EE447 Mobile Internet Lab 4

Zhou Litao 518030910407 F1803016

June 11, 2021, Spring Semester

1 Introduction

PDF documents have become the mainstream document format because of its unique cross-platform convenience advantage. PDF documents contain a large amount of valuable data information, and the table is one of the important carriers of these data. However, the structure of PDF documents is complex, and it is difficult for us to obtain accurate table information directly from the document format. Therefore, for PDF tables, we need to reconstruct the structure of the table, so as to achieve the extraction of the table.

2 Purpose

This Lab focuses on the table line reconstruction for the tables without frame lines. In this Lab, we use python to complete table line drawing of a specific table without frame lines.

3 Line Construction

The implementation of line construction is summarized as follows.

1. Input an image of table and convert it to greyscale.
2. Use `cv2.Canny` and `cv2.HoughLinesP` to extract edge and lines from the original image.
3. Among the extracted lines, find the contour of the whole table by counting the leftmost/rightmost/top/bottom location of all the lines detected above.
4. Remove the lines in the orginal image.
5. Within the discovered region, loop through every row/column of pixels. Mark consecutive regions of all blank pixels as a "tab" separator. Draw a line in the middle of the region.
6. Output the image with the lines discovered above.

A running example can be found in Figure 1

Remark The implementation has been provided by the teaching assistant in `table_reconstruction.py`. However, this implementation has a few limits, remarked as follows

- In step 2, the contour of the table has to be determined with the help of existing frames (e.g. the top frame and the bottom frame). In other words, this method will not work on a table without any frames. But this can happen since tables in real pdf files can have various formats.
- In step 3, the contour of the table is determined by the outmost point of line segments. Therefore, this method requires that the input should be an independent image instead of a whole page.
- In step 5, the lines are drawn by scanning the whole row. However, some table may have joint headers, disabling the detection of some lines between columns, shown in Figure 2

Algorithms	Measures	Datasets			
		Dolphin	Football	Karate	Strike
CLP-EB	Best	0.8850	0.8100	0.8450	**0.9400**
	Mean	0.7760	0.7610	0.6925	**0.8095**
	SDev	0.0716	0.0402	0.0868	0.1528
CLP-EP	Best	0.8900	0.8300	0.8300	**0.9400**
	Mean	0.8135	0.7785	**0.7540**	0.7525
	SDev	0.0454	0.0506	0.0581	0.1187
CLP-ES	Best	**0.9050**	**0.9000**	**0.8800**	0.9300
	Mean	**0.8140**	**0.7805**	0.7225	0.6975
	SDev	0.0639	0.0569	0.0787	0.1647

Algorithms	Measures	Datasets			
		Dolphin	Football	Karate	Strike
CLP-EB	Best	0.8850	0.8100	0.8450	**0.9400**
	Mean	0.7760	0.7610	0.6925	**0.8095**
	SDev	0.0716	0.0402	0.0868	0.1528
CLP-EP	Best	0.8900	0.8300	0.8300	**0.9400**
	Mean	0.8135	0.7785	**0.7540**	0.7525
	SDev	0.0454	0.0506	0.0581	0.1187
CLP-ES	Best	**0.9050**	**0.9000**	**0.8800**	0.9300
	Mean	**0.8140**	**0.7805**	0.7225	0.6975
	SDev	0.0639	0.0569	0.0787	0.1647

Figure 1: Example of Line Construction

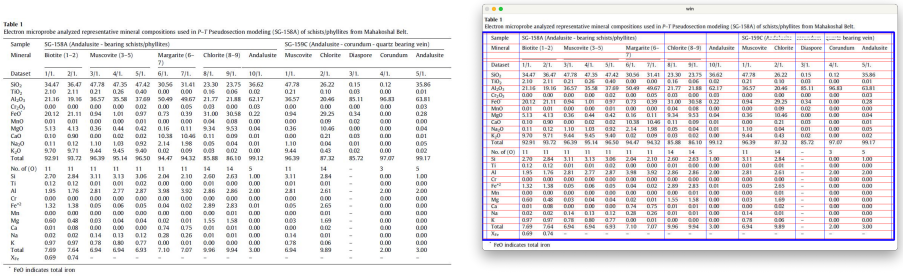

Figure 2: Example of Line Construction

4 Table Detection

The above problem mainly stem from the fact that the methods implemented in `table_reconstruction.py` does not fully consider textual information, but relies on an outmost "frame" to locate the table.

To address this issue, we tried another method[1] in `detect_table.py`, which can not only reconstruct table lines, but also detect the table on a page with texts.

The steps of the methods are listed as follows.

1. Use `cv2.dilate` to convert the text into the solid spots, shown in Figure 3.
2. Apply `cv2.findContours` to find text bounding boxes.
3. Filter the bounding boxes within an appropriate height range (i.e. the height of a normal-sized font).
4. Cluster the text boxes into groups by their coordinates, so that we can find a groups of text areas aligned into rows and columns.
5. Sort them by x and y coordinates, find if the grouped text boxes can form a table.

The reconstructed image is shown in Figure 4
The above method also serves as a solution to Exercise 1.

Exercise 1 How to automatically locate the tables in a PDF?

Solution. The main idea is that unlike texts, all table structures share a clear space separator for us to infer its table lines. To make it easier for us to analyze, we can first dilate the text areas, so that texts will be joined together but not for tables. We can then find contours on the image and group the boxes to detect if they form a whole table. □

[1]Reference: `https://stackoverflow.com/questions/50829874/how-to-find-table-like-structure-in-image`

Figure 3: Dilated table image

Exercise 2 What do you think is the most difficult step to extract the table from the PDF? why?

Solution. In our new implementation, we will have to tune many parameters, such as dialating size, the range of the filtered boxes and the criteria for recognizing a table. In real world, the tables in different papers have different table formats and some tables may have nested structures. These parameters should be tuned carefully so that the method can achieve its generality. Some deep learning methods may help with this issue. □

Table 3 Comparison of CLP-EB, CLP-EP, and CLP-ES in terms of Ranking Score (RS)

Algorithms	Measures	Datasets			
		Dolphin	Football	Karate	Strike
CLP-EB	Best	**0.2914**	0.2751	**0.4608**	**0.7960**
	Mean	**0.2272**	**0.2386**	**0.3176**	0.2225
	SDev	0.0437	0.0157	0.0878	0.2217
CLP-EP	Best	0.2376	0.2428	0.3597	0.5058
	Mean	0.1827	0.2280	0.2523	0.2617
	SDev	0.0401	0.0122	0.0709	0.1506
	Best	0.2824	**0.2814**	0.3628	0.6402
CLP-ES	Mean	0.1919	0.2145	0.2647	**0.3090**
	SDev	0.0566	0.0334	0.0773	0.1864

parameters: δ, ϵ and q. We have used the values suggested in [29]: $\delta = \frac{1}{n}$, $\epsilon = 0.05$ and the value of q is computed with the formula $\left\lceil \frac{(\log((2\times m)/\delta))}{(2\times\epsilon^2)} \right\rceil$, where n is the number of nodes and m is the number links in the network. The algorithm of edge k-path centrality measure [17] also has three parameters: k, ρ and β. We have considered $k = 5$, $\rho = m - 1$ and $\beta = \frac{1}{m}$, where m is the number of links in the network. Edge betweenness centrality algorithm [12] does not have any parameter so CLP-EB is free from parameter setting. All the algorithms are implemented using MATLAB scripting language and executed on MATLAB version R2010a. All the experiments were conducted on a 64-bit Computer having Intel (R) Core (TM) i3-3217U CPU@ 1.80GHz processor and 6GB memory.

Figure 4: Image after table detection