PR – Exercise 3 – Keyword Spotting Outline

- Problem and approach
- Tools
- Preprocessing
- Feature extraction
- Testing and results

Louis Müller and Sebastian Graf
Pattern Recognition, 2018, UniFr

Problem and overall approach

- Keyword spotting in handwritten text
- Dataset: Washington DB containing handwritten documents of George Washington
- Word segmentation → one image each word
- Image binarization, white space cropping and overall width & height normalization
- Sliding window method for feature extraction
- Compute DTW distance between sequences of feature vectors

Tools

- .. we finally switched to Python
- **mply** for DTW implementation
- **Pillow** and **OpenCV** for image processing and MinMax feature value range scaling

Preprocessing (½)
Images

- Segmented word (png)
- Binarized and white border cropped
- Resized down to 120 x 30

Preprocessing (2/2) Considerations

- Resizing images on equal width & height, eg.: 200x200, didn't work for our feature set
  - The contrast in images is already very decent and this made binarization straightforward
- We did not deal with words with commas, points, punctuation in general

Feature extraction

- Method: 1 px sliding window from left to right
- Out of each 1px column of b&w pixels we computed:
  - Overall fraction of black pixels
  - Position of first black pixel, from above. *uc* (upper contour)
  - Max number of consecutive black pixels
  - Number of b&w transitions
- Feature left out:
  - Difference between *previous uc* and *current uc*

Testing (1/3)
Choices

- DTW needs as train sample for each keyword search and we pick simply the first example we get out of the Transcript.txt
- We decided to concentrate on the recall rate in order to try to spot all occurrences
- We compute the recall rate out of a top 50 nearest spotted keywords

Testing (2/3)
Results

- Even considering a top 50 list of nearest spotted keywords we have not a 100% recall rate
- Using the entire keyword set we remain on a recall rate which is around **79%**
- I'd not consider our precision result because it is not meaningful: for each word we spot, we have top 50 list of nearest words, and 50 goes beyond the occurrences of many numbers in our set

Testing (3/3)
Results

- Considering a top 20 list (because there is a word which occurs 20 times) we get a recall rate of **70%** and this somehow tells us that DTW is working
- Concluding, about performance:
  - having images of 120x30
  - 4 features for each of the 120 columns

  the computation time for a complete run is not that quick: ~45min for the entire keyword.txt (107 words) on all images (3'726), on an 5 year old i7-3635MQ