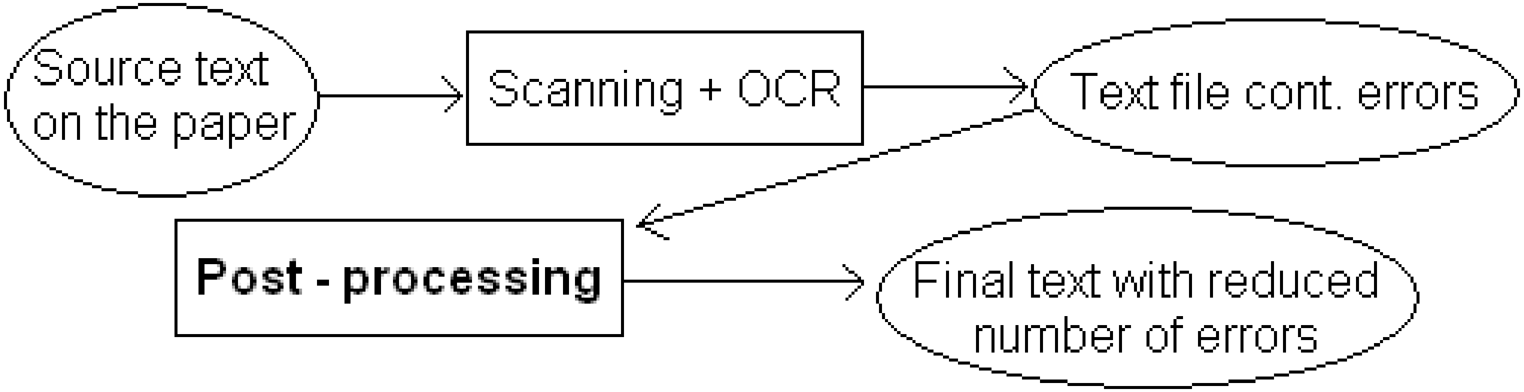
# OCR Post-Processing

Michal Richter

# Noisy channel approach I

- Scanning of the document and OCR introduce errors – noise
- Post – processing step reduce the number of errors



# Noisy channel approach II

- Post – processing corrects one sentence at the time.
- OCR output is modified by small amount of editing operations including:
  - single character insertion
  - single character deletion
  - single character substitution
  - multiple character substitution ( ab → ba )
  - word split, word merge

# Intuitive describtion

- In post-processing we want to replace the input sequence of characters with another sequence of characters that is graphically similar and form the likeable sentence of the given language
- These two aspect are handled separately

# General form of the model

$$P(O, S) = P(O \mid S) * P(S)$$

O – output of the OCR system

S – candidate sequence of character

P( O | S ) – probability, that the sequence S will be recognized as O by OCR – corresponds to optical similarity between O and S – usually denoted as *error model*

P( S ) – probability of S – corresponds to the likeabelness of the sequence S – this quantity should have greater value for well-formed sentences – denoted as *language model*

# Language model – P( S )

- Word based
  - Uses lexicon – sequence of characters is identified with the item in the lexicon
  - Smoothness of the sentence is ensured by word based n-gram model ( usually trigram )
  - Problem: High coverage lexicon and huge amount of on-line text needed ( for n-gram model estimation )

$$P(w_1,\ldots,w_m) = \prod_{i=1}^{m} P(w_i|w_1,\ldots,w_{i-1}) \approx \prod_{i=1}^{m} P(w_i|w_{i-(n-1)},\ldots,w_{i-1})$$

# Language model – P( S )

- Character based
  - Smoothness of the sentence is ensured on the character level
  - No need of lexicon, lower amount of training data needed for language model estimation
  - Character based language model used

  (even 6-gram is possible)

# Error model – P( O | S )

- Levenshtein distance
  - Number of insertions, deletions and substitutions needed to transform input into the target
  - Example: LD between kitten and sitting is 3

  kitten → sitten → sittin → sitting
- Modified Levenshtein distance
  - Editing operations have different costs according to their probability
  - Example: low cost for in ↔ m, high cost for w ↔ R

# Error model – P( O | S )

- Word segmentation
  - Can be treated by word segmentation model

$$P(O, b, a, C) = P(O, b|a, C)P(a|C)P(C)$$

  - Another possibility is to avoid special treatment of the space character – word segmentation errors are corrected via insertion/deletion of space character

# Search of the correct sentence S

- Viterbi decoding
- Weighted Finite State Transducers
  - Language model and error model are represented in the form of finite state transducers
  - Make the composition of the automaton representing OCR output with the automaton representing error model and language model
  - Find the shortest path in the composed transducer
  - *blackboard?*

# Post-correction accuracy measure

- Word error rate metric

$$WER(C,O) = \frac{WordEditDistance(C,O)}{WordCount(C)}$$

# Post-correction accuracy

- (Kolak, Resnik; 2005)
  - WER reduction up to 80%
  - African language Igbo
  - Character based model
  - Miniature size training data – 6727 words!

# Post-correction for historical domain

- Insufficient amount of training data ( if any )
- Usually absence of high-coverage lexicons

→ This implies, that the use of word based approach is often impossible

# References

Okan Kolak; Philip Resnik. OCR Post-Processing for Low Density Languages. EMNLP-2005.