Table Extraction via Eye Gaze Tracking

Columbia Team:

Yibai Liu

Yijia Jin

Yeqi Zhang

Shihang Wang

Yinqiu Feng

Mentors from JP Morgan & Chase:

Nonie Thomas, Saheed Obitayo, Daniel Borrajo, Naftali Cohen

Motivation and Objective

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 in making processes more efficient for numerous teams throughout the business, and it is a task which other machine learning methods, such as NLP, struggle to perform. A use case can be to automatically extract all tables from an earnings report or other comprehensive document by gazing at the tables.

Motivated by real-world scenarios, this project focuses on:

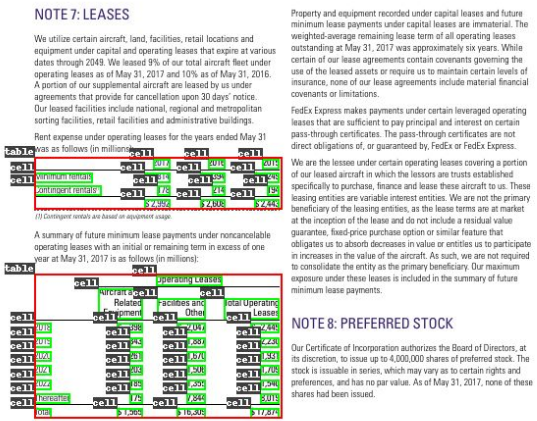
- Leveraging eye gaze technology and **Computer Vision (CV)** to design a system that automatically detects tables from an image-form document and distinguishes the table of interest by eye fixations.
- Extracting texts from table with the **Optical Character Recognition (OCR)** tool Tesseract.

Dataset Description

This project utilized IBM's open source dataset **FinTabNet** for eye gaze tracking and table extraction tasks. The dataset contains documents from the annual reports of S&P 500 companies with diverse borderless tables, and we used a subset of **2,000 PDF** files, each PDF has detailed annotations in JSON format for table structure and content.

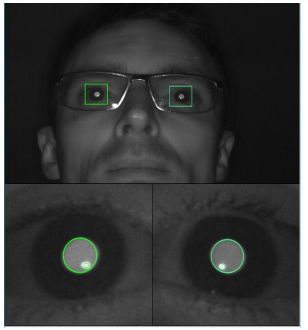
Compared to tables in scientific and government documents, financial tables have more complex styles and color variations, making the tasks more challenging but also more flexible and practical in applications.

To match the scope of this project, we transformed PDF files into images using the python package `pdf2img`, and used the **2,000 image-form documents** as the training/validation/testing data of our models.



Experimental Data Collection

- We designed a **webcam-based eye tracking experiment** in Python using the open-source toolbox PyGaze, a Python wrapper for Gazepoint's OpenGaze API, to establish connection with the **Gazepoint equipment GP3**.
- In an experiment trial, there are instructions that lead the participant to calibrate the device, then a number of image-format documents are displayed on the screen. The participant is asked to look for table(s) in the image and gaze at **one table of interest** for 10 seconds per image. The Gazepoint eye tracker records and writes in a .tsv file the fixation duration and coordinates, the gaze direction, the 3D data of pupils, timestamps, etc.
- The experiment data were used to identify eye movements and areas of interest (AOIs).



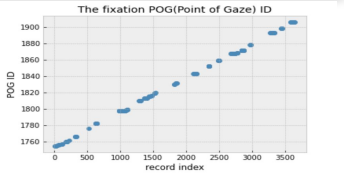
Data Exploration and Insights

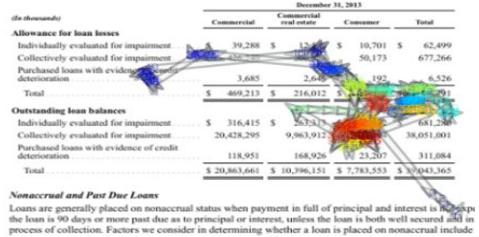
Consistency of missing value: Some data were missed during regular blinking, and each blink took about 0.3 seconds. The frequency of blinking = total time / the count of blinking. We could use a participant's fixation data to calculate blinking frequency, which was ~2.5 seconds/blink.

Difference of reading patterns: In initial eye gaze explorations, each of the team members tried reading tables by our own means. The scanpaths showed clear difference between reading habits, e.g. from left to right, from top to bottom, from result to details, from number to texts, …

Time Series analysis: After performing time series dependency tests, we can potentially predict the eye movements under the experiment settings via ARIMA(0,1,1) model.

Calculation of fixations: In the initial eye gaze trials, we found FPOGX (x-coordinate of fixation point-of-gaze) sometimes had a slightly higher correlation with one eye than the other, indicating the calculation of POGs weighs two eyes differently. This can relate to the trivial difference between the 3D data of two eyes, or it might relate to the eye dominance (some people's focus rely more on one eye than the other).



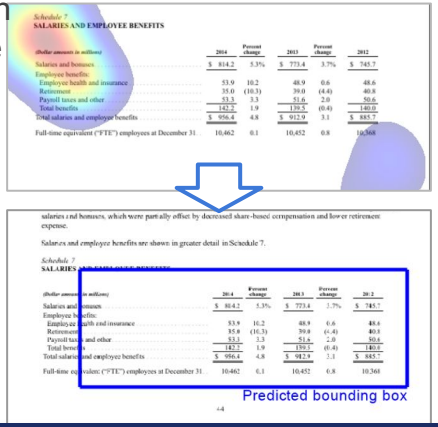


Baseline Approach - Eye Gaze only

The baseline approach only utilizes eye fixations to determine a bounding box for the table of interest.

- The participant is asked to only stare at the **upper left corner and the bottom right corner** of the table, similar to cropping an image.
- By **DBSCAN clustering**, we can determine centroids of two clusters and draw a box based on coordinates of the centroids.

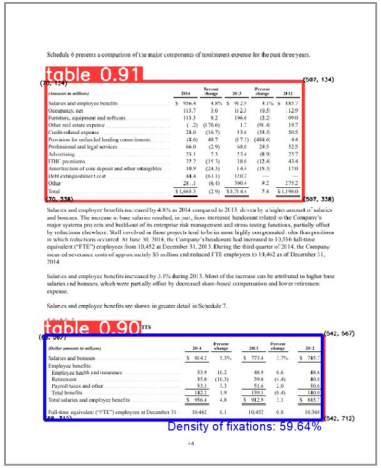
However, this approach largely relies on accurate calibration results and precise fixation points. It needs multiple calibrations until getting high accuracy, and any trivial posture change of the participant can affect the results.

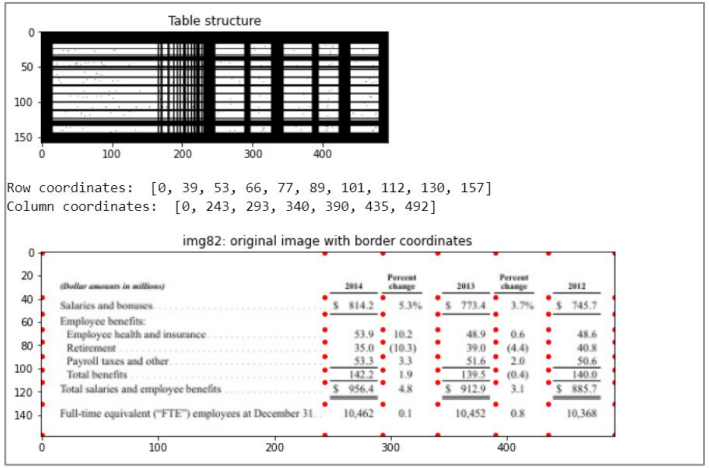


Modeling Approach - CV + OCR

To reduce dependency of the fixation data accuracy, the modeling approach utilizes a CV model and an OCR model to predict table boundaries and recognize texts inside the tables, and then combines eye gaze technology to determine the table of interest.

- We implemented ***YOLO-v5,*** an object detection CV architecture that divides images into a grid system, and fine-tuned it with our document data to enable accurate table predictions with confidence scores.
- ***Tesseract*** is an OCR tool that recognizes texts within tables. We used the package `pytesseract` and retrained model weights to adapt to our dataset.
- Based on model outputs, the eye gaze experiment captures the participant's focus and calculates fixation densities to identify the table with higher density.

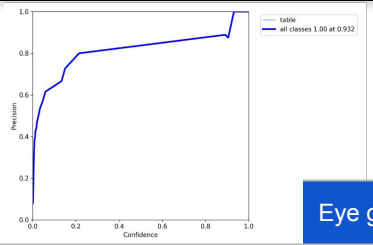




Key Results

CV model

- Trained using yolov5x.pt pretrained weights for 300 epochs on 80 annotated images, tested on 20 images; the mAP was 0.824.
- The Classification Loss, Localization Loss and Confidence Loss were 0.0, 0.005, 0.004.
  - Total Loss = Classification Loss + Localization Loss + Confidence Loss
  - For only one category, the 'tables', the classification loss is 0.
- The P-Curve shows precision > 0.8 for predictions with confidence score > 0.2 in the test set.



OCR model

- Trained using Tesseract OCR with English-language pretrained model for 5000 iterations on 2000 images, on single-line segmentation mode (psm-7).

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

Acc. = 1-CER

		Eye gaze	Eye gaze +CV+OCR
Mean time (4 images per trial)		4.5 min	3.5 min
Time decomposition (includes response wait time)	Model prediction	/	13 sec
	Initiation	70 sec	70 sec
	Calibration	70 sec	30 sec
	Experiment	130 sec	90 sec
	Preprocessing	0.67 sec	0.06 sec
	AOI prediction	0.06 sec	0.01 sec

Conclusion and Next Steps

- Implemented transfer learning and developed a YOLOv5 object detection model and a Tesseract OCR model to identify tables in a document and recognize texts within the tables;
- Designed a baseline eye gaze tracking experiment and an advanced experiment combining CV models to collect eye fixation data and predict bounding boxes for tables in AOIs.

Next Steps

- **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 applied to FinTabNet's borderless tables and semi-structured tables with borders, but 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.

Thanks for listening!