CMPT 310 Group Project MileStone 2

Aug 2, 2025

Group 6, Daniel Shi, Farzan Ustad and Andy Wang

1. **Brief Project Recap**

Our project is working, finding the total from a receipt photo and extracting that into text. This is useful when users may want to keep track of receipt totals but do not want to manually input it every time. The technical aspect of our project is preprocessing our receipt database and then using text recognition on it.

2. **Significant Accomplishments**

Accomplishment 1: Preprocessing pipeline for receipts with non-adjustable parameters

We have built a modular preprocessing pipeline designed to increase accuracy on our receipt image database. Our pipeline currently consists of 3 major components.

1. Grayscale Conversion

This simplifies the input image by compressing color information into a single intensity channel, which helps reduce later computational complexity. The library we used to implement this function is OpenCV.

2. Binarization

This step converts our grayscale image into a strict black or white image to further sharpen text boundaries. The libraries used to implement this are OpenCV and NumPy.

3. Skew Correction

Lastly, Skew correction tilts the receipt by trying to rotate the image in small steps (-5 to +5 degrees). Then it scores each rotation by how well the text lines up horizontally. Using this information it will produce an ideal rotation. For example, image 010.jpg (provided as images with the report) has a deskew angle of -3 degrees. The libraries used to implement this function are OpenCV, NumPy and SciPy.

4. Rescale

This is something that is not currently in our pipeline, however we have implemented and are currently experimenting with it. In our tests, it seems to lower accuracy therefore we have kept it out.

Accomplishment 2: Cross Validation Training + Preprocessing Pipeline with parameters

After implementing the basic pipeline we began to work on a model that we could implement with Cross Validation.

1. Grayscale Conversion

For Grayscale Conversation, we have introduced CLAHE, which helps images that have uneven lighting or faded text. The two parameters we have introduced for this function is clip_limit, which is how much local contrast is boosted and tile_grid_size, which is the size of the grid for applying CLAHE.

2. Biniarization

The updated Biniaization method now accepts a method parameter, between Otsu or Adaptive and tuning parameters for block size and constant subtraction. The tuning parameters only affect the function when the Adaptive method parameter is selected.

3. Skew Correction

The updated Skew correction method now has two parameters, delta for how small the steps are when rotating and limit for how far in either direction to rotate each image. The limit primarily affects the computational speed since as long as the true skew is within the limit range, increasing it will not improve accuracy.

4. Rescale

Although not in our pipeline, we have added three parameters to this function for testing. Font_size_thresh which is used to filter OCR outputs by minimum font size, large_scale which is a large text scaling multiplier and small_scale, a small text scaling multiplier.

5. Cross Validation System

To maximize these parameterized steps, we wrote a separate cross validation script. This script runs the pipeline in order across a variety of preset grid parameters using randomness, then compares the extracted total with the truth value in our database JSON file. Accuracy is logged for each combination. Along with this we created a progress bar with an ETA using a progress bar library. This brought to our attention the long training time our model would take therefore we introduced three parameters to the cross validation function for more control. A max_files which controls how many unique images from the database will be used, n_splits which is the amount of folds in the training model and sample_size which is how many parameter grid combinations we will test. Rescale has yet to be added to this function, due to it lowering the accuracy of our system in all tests.

6. Lessons from this accomplishment

So far although its overall accuracy is much lower than our basic pipeline through this model we have learned that Otsu threshold scores a higher accuracy when compared to Adaptive. Of course, so far the cross validation is not as exhaustive as we would like it but that is our major take aways so far.

Accomplishment 3: Text Extraction and Full Scale Testing

The text extraction is divided into three steps: 1. Correct format; 2. Text recognition, extraction, and sorting; 3. Identify the total amount.

1. Correct format, image pre-processing

2. Text recognition

We use the *pytesseract.image_to_data* function to detect the text and its relative location and to create a table. We then sort the table and group columns that have the same block number and line number. Before finding the total and date, we convert the grouped data into a list of strings.

3. Identify the total amount

This separate function reads the list line by line and identifies if there is any match to a list of keywords used in receipts that represent ‘total’. Then, we compare this line with another set of keywords used in receipts that do not represent ‘total’, such as “sub total” or “total without GST”. We then extract the number from the line of text and store it in a list.

We then return the max of the list as our detected total.

This entire function will return 2 values, the total amount and the date from the receipt.

- Currently, we have not implemented date detection.

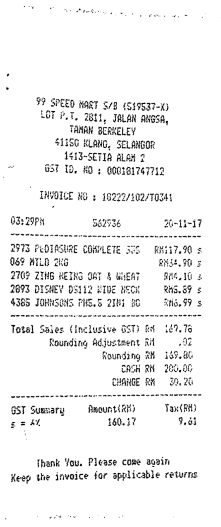
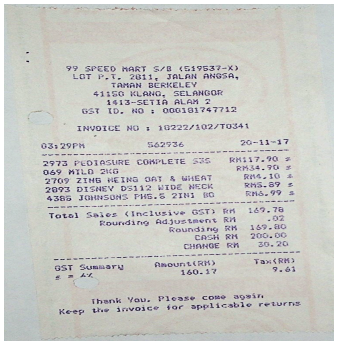
fullScaleTest.py is a test function that reads a group of images, extracts text from each image, and stores the results in a list. This test function also reads the corresponding solution data file and stores it in a separate list.

We then use these lists to generate a CSV file. By analyzing the CSV file, we can identify errors made during text detection, unreadable images, and calculate the overall accuracy.

3. **Proof of Accomplishment**

Key Accomplishment 1 proof:



99 SPEED MART S/B (519537-X)
LOT P.T. 2811, JALAN ANGSA,
TAMAN BERKELEY
41150 KLANG, SELANGOR
1413-SETIA ALAM 2
GST ID. NO : 000181747712

INVOICE NO : 10222/102/T0341

03:29PM 562936 20-11-17

2973 PEDIASURE COMPLETE 325	RM117.90 s
069 MILO 2KG	RM34.90 s
2709 ZING HEING OAT & WHEAT	RM4.10 s
2893 DISNEY DS112 WIDE NECK	RM5.89 s
4386 JOHNSONS PMS.S 2IN1 BO	RM6.99 s
Total Sales (Inclusive GST) RM	169.76
Rounding Adjustment RM	.02
Rounding RM	169.80
CASH RM	200.00
CHANGE RM	30.20

GST Summary	Amount(RM)	Tax(RM)
S = AX	160.17	9.61

Thank You. Please come again
Keep the invoice for applicable returns

This is image 010.JPG in our database, and the images show the pipeline running modularly from left to right. Firstly, it's the base image, then it is grayscale, then it is binarized using Otsu, and lastly it is deskewed. For this example the deskew degree is -3. The image at the bottom row is an example of our rescale function being run directly onto the base image.

Key Accomplishment 2 proof:

```python
# 2) Hyper-parameter grid
grid = {
    'clip_limit':       [1.0, 2.0, 3.0],
    'tile_grid_size':   [(8, 8), (16, 16)],
    'bin_method':       ['otsu', 'adaptive'],
    'block_size':       [11, 15, 21],
    'C':                [2, 5, 10],
    'font_size_thresh': [12, 14, 16],
    'small_scale':      [1.5, 2.0, 2.5],
    'large_scale':      [0.75, 1.0, 1.25],
}
```

```
▶ Best hyper-parameters (mean accuracy):
   • C               : 2
   • bin_method      : otsu
   • block_size      : 11
   • clip_limit      : 2.0000
   • font_size_thresh : 16
   • large_scale     : 1.2500
   • small_scale     : 1.5000
   • tile_grid_size  : (8, 8)
   • accuracy        : 0.6111
```

```
▶ Best hyper-parameters (mean accuracy):
   • C               : 5
   • bin_method      : otsu
   • block_size      : 21
   • clip_limit      : 2.0000
   • font_size_thresh : 16
   • large_scale     : 1.0000
   • small_scale     : 1.5000
   • tile_grid_size  : (16, 16)
   • accuracy        : 0.6587
```

Here are two examples of our cross validation function being run. The left is run with 20 unique receipts from the database, 3 folds, and 10 unique parameter combinations. The right one is run with 20 unique receipts from the database, 5 folds and 30 unique parameter combinations. Along with a screenshot of our parameter grid that we are using currently, which is likely to be improved on. These tests differ in results due to neither of them being able to exhaust all the combination possibilities. Also to be noted the accuracy, in this test is much higher then our FullScaleTest due to the much lower test sample size.

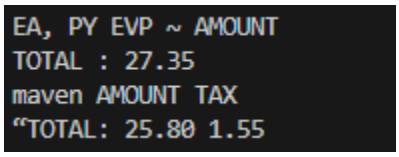
Key Accomplishment 3 proof:

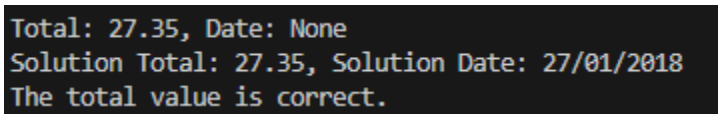
After pre-processing, the image_to_data() function generated this table, which contains the real and relative locations of the text in the image.

	level	page_num	block_num	par_num	line_num	word_num	left	top	width	height	conf	text
16	5	1	3	1	1	1	852	271	54	33	79.802917	TEO
17	5	1	3	1	1	2	919	270	77	33	84.607506	HENG
18	5	1	3	1	1	3	1008	261	128	85	67.238937	STATON
19	5	1	3	1	1	4	1150	269	42	32	81.882103	RY,
23	5	1	4	1	1	1	1032	310	55	37	0.000000	coors
..	...	...	...	...	...	...	...	...	...	...	...	...
245	5	1	26	1	1	3	1053	1901	44	19	95.136887	ARE
246	5	1	26	1	1	4	1106	1901	45	18	92.606636	NOT
247	5	1	26	1	1	5	1160	1900	108	19	91.918213	RETURNAI
249	5	1	26	1	2	1	1038	1933	71	18	96.686417	THANK
250	5	1	26	1	2	2	1122	1932	49	18	92.064102	YOU.

[139 rows x 12 columns]

When we detect text, it will only process the lines that contain the target keywords and exclude those with “misleading” keywords, such as “Sub-total”. This will help find the correct total.





4. **Challenges or Roadblocks**

Some roadblocks we have faced are running into long training times when using the Cross Validation training. Due to this, our model with less parameters works with higher accuracy. To resolve this issue, we plan to test parameters isolated rather than every individual combination to find the optimal parameter combination. Another issue is a preprocessing function we were implementing called Rescale which rescales the image to zoom in on the important section lowers the accuracy immensely. Based on our tests the accuracy dropped from 55 percent to 26 percent. Currently we have excluded this from our pipeline however we plan to further experiment and play around with it this week.

5. **Changes from Original Plan**

In our proposal we did not plan to have a test for accuracy, however based on feedback given on our last milestone we implemented it as a way to test the efficiency of our system. Another unplanned change we implemented cross validation. We did this in an attempt to increase the accuracy of our system which it has not yet, however we believe it has potential with a better cross validation implementation. Lastly, using a text detection(i.e. bounding boxes with CRAFT) separate from recognition was not deemed necessary by our group at this time.