17 April 2019

Optical Character Recognition Post-Processing

Group 8

Steps

1. Data Pre-Processing
2. Error Detection: Binary Digrams
3. Error Correction: Topic Modelling
4. Performance Measure

Error Detection: Binary Digrams

- Construct a dictionary using the ground truth data, and then use the dictionary to detect errors
- Dictionary is formed using positional binary digrams
- Binary digrams are sparse 26×26 matrices with entries being either 0 or 1

Error Detection: Binary Digrams

- For each possible word length n, we have $\frac{n(n-1)}{2}$ binary digrams for each possible pair of letter positions
- For example, for $n = 3$, there are 3 binary digrams for the letter positions (1,2), (1,3) and (2,3)
- Adding the word 'bat' to the dictionary means that we enter 1 for (2,1) in the 1st matrix, (2,20) in the 2nd, and (1,20) in the 3rd

Error Detection: Binary Digrams

- We add all the words in the ground truth files to our dictionary
- Errors are detected by comparing each pair of letter positions in each word from the OCR files to the dictionary
- If we detect 'btu', we may find that the (2,20) entry in the 1st matrix for $n = 3$ is 0

Error Correction: Topic Modelling

For each detected error, we refer to candidate words of the same length which are at most 2 letters different, and give each candidate a score,

$$\mathrm{P}\left(w\right)\prod_{i=1}^{n}\mathrm{P}\left(l_i^f \mid l_i^s\right) \tag{1}$$

where $\mathrm{P}\left(w\right)$ is the probability of the word, and $\mathrm{P}\left(l_i^f \mid l_i^s\right)$ is the probability that the OCR reads 'f' given that the actual letter is 's'

Error Correction: Topic Modelling

- Topic modelling provides us with more information in each document about $\mathrm{P}(w)$
- For example, 'tonque' can be corrected as 'tongue' or 'torque' based on the content of the document
- We used an LDA topic model which provides us with:
  1. Distribution of words for each topic
  2. Distribution of topics for each document

Error Correction: Topic Modelling

Using our topic model, we can calculate the probability of the candidate word,

$$\mathrm{P}\left(w\right) = \sum_{k=1}^{K} \mathrm{P}\left(w \mid t_k\right) \mathrm{P}\left(t_k\right) \tag{2}$$

where we have K number of topics, $\mathrm{P}\left(w \mid t_k\right)$ is the probability of the word given each topic, and $\mathrm{P}\left(t_k\right)$ is the probability of each topic.

Performance Measure

We used unique words in each document to segment the document, and then compare the differences in each segment to calculate precision and recall:

	Tesseract	**With post-processing**
Word wise recall	0.63110	0.70789
Word wise precision	0.62249	0.69961
Character wise recall	0.72345	0.72536
Character wise precision	0.71739	0.71966