- Python Version: 3.6.5

High-Level Approach

Since the hOCR files are good representation of the *.pdf files. For this work I have relaxed certain operations. Such as there could be mistakes in the file and has to be taken care. (Note: Again few errors are there in the hOCR. e.g: “Invoice Number” in 4th PDF has detected 1 instead of l). This mistakes can be taken care at the pre processing point by applying corrections. For example if the first letter of invoice number denotes the type (e.g l for late, o for on time) then 1 can be converted to l and 0 can be convered to O.

Problem Statement:

Given a file extract `KEY:VALUE` pairs from the file with some confidence score *S{i}* and provide training/feedback mechanism for correction of mistakes.

Solution:

For the given samples of document and assuming to be coming homogeneously from the population it can be assumed that the rule would be more or less rigid for same templates. Here the documents are generated via computers and have fixed format.

Level 0:

Is the document valid

- In many cases the document could be in other language. In such cases it is necessary to convert the document to **ENGLISH**.
- This could be done via using naive-bayes approach or something of similar level as the accuracies are already saturated for such jobs.

Level 1:

Extracting the data from the document

- To extract data from the document, libraries such as `beautifulsoup` could be used. Each line could be stored as an `OBJECT` with properties such as the value, position in the document(Page Number, Co-

ordinates).

- We call this independent line (`textBox`) as an element of `textBlob` .

Level 2:

Finding out the keys in the document

- The VALUES could only be correctly identified if the KEYS are correctly located in the document. To locate KEYS in the document similarity check could be done on the textBox extracted above.
- The similarity check will give us all the possible candidates for the KEYS.
- *Problems:*
    - For example while searching for “INVOICE NUMBER” many field like “Invoice, Invoice Date, Invoice No” would come as possible candidate
- *Workaround:*
    - In order to eliminate the false positives, we can give weightage to the keys to look for and the keys not to look for.
    - `KEY_INVOICE_NUMBER = {"invoice": 0.7, "number": 0.25, "No": 0.2, "#": 0.2, ":": 0.23, "date": -0.7}`
- Similarity could be measured by calculating the weighted sum of keys-score on word level. Other methods of similarity were too complex and not suitable here. e.g. Weighted score for char-level and word-level similarity

Level 3:

Finding the Values

- From Keys
    - Its possible that the textBox having the key is also having the VALUE. e.g: “Invoice No. : 23453”
    - In the case mentioned above we look for such possibility and try to extract Value if it seems to be there
    - One way to find out if the value is there or not is by using the symbol “:”. In many cases “:” denotes the next
      horizontal precedence as the VALUE for the key.
    - If the key is not there we move to guess it from the near textBox available.
    - It could be made a rule at this level about discarding the values not relevant based on the position of textBox.

E.g it is irrelevant to have a total amount on the top right corner. (For the sample document given.)

- From Near Regions
    - We can start looking at the nearest regions one by one and would only proceed to the next region if the VALUE isn't found.
    - In order to look into the nearest regions we need to rank them in a manner that the textbox containing the value comes up to the top.
    - This could be done by using Euclidean Distance and static rules like the value should be on the right, bottom or at a point with a static rule defining the preferences. This rules can be formulated to work according to the sample we have.

Level 4:

Automating the process, Accuracy, Feedback mechanism and other factors

- The values for threshold in rules such as the weightage for words in the keys, the relaxation in error in marking the hOCR box, and thresholds for static rules could be learned from the documents and could be used.
- This learning can also be made user interactive to take feedback and change the values. This will allow user to kind of train the model in long run and automate the process.
- The accuracy depends on many factor and I personally prefer to give a score that is the weighted dot product of different steps introduced in the extraction process.

Other Enhancements:

- Using Database or K-d tree to store textBlob for faster queries to extract the nearest region and text.
- Using Multitheading on file level / key level to speed up the extraction.

Other Details:

- Extracting relevant tables:
    - Relevant tables could be extracted using <row> and <cell> identifiers present in the hOCR output. To further get specific tables related to the quantity and purchase history the total amount could be used.
    - For most of the use case the total amount is on the right end of the table in consideration. This information could be used to extract the right table.
    - For this usecasse the same fucntion for invoice numeber was good enough for invoice date, payment terms and amount to be extracted. Hence no extra effort has been taken to consider

that.