is unsupervised and ignores the class label.

PCA results:

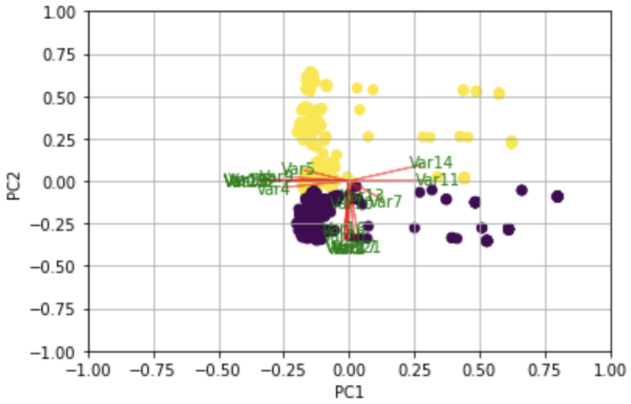


Figure 10: PCA Result Visualization

	0	1
0	PC0	LPOGV
1	PC1	FPOGID
2	PC2	RPOGX
3	PC3	FPOGY
4	PC4	LPCX
5	PC5	FPOGD
6	PC6	FPOGV
7	PC7	FPOGY

Table 1

The first principal component PC1 formed by PCA will account for maximum variation in the data. PC2 does the second best job in capturing maximum variation and so on.

LPOGV	The left fixation POG valid flag is 1 for valid and 0 for not valid
FPOGID	The fixation POG ID number
RPOGX	The X-coordinate of the right eye fixation POG, as a percentage of the screen width
RPOGY	The Y-coordinate of the right eye fixation POG, as a percentage of the screen height
LPCX	The X-coordinate of the left eye pupil in the camera image, as a percentage of the carama image size
FPOGD	the duration of the fixation POG in seconds
FPOGV	The FPOG valid flag is 1 for valid and 0 for not valid
FPOGY	Y-coordinate of fixation POG, as a percentage of the screen height

Table 2: Explanation of Important feature terms

From the eye tracking experiments, we discovered that gazing at the center of the table would provide more stable and accurate fixation points.

3.3.6 Clustering Analysis

In order to extract points or areas of interest, we conducted clustering analysis on the obtained experiment data, in hopes of forming cluster groups of gaze points which can be identified as multiple table entities, or different components and/or the borders of a single table. In order to achieve the latter, we redesign our experiment protocol to have participants staring at each corner of the table on the given page for two seconds. The clustering technique is applied both directly on ordinary experiments and the experiments with the special protocol.

We have tried various popular clustering approaches including DBSCAN, KMeans, and Affinity Propagation. With parameter tunings, the result on our sample data shows that DBSCAN clustering (Figure 11 right) slightly outperforms the others, probably due to the nature of density-based spatial clustering.

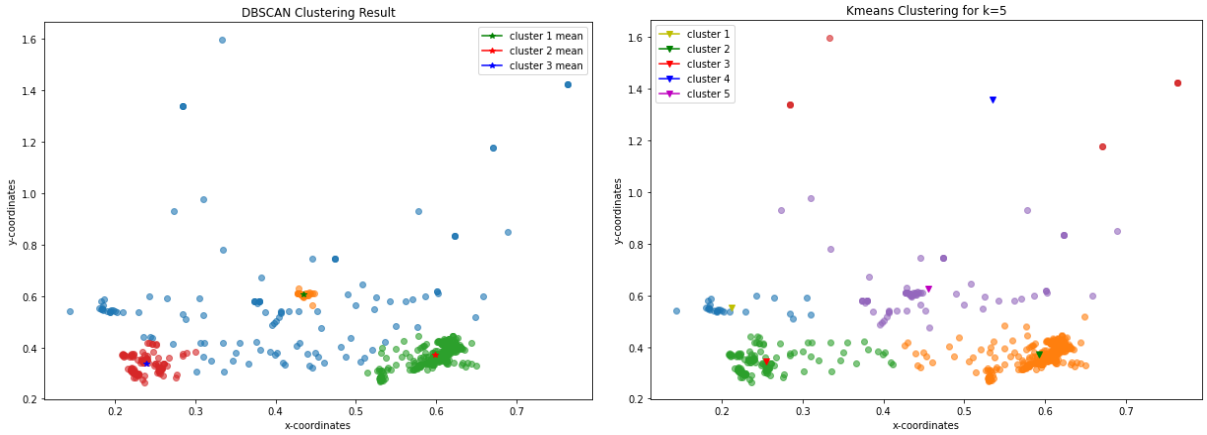


Figure 11: Clustering result for DBSCAN(left) and K-Means(right)

4 Modeling

4.1 Baseline Method: Eyegaze + DBSCAN

Experiments

To collect eye-movement data with Gazepoint's eye tracker equipment, in the first phase of the project, we designed an experiment with the Python package PyGaze [14], which provides an API that establishes data connection to the tracker, calibrates the camera, displays several image-based documents in sequence, and then records eye gaze data and logs events in local files. In each experiment, it performs n trials corresponding to n test images, and in each trial it asks participants to fix their eyes on a table inside the image for 10 seconds. We adopted two approaches and developed several versions of the experiment that differentiate on the instructions, preprocessing process, and prediction models.

DBSCAN Clustering

The baseline method of table extraction was to predict a bounding box for the table of interest solely based on the density distribution of eye gaze data. The rationale behind this approach was that if we capture the eye fixations densely falling around corners of a table, then we can estimate the AOI as a prediction of the table coordinates.

The experiment initially asked participants to stare at each corner of the table for 2 seconds, however, after processing and visualizing the tracking data, we realized that focusing on four corners has problems. First, the dataset contains primarily borderless tables, which, compared to normal tables with clear corner points, made it more difficult for participants to stare at a blank. Also, too much eye movement between the corners led to more noisy and inaccurate data points. Besides, spending approximately equal time on each corner was hard to achieve, which biased the density distribution.

After discussion, we came up with a more efficient way of stare similar to cropping a picture, which was to ask participants to only stare at the upper left corner and the lower right corner, each for 5 seconds. By calculation this method also gives the table coordinates but reduces noise.

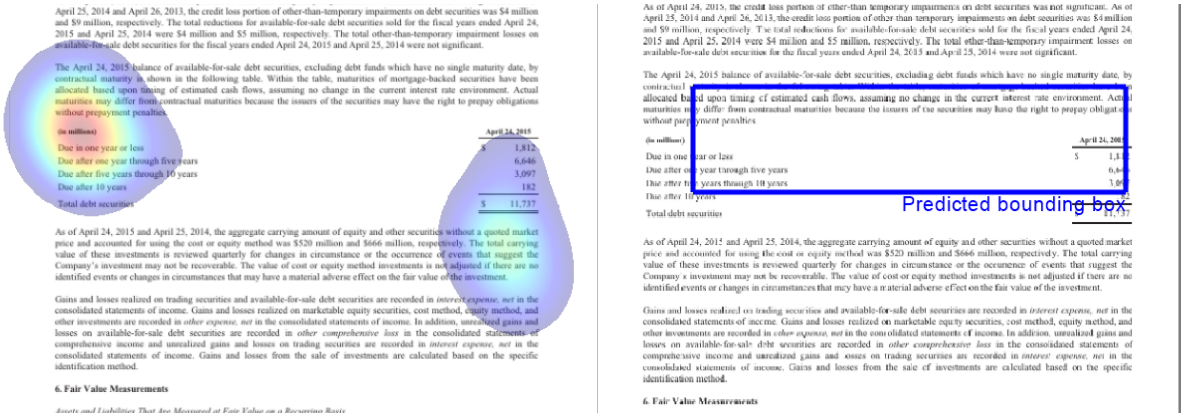


Figure 12: Heatmap of fixation points around two corners

Within a 10-second display time, the eye gaze data was recorded. Like shown in the heatmap above, after denoising the data points would cluster around two centroids, which are approximately the two corners of the table. Therefore we applied DBSCAN clustering to estimate the coordinates of two centroids and draw a bounding box based on their values. The preprocessing pipeline cut points falling outside the image edges and removed the unfixed eye movements, and the DBSCAN model with maximum intra-cluster distance of 50 pixels and 7 minimum samples per cluster could provide the centroids of two desired clusters. The coordinates of the centroids were used to form an AOI bounding box for the table.

4.2 Advanced Method: Eyegaze + CV + OCR Model

Since tables in the documents were borderless, generating accurate bounding boxes solely relying on AOIs was not easy to achieve. Following our initial idea of using the intersection of the AOI bounding box and the box extracted by CV modeling, we implemented a more advanced method combining eye gaze with more precise CV and OCR models, so if there are multiple tables in a document and the user only wants a specific one, AOI would be able to tell which table to extract.

The advanced method combines the CV object detection model and the eye gaze experiment to achieve a one-step solution that performs more precise table detection and extraction. The rationale behind this approach was: all documents have one or more table(s), and regardless of the table of interest, we could utilize CV to first extract all tables from a document and then perform eye gaze trials to determine the AOI and output the information about the desired table. The CV model was embedded in the experiment to predict and display results within the experiment, but they can also work independently with or without the eye tracker equipment, which makes this approach more flexible. In the future, if the equipment was not available or if we would rather recognize all tables within the image, the CV and OCR models could be adapted to different circumstances.

Compared to the baseline version, this method again changes the way of stare. Since the CV model provides more precise coordinates of the tables, the eye tracker only needs to collect information about the approximate location of AOI, so that participants were asked to fix their eyes on the center of the table during the display.

In the end-to-end eye gaze experiment, we ran the CV model first to obtain coordinates of the predicted bounding boxes for all tables, started the experiment environment to collect eye movement data, and calculated the fixation density in each bounding box and crop the table of interest.

4.2.1 YoloV5-based Table Extraction CV Model

It is observed that the document pictures in the dataset are mainly black-and-white or single color, and many of them have similar table contents, sizes, and aspect ratios. Therefore, in order to improve the efficiency of training and alleviate over-fitting, we started from annotating the image dataset based on their different sizes, colors, positions and text densities, among which 83 pictures were selected as the final dataset by their diverse forms. Then, we processed the dataset into a VOC format, and split them into a train set, validation set and test set by the ratio of 0.6 : 0.2 : 0.2.

We choose the YOLO-v5 network to solve the problem. Based on yolov5x pre-trained model, we finetune the model for 300 epochs on 80 annotated images and tested on 20 images.

After running predictions with the images, it is observed that the model can bound the table with a high confidence even if the table is very small, thin, or fat. Figure 13 and Figure 14 shows the bounded table and cropped table content.

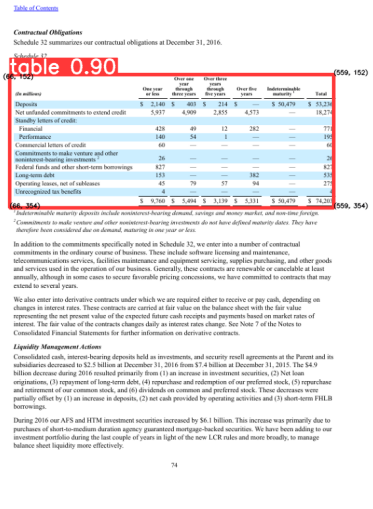
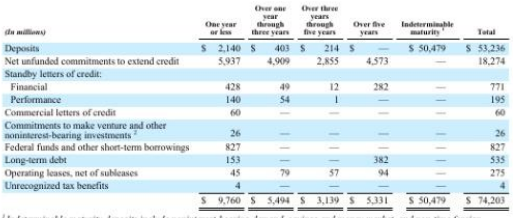


Figure 13: The bounded table



(In millions)	One year or less	Over one year through three years	Over three years through five years	Over five years	Indeterminable maturity	Total
Deposits	$ 2,140	$ 403	$ 214	$ —	$ 50,479	$ 53,236
Net unfunded commitments to extend credit	5,937	4,909	2,855	4,573	—	18,274
Standby letters of credit:						
Financial	428	49	12	282	—	771
Performance	140	54	1	—	—	195
Commercial letters of credit	60	—	—	—	—	60
Commitments to make venture and other noninterest-bearing investments [2]	26	—	—	—	—	26
Federal funds and other short-term borrowings	827	—	—	—	—	827
Long-term debt	153	—	—	382	—	535
Operating leases, net of subleases	45	79	57	94	—	275
Unrecognized tax benefits	4	—	—	—	—	4
	$ 9,760	$ 5,494	$ 3,139	$ 5,331	$ 50,479	$ 74,203

Figure 14: The cropped table

The YOLO [15] network is mainly composed of three main components:

- ***Backbone***: The convolution neural network of image features is formed by aggregating and merging on different fine-grained images.
- ***Bottleneck***: A series of network layers that mix and combine image features, and transfer image features to the prediction layer [16]
- ***Head***: Predict the image features, generate boundary boxes and predict categories.

Figure 15 below shows the general architecture of the object detection network.

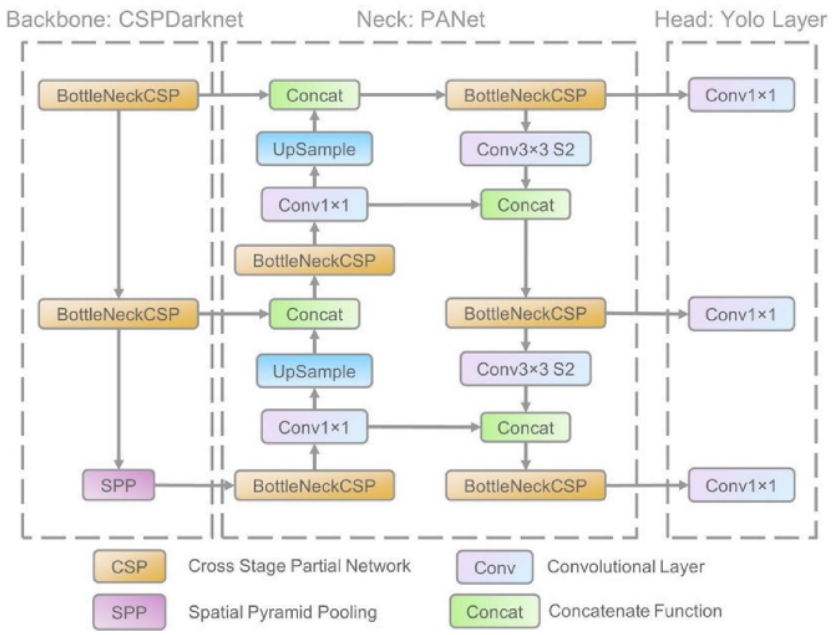


Figure 15: Architecture of YOLO

Image source: https://www.researchgate.net/figure/The-network-architecture-of-Yolov5-It-consists-of-three-parts-1-Backbone-CSPDarknet_fig1_349299852

Training

The training was conducted using Pytorch framework and it is the second development of the source code ultralytics/yolov5: YOLOv5 🚀 in PyTorch > ONNX > CoreML > TFLite (github.com). The whole process of the training are as following:

Annotate the dataset: the original dataset is unannotated, so we use labelImg tool to bound the table and generate an .xml format file which contains the class label and the four coordinates of the bounding boxes.

Process the dataset: After an subset (about 100 images) of the full dataset set is annotated as the dataset for the later experiments, the dataset is transformed into a VOC dataset, which is one of the mostly used dataset format for detection tasks, and is split into train set and validation set, the ratio is 8:2

Train the network: The network was trained on 300 epochs, costing about 1001.62s on an RTX 2070

Finally, the mAP on the test set is 0.824. In Figure 16, training loss and validation loss curve have a steady descending trend, and the ascending curve of mAP, precision and recall shows the network has fit the dataset and extracted the features of the tables.

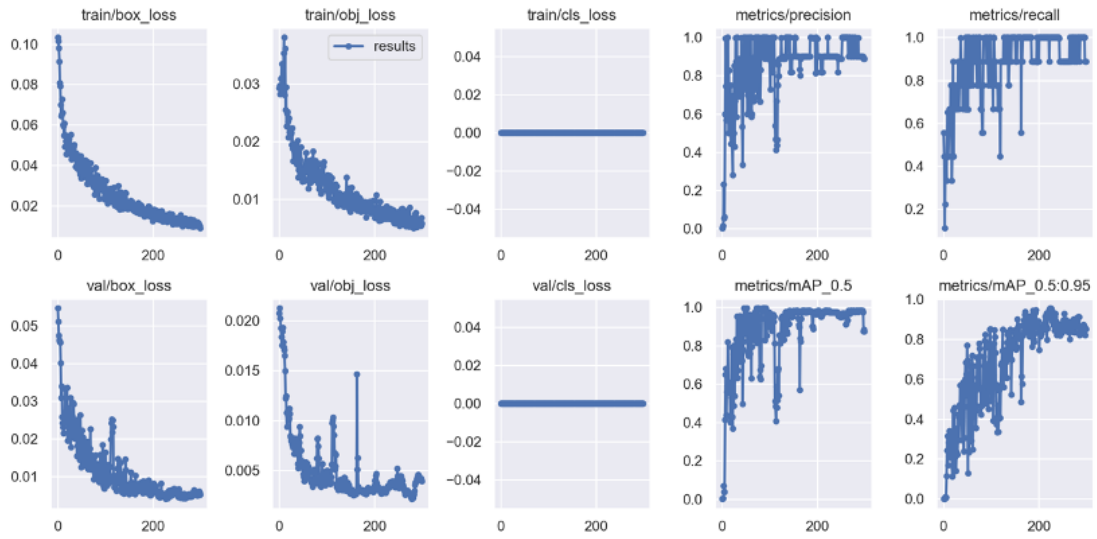


Figure 16: training and validation loss

The AUC value is 0.74, and the P-curve in Figure 17 shows the model has a high confidence on recognizing a table.

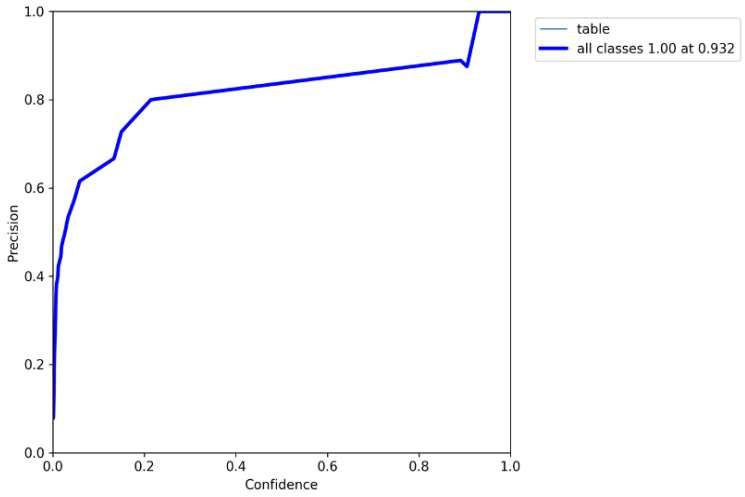


Figure 17: Precision curve of the network

Results

In Figure 18, the prediction result on the test set shows the network has a good ability on recognizing borderless tables with different size, shape and format in average confidence of 0.9. Almost all types of borderless tables could be detected precisely by the model as long as we train enough samples.

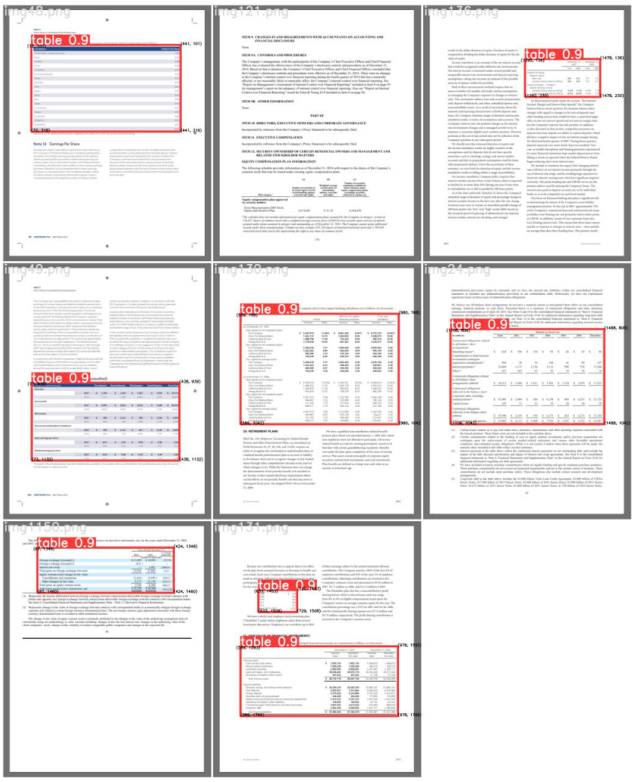


Figure 18: the prediction results on a batch of test images

4.2.2 Textual Information Retrieval (OCR)

Task Objectives

So far, using either Eyegaze + CV advanced method or the Eyegaze end-to-end method, we are able to obtain a cropped image of the table of interest. The current output for our table recognition model is a bounding box for each single table, stored in json format per input image. For the purpose of use in the industry, for example, to improve efficiency of reading financial statements and annual reports for analysts, it would be better if we could deliver a structured table in textual forms. In order to deliver such information, our solution is to develop an OCR model, which transforms a bounded image of a table, with or without structure, to text-based table structure. Upon finishing, we expect to deliver an integrated model with table detection, table recognition and information retrieval.

Common choices for OCR engines include EasyOCR(lightweight but mostly for pdf conversion), Tesseract(for image conversion) and ocropus(based on LSTM, less used nowadays), etc. With model selection, we've chosen Google's open source tesseract OCR engine [17] due to its good performance in high-resolution images.

Training

The Tesseract package offer a pretrained model based on English-language corpus. As a result, our OCR model is trained based on the built-in Tesseract English-language model, with our dataset, which contains annotated images of 2000 tables sampled from FinTabNet dataset. Because the training module of Tesseract, "tesstrain", supports single-line page segmentation(psm=7) only, we have cropped images of each single cell from the entire tables so that the format of our training data aligns with the package requirements. Our final train data contains 12.8k image of single-cell images. With 5000 iterations of training, we obtain a retrained language model "fintabnet_full.traineddata" specialized for the fonts and formats of our table images.

Preprocessing

As we conduct OCR predictions **u**sing the retrained model, we have encountered a few problems: firstly, the header of a table are usually in a darker background color, which makes the model harder to recognize and make precise predictions. Secondly, we found the model performance much lower when recognizing the whole table than recognizing each cell, due to the single-cell segmentation mode during the training stage.

In order to solve the above mentioned problems, we apply image preprocessing techniques on the input images before prediction. Effective preprocessing steps include removal of cell backgrounds and reconstruction of table structure, so that we could separate each cell from the entire table. Thus, we are able to obtain a cropped augmented image with white background for each single cell in every table. The background removal and structural recognition methods are all implemented using OpenCV module.

Prediction

Our OCR predictor offers two modes for table recognition, the *table* mode and the *cell* mode. The table mode takes as input an image of the entire table, and outputs all text with built-in table structure. This mode is relatively faster and is able to preserve the original row-column table structure. The cell mode takes as input an image of a single-line image, namely of a single cell or row of the table, recognizes text per cell, then we reconstruct the original table structure with our proposed preprocessing techniques. The cell mode is more accurate while it preserves row-wise table structure only.

Evaluation Result

The model is evaluated using 10% of the entire dataset, which contains 1.2k single-line images of table cells. Metrics used for evaluation are Character Error Rate (CER), Word Error Rate (WER) and Accuracy. As **Table 3** indicates, we obtain the highest accuracy of 98.9% over the test dataset with single-line page segmentation, namely our *cell mode*.

$$CER = (S_c + D_c + I_c) / N_c$$
$$WER = (S_w + D_w + I_w) / N_w$$
$$Accuracy = (1 - WER) * 100\%, where:$$
$$S = Number\ of\ Substitutions,\ per\ character(word)$$
$$D = Number\ of\ Deletions,\ per\ character(word)$$
$$I = Number\ of\ Insertions,\ per\ character(word)$$
$$N = Number\ of\ characters(words)\ in\ reference\ text\ (aka\ ground\ truth)$$

Recognition Mode	Structured Text mode (psm 6)	Unstructured Text mode (psm 12)	Single-line (psm 7)
Train Acc.	-	-	98.4%
Test Acc.	97.0%	97.7%	98.9%

Table 3: evaluation result for each OCR recognition mode

4.3 Comparison

The evaluation of two methods focused on three aspects: time consumption, accuracy, and consistency of results.

4.3.1 Time

For time consumption, a single experiment using either method took approximately the same amount of time: 4 test images took around 3 minutes 40 seconds in minimum, and 4 minutes on average. Adding an additional image would increase the experiment time by 15~20 seconds. The calibration step accounted for a large proportion of the time fluctuations since it might need several calibrations until getting accurate results, and the baseline method requires higher calibration accuracy than the advanced method. Repeating the calibration for another time would increase the experiment time by 30 seconds approximately.

The following two pie charts represent the time decomposition for a sample baseline experiment and a sample advanced experiment using 4 identical test images. In the sample experiments, the total time consumption for the baseline approach was 3 minutes 40 seconds and for the advanced approach was 3 minutes 36 seconds.

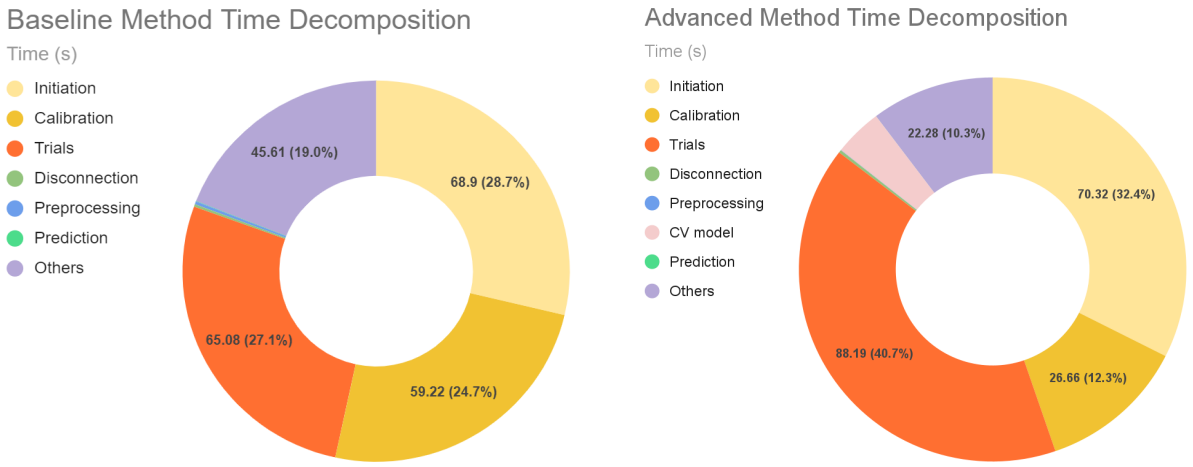


Figure 19: Time decomposition of a sample advanced experiment with 4 test images

As shown above, the initiation (connection to device), calibration, and the experiment trials (display of images) took more than 80% of the time, whereas the preprocessing and prediction cost trivial time. The difference in time spent on initiation and trials between two approaches was caused by fluctuations related to requests, buffering, and logging, but the difference in time spent on calibration was based on the times of calibration attempts. In the samples, the baseline experiment carried out two calibrations while the advanced experiment only performed one. This simulated the real scenarios because the baseline prediction requires higher calibration accuracy.

In the advanced experiment, the CV model running time was only 8.73 seconds, taking 4% of the total. The 'Others' part in two experiments was primarily related to the user, and it included the time waiting for keyboard press or mouse clicks, as well as the time for manually displaying outputs after predictions.

4.3.2 Accuracy

The accuracy of the baseline method fluctuates violently and greatly depends on the accuracy of calibration and recording because it counts on precise eye gaze data points to estimate the table coordinates. When one or both eyes deviate from the calibrated position, the clustering model would not be able to clearly differentiate clusters, thus the predicted table using centroid coordinates could be badly inaccurate.

However, the advanced approach only uses fixation data to calculate how much of the attention falls into each bounding box and determines the table of interest. The advanced method was more robust against inaccurate calibrations and also provided much more intuitive and accurate results that completed our major tasks.

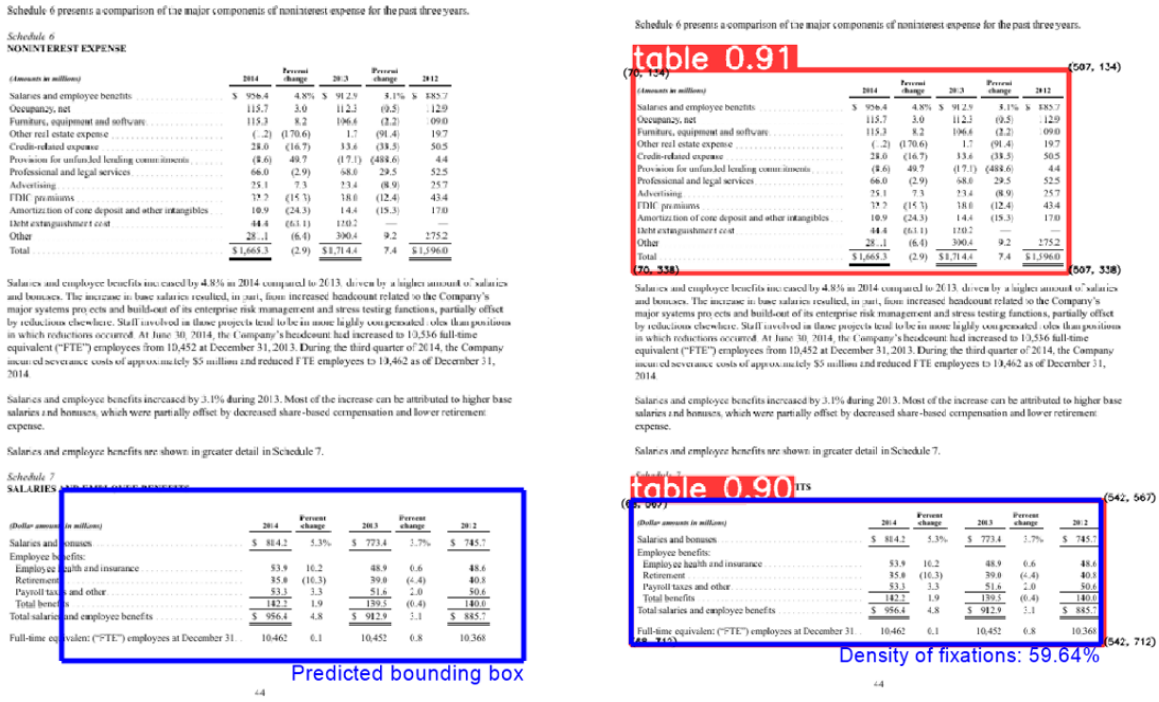


Figure 20: Table in the same image predicted by the baseline vs. advanced method

As shown above, the bounding box predicted by the advanced approach is much more accurate than the baseline approach, which did not crop the whole table.

4.3.3 Consistency

The consistency of model predictions is significant. The quality of the eye gaze data was impacted by various factors, like device positioning, lighting, mobility of participants, calibration accuracy, quick blinks, fixation duration threshold, etc., and most noisy factors could not be eliminated by simple variable control or training. Comparing the two methods, the advanced method showed much higher consistency contributed by the well-trained CV results. Integrating deep learning models into the experiment largely reduced the information dependence on the eye tracker and thus led to much more stable performance and predictions.

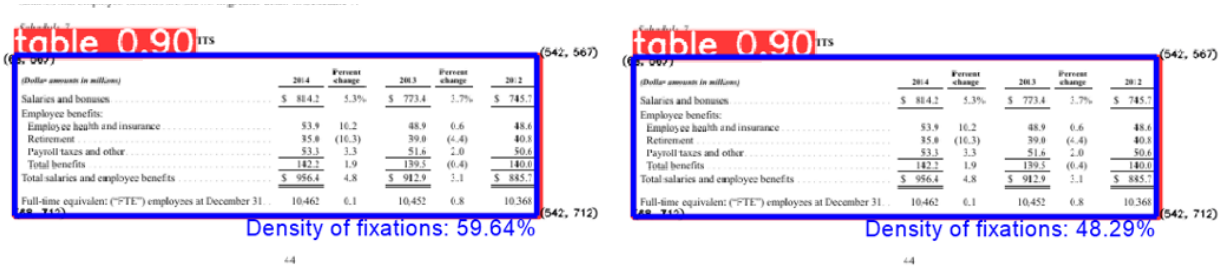


Figure 21: Prediction of the same image in two independent experiments by the advanced method

Figure 21 shows the reliability of the advanced approach as it generates consistent prediction results when the same test image is displayed in multiple independent experiments.

5 Production

The final task of the project is to wrap up the source code for all processes mentioned above and write production-level code. We built a Python package `pygazeTE` adapted from an open-source package named pygazeanalyser [14], which is a visualization library affiliated with PyGaze. We developed a preprocessing module that includes functions for data preprocessing, and we adjusted the original plotting functions to draw heatmaps, scanpaths, etc. of the eye gaze data. We automated all processes in the experiments and packaged all source code in a repository. It contains the environments, datasets, models, experiments in Jupyter Notebooks, and reports of the project so that the users can download the depository and run an instance on their local machine.

6 Future Work

- **Eye gaze data quality improvements**: The accuracy of eye gaze data was unstable caused by calibration inaccuracy and trivial posture changes. Improving eye gaze validation would lead to more consistent results.
- **Generalization to more diverse tables**: Our current solution was not tested on unstructured tables or tables with multi-level headers.
- **Deep learning methods for table structure**:We used OpenCV's image processing techniques to recognize table structures, but DL solutions would allow training and tuning to provide more flexible and accurate results.

7 Contribution

Shihang Wang: Main contributor on public data collection and labeling, experiment design and data collection and times series prediction; dataset treatment and annotation acquisition.
Yeqi Zhang: Main contributor on exploratory data analysis, dimensionality reduction, times series prediction; YoloV5-based Table Extraction CV model development and training.
Yibai Liu: Team Captain: set up timelines, manage progress and coordinate work; main contributor on experiment code interface design, experimental data collection and visualization. Eye Gaze experiments design and model integration, PyGazeTE package build-up.
Yijia Jin: Main contributor on exploratory data analysis, clustering analysis; Other table detection CV algorithms: TableNet-based model, OCR model development.
Yinqiu Feng: Main contributor on data preprocessing, feature extraction and dimensionality reduction.

REFERENCES

[1] Ren, S., He, K., Girshick, R., & Sun, J. (2015). Faster r-cnn: Towards real-time object detection with region proposal networks. Advances in neural information processing systems, 28.

[2] Tensmeyer, C., Morariu, V. I., Price, B., Cohen, S., & Martinez, T. (2019). Deep splitting and merging for table structure decomposition. In 2019 International Conference on Document Analysis and Recognition (ICDAR) (pp. 114-121). IEEE.

[3] Wang, N. X. R., Burdick, D., & Li, Y. (2021). TableLab: An Interactive Table Extraction System with Adaptive Deep Learning. In 26th International Conference on Intelligent User Interfaces-Companion (pp. 87-89).

[4] Qasim, S. R., Mahmood, H., & Shafait, F. (2019, September). Rethinking table recognition using graph neural networks. In *2019 International Conference on Document Analysis and Recognition (ICDAR)* (pp. 142-147). IEEE.

[5] Mori, S., Nishida, H., & Yamada, H. (1999). Optical character recognition. John Wiley & Sons, Inc..

[6] Raja, S., Mondal, A., & Jawahar, C. V. (2020). Table structure recognition using top-down and bottom-up cues. In European Conference on Computer Vision (pp. 70-86).

[7] Makihara, Y., Takizawa, M., Shirai, Y., Miura, J., & Shimada, N. (2002, August). Object recognition supported by user interaction for service robots. In Object recognition supported by user interaction for service robots (Vol. 3, pp. 561-564).

[8] Ehinger, K. A., Hidalgo-Sotelo, B., Torralba, A., & Oliva, A. (2009). Modelling search for people in 900 scenes: A combined source model of eye guidance. Visual cognition, 17(6-7), 945-978.

[9] H. Muñoz, F. Vilariño and D. Karatzas, "Eye-Movements During Information Extraction from Administrative Documents," 2019 International Conference on Document Analysis and Recognition Workshops (ICDARW), 2019, pp. 6-9, doi: 10.1109/ICDARW.2019.20045.

[10] Beymer, D., & Russell, D. M. (2005, April). WebGazeAnalyzer: a system for capturing and analyzing web reading behavior using eye gaze. In CHI'05 extended abstracts on Human factors in computing systems (pp. 1913-1916).

[11] Shanmuga Vadivel, K., Ngo, T., Eckstein, M., & Manjunath, B. S. (2015). Eye tracking assisted extraction of attentionally important objects from videos. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 3241-3250).

[12] Christophe Rigaud, Thanh-Nam Le, J.-C Burie, J.-M Ogier, Shoya Ishimaru, et al.. Semi-automatic Text and Graphics Extraction of Manga Using Eye Tracking Information. 12th IAPR Workshop on Document Analysis Systems (DAS), Apr 2016, Santorini, Greece. pp.120 - 125, ff10.1109/DAS.2016.72ff.ffhal-01336346ff

[13] Zheng, X., Burdick, D., Popa, L., Zhong, X., & Wang, N. X. R. (2021). Global table extractor (gte): A framework for joint table identification and cell structure recognition using visual context. In Proceedings of the IEEE/CVF winter conference on applications of computer vision (pp. 697-706).

[14] Dalmaijer E. and Mathot, S. (2014). PyGaze, GitHub. Retrieved on Apr. 16, 2022 from https://github.com/esdalmaijer/PyGazeAnalyser

[15] Joseph Redmon, Santosh Divvala, Ross Girshick, Ali Farhadi, "You Only Look Once: Unified, Real-Time Object Detection"

[16] Kaiming He, Georgia Gkioxari, Piotr Dollár, Ross Girshick, "*Mask R-CNN for Object Detection and Segmentation*" (Smith, 2007, #)

[17] Smith, R. (2007). An overview of the tesseract OCR engine. Ninth International Conference on Document Analysis and Recognition (ICDAR 2007) Vol 2 (pp. 629–633). Presented at the Ninth International Conference on Document Analysis and Recognition (ICDAR 2007) Vol 2, IEEE.

[18] Farnsworth, B. (2020). 10 Most Used Eye Tracking Metrics and Terms. Imotions. Retrieved on Feb 4, 2022 from https://imotions.com/blog/10-terms-metrics-eye-tracking/#aoi

[19] Holomb, V. (2021). "Borderless Tables Detection with Deep Learning and Opencv." Medium, Towards Data Science, Retrieved on Feb. 15, 2022 from https://towardsdatascience.com/borderless-tables-detection-with-deep-learning-and-opencv-ebf568580fe2.

[20] Khan, S. A., Khalid, S. M. D., Shahzad, M. A., & Shafait, F. (2019, September). Table structure extraction with bi-directional gated recurrent unit networks. In 2019 International Conference on Document Analysis and Recognition (ICDAR) (pp. 1366-1371).

[21] Koehler, K., Guo, F., Zhang, S., & Eckstein, M. P. (2014). What do saliency models predict?. Journal of vision, 14(3), 14-14.

[22] Laique, S. N., Hayat, U., Sarvepalli, S., Vaughn, B., Ibrahim, M., McMichael, J., ... & Rizk, M. K. (2021). Application of optical character recognition with natural language processing for large-scale quality metric data extraction in colonoscopy reports. Gastrointestinal endoscopy, 93(3), 750-757.

[23] Li, Y., Huang, Z., Yan, J., Zhou, Y., Ye, F., & Liu, X. (2021). GFTE: graph-based financial table extraction. In International Conference on Pattern Recognition (pp. 644-658).

[24] Marius, H. (2021, December 10). A table detection, cell recognition and text extraction algorithm to convert tables to excel-files. Medium. Retrieved May 12, 2022, from https://towardsdatascience.com/a-table-detection-cell-recognition-and-text-extraction-algorithm-to-convert-tables-to-excel-files-902edcf289ec

[25] Muñoz, H., Vilariño, F., & Karatzas, D. (2019, September). Eye-Movements during information extraction from administrative documents. In 2019 International Conference on Document Analysis and Recognition Workshops (ICDARW) (Vol. 3, pp. 6-9).

[26] Paliwal, S. S., D, V., Rahul, R., Sharma, M., & Vig, L. (2019). TableNet: Deep Learning Model for End-to-end Table Detection and Tabular Data Extraction from Scanned Document Images. 2019 International Conference on Document Analysis and Recognition (ICDAR) (pp. 128–133). Presented at the 2019 International Conference on Document Analysis and Recognition (ICDAR), IEEE.

[27] Patel, C., Patel, A., & Patel, D. (2012). Optical character recognition by open source OCR tool tesseract: A case study. International Journal of Computer Applications, 55(10), 50-56.

[28] Redmon, J., & Farhadi, A. (2018). Yolov3: An incremental improvement. arXiv preprint arXiv:1804.02767.

[29] S. Karthikeyan, Thuyen Ngo, M. Eckstein and B. S. Manjunath, "Eye tracking assisted extraction of attentionally important objects from videos," 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015, pp. 3241-3250, doi: 10.1109/CVPR.2015.7298944.

[30] S. Paliwal, V. D, R. Rahul, M. Sharma and L. Vig, "TableNet: Deep Learning Model for End-to-end Table Detection and Tabular Data Extraction from Scanned Document Images," in 2019 International Conference on Document Analysis and Recognition (ICDAR), Sydney, Australia, 2019 pp. 128-133.

[31] Ultranalytics (2021). YoloV5 in PyTorch, GitHub. Retrieved on Apr. 16, 2022 from https://github.com/ultralytics/yolov5

[32] Zheng, X., Burdick, D., Popa, L., Zhong, X., & Wang, N. X. R. (2021). Global table extractor (gte): A framework for joint table identification and cell structure recognition using visual context. In Proceedings of the IEEE/CVF winter conference on applications of computer vision (pp. 697-706).