The LITIS arabic handwriting recognition system

Lattice-based Combination Framework for HMM-based Handwriting Recognition Systems

Kamel Ait Mohand and Thierry Paquet

LITIS laboratory, University of Rouen

OpenHaRT workshop
August 23, 2013
Summary

1. Introduction

2. Pre-processing

3. Baseline system

4. Combination of systems

5. Conclusion
OpenHaRT

- LITIS laboratory, Rouen, France
- DIR task, OpenHaRT 2013 competition
- Constrained and LINE segmentation condition
- Two systems submitted:
  - baseline system based on Hidden Markov Models
  - combination of the outputs of several systems (Primary)
Summary

1. Introduction
2. Pre-processing
3. Baseline system
4. Combination of systems
5. Conclusion
Pre-processing

- First: extract line images (coordinates from MADCAT segmentation files)
- Process line: image → set of feature vectors
- Pre-processing chain:
  - Image quality enhancement:
    - Wiener and bilateral filtering
    - Contrast enhancement
    - Mathematical morphology operations (noise removal)
  - Adaptive binarization (Sauvola algorithm)
  - "Normalize" style of writing:
    - Deskew
    - Deslant
    - Size normalization
**Line deskew**

**Principle**

- **Correction of the line slope** *(deskew)*
  - skewed line image
  - find extrema points
  - estimate line slope
  - slope correction *(rotate line in the opposite direction)*

**Illustration**

[Image of a handwritten line with corrections applied]
Line deskew

**Principle**
- Correction of the line slope (*deskew*)
  - skewed line image
  - find extrema points
  - estimate line slope
  - slope correction (rotate line in the opposite direction)

**Illustration**

![Illustration of deskewed text](image)
Line deskew

**Principle**
- Correction of the line slope (*deskew*)
  - skewed line image
  - find extrema points
  - estimate line slope
  - slope correction (rotate line in the opposite direction)

**Illustration**
Line deskew

**Principle**

- Correction of the line slope (*deskew*)
  - skewed line image
  - find extrema points
  - estimate line slope
  - slope correction (rotate line in the opposite direction)

**Illustration**

العراقية والأنجليزية خلال الشهر
Line deslant

**Principle**
- Estimate the average slope angle of the characters:
  - histogram of the directions of Freeman contour
- Slope correction by a linear transformation:
  - shift each foreground pixel depending on its position

**Illustration**
Line deslant

**Principle**
- Estimate the average slope angle of the characters:
  - histogram of the directions of Freeman contour
- Slope correction by a linear transformation:
  - shift each foreground pixel depending on its position

**Illustration**

العراقيين وأهل المكسيك تشارك الآلية
**Size normalization**

**Principle**
- Normalization of the line height
- Interpolation (Sinc kernel, “Lanczos”)
- Standard value of 48 pixels
- Purpose: homogeneity of lines content

**Illustration**

![Image of handwritten text](image-url)
Feature extraction

Procedure

- Sliding window approach (no explicit segmentation)
- For each window position:
  - 128 features histogram of gradient orientation
    - 4 × 4 grid
    - 8 discrete values for the gradient orientation
    - total: 128 features
  - 5 features for position and size of the connected components
  - Finally: 133 features
- Good performance on latin script (arabic?)

![Diagram of feature extraction process](image)
Summary

1. Introduction
2. Pre-processing
3. Baseline system
4. Combination of systems
5. Conclusion
Character modelisation

- Primarily designed for latin script (no adaptation for arabic)
- One character = one Hidden Markov Model (HMM)
  - left-right continuous
  - mixtures of Gaussians data modelisation
- 144 characters :
  - contextual Arabic letters
  - digits
  - punctuations
  - inter-word space

left-right HMM
Hidden Markov Models

- **Hidden Markov Model**: a set of $N$ states, a mixture of $G$ gaussians for each state, parameters: transition prob., Gaussians $\mu$ and $\Sigma$

- **Train HMMs**: find the best structure (define $G$ and $N$)
  - heuristic method of Zimmermann and Bunke
  - estimate the parameters values
    - Baum-Welch algorithm

- **Optimal values**:
  - number of states: from 8 to 24
  - $G = 20$
  - 20 Baum-Welch iterations
Recognition phase

- Process the image (pre-processing, feature extraction)
- Recognition engine:
  - set of HMM models
  - arabic lexicon (64,000 words)
  - n-gram language model estimated on a 10,000,000 words corpus
- decoding with a two-pass forward-backward search
  - 1st pass: frame-synchronous beam search algorithm (2-gram)
  - 2nd pass: stack decoding search (3-gram)
- running time: less than 2 minutes on an average for one document
Baseline system results

- Origin of errors:
  - insufficient discriminating capabilities of mixtures of Gaussians
  - language modelisation problem:
    - bad lexicon (31.1% OOV)
    - small words concatenation (caused by the language model)
  - Rule-lines (ex.) : 14.30% vs. 27.71% WRR
  - Overlapping lines (ex.)
  - word segmentation errors (line-level recognition)

<table>
<thead>
<tr>
<th></th>
<th>NEWSWIRE</th>
<th>WEB</th>
<th>ALL</th>
</tr>
</thead>
<tbody>
<tr>
<td>1-WER</td>
<td>0.2354</td>
<td>0.2467</td>
<td>0.2409</td>
</tr>
</tbody>
</table>
The LITIS arabic handwriting recognition system

Kamel Ait Mohand and Thierry Paquet

Summary

1. Introduction
2. Pre-processing
3. Baseline system
4. Combination of systems
5. Conclusion
Word lattices and confusion networks

**Word lattice**

- Structured representation of N-best recognition hypotheses
- Each word (node) has:
  - word confidence score
  - time boundaries
Word lattices and confusion networks

**Confusion network (CN)**

- Weighted directed graph, compact representation of lattices
  - competing hypotheses organized in different sets (nodes)
  - words in sets are sorted by their scores
  - each set can also contain one empty word (ε)
- Decoding: select first word (highest probability) in each set
Systems combination

Principle

- **Principle**: combine outputs of several recognition engines
- **Procedure**:
  - take into account the N-best sequences of each system
  - extract lattices of each system and merge them
  - convert obtained lattice to a confusion network
- **Advantage of converting a lattice to a CN**:
  - create new paths with words from different engines
  - reinforce “good” word hypotheses
- **Still under development...**
Combination procedure

**Successive operations**

- run the recognition for several recognition systems
- output a word lattice for each system
- vertically concatenate lattices (merge their start and end nodes)
- weight the scores of each hypothesis (different weights for each system)
Combination procedure

Successive operations

- run the recognition for several recognition systems
- output a word lattice for each system
- vertically concatenate lattices (merge their start and end nodes)
- weight the scores of each hypothesis (different weights for each system)
Combination procedure

Successive operations

- run the recognition for several recognition systems
- output a word lattice for each system
- vertically concatenate lattices (merge their start and end nodes)
- weight the scores of each hypothesis (different weights for each system)
Combination procedure

Successive operations

- run the recognition for several recognition systems
- output a word lattice for each system
- vertically concatenate lattices (merge their start and end nodes)
- weight the scores of each hypothesis (different weights for each system)
Combination procedure

Successive operations

- run the recognition for several recognition systems
- output a word lattice for each system
- vertically concatenate lattices (merge their start and end nodes)
- weight the scores of each hypothesis (different weights for each system)
Combination procedure

Conversion of combination lattice to a CN

- initialize with highest-probability path
- align remaining partial lattice paths to the CN
- rescore the words hypothesis (LM)
- decode the CN to get best path

Lattice

Confusion network
Combination procedure

Conversion of combination lattice to a CN
- initialize with highest-probability path
- align remaining partial lattice paths to the CN
- rescore the words hypothesis (LM)
- decode the CN to get best path

Lattice

Confusion network
Combination procedure

Conversion of combination lattice to a CN

- initialize with highest-probability path
- align remaining partial lattice paths to the CN
- rescore the words hypothesis (LM)
- decode the CN to get best path
Combination procedure

Conversion of combination lattice to a CN

- initialize with highest-probability path
- align remaining partial lattice paths to the CN
- rescore the words hypothesis (LM)
- decode the CN to get best path

Lattice

Confusion network
Combination procedure

Conversion of combination lattice to a CN

- initialize with highest-probability path
- align remaining partial lattice paths to the CN
- rescore the words hypothesis (LM)
- decode the CN to get best path

Lattice

Confusion network
Combination procedure

Conversion of combination lattice to a CN

- initialize with highest-probability path
- align remaining partial lattice paths to the CN
- rescore the words hypothesis (LM)
- decode the CN to get best path

Lattice

Confusion network
Combination procedure

Conversion of combination lattice to a CN
- initialize with highest-probability path
- align remaining partial lattice paths to the CN
- rescore the words hypothesis (LM)
- decode the CN to get best path

Lattice

Confusion network
**Combination procedure**

**Conversion of combination lattice to a CN**
- initialize with highest-probability path
- align remaining partial lattice paths to the CN
- rescore the words hypothesis (LM)
- decode the CN to get best path

**Lattice**

**Confusion network**
Combination procedure

Conversion of combination lattice to a CN

- initialize with highest-probability path
- align remaining partial lattice paths to the CN
- rescore the words hypothesis (LM)
- decode the CN to get best path
Combined systems

- Several different systems
- Outputs must be "complementary" : different classifiers or different feature extractors
- Lack of time : same classifier (HMM), same feature extractor
- Lack of time : different line sizes (normalization step)
  - 3 different image resolution values
  - get different HMM alignments on feature frames (different outputs)
- Long running time (N-best list extraction is time-consuming)
Combination results

- Low Recognition rate results (less than baseline)

- Errors due to:
  - Same problems than of baseline system
  - Better results if $N$ is high. But we only used $N = 3$
  - outputs of combined systems are too close

### Results

<table>
<thead>
<tr>
<th></th>
<th>NEWSWIRE</th>
<th>WEB</th>
<th>ALL</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Baseline</strong></td>
<td>0.2354</td>
<td>0.2467</td>
<td>0.2409</td>
</tr>
<tr>
<td><strong>Combination</strong></td>
<td>0.2189</td>
<td>0.2295</td>
<td>0.2241</td>
</tr>
</tbody>
</table>
Summary

1. Introduction
2. Pre-processing
3. Baseline system
4. Combination of systems
5. Conclusion
Conclusion

- Arabic handwriting recognition engine based on Hidden Markov Models
  - low accuracy on evaluation dataset
  - several improvements needed (language modeling, discriminative classifier, line-removal)
- Combination framework of systems outputs that uses word lattices
  - unfinished (lack of time...)
  - running time optimisation
  - develop complementary systems for a successful combination
Thank you for your attention