Deep Learning Feature for Handwritten Keyword Spotting

Baptiste Wicht    Andreas Fischer    Jean Hennebert

iCoSys, University of Applied Sciences of Western Switzerland HES-SO

DIVA Group, University of Fribourg, Switzerland
Who’s who

Deep Learning Feature for Handwritten Keyword Spotting

Baptiste Wicht
Deep Feature Extraction

Andreas Fischer
Keyword Spotting

Jean Hennebert
Enjoying Conference
Introduction - Research Questions

- Are deep learning features good for keyword spotting applications?
- Sub-questions:
  - Are such features robust for different systems?
    - template-based (DTW)
    - learning-based (HMM)
  - Does it work across very different handwritten inputs, i.e. historical 13th century docs to modern English handwriting?
  - Are such features better than state-of-the-art hand-crafted features?
  - How much cooking to get decent performances?

(a) IAM database
(b) George Washington database
(c) Parzival database
Introduction - Keyword Spotting System

Deep Features for Keyword Spotting

Keyword Query + Word Image

Deep Learning Feature Extractor

Unlabeled Data

DTW

Keyword Score

HMM

Keyword Score

Labeled Data

Correct spotting

False positive
Preprocessing

1. The system operates on segmented word images
   - binarized, normalized to remove the skew and slant
   - resized to a third of their height

2. Patches are extracted using an horizontal sliding window
   - no vertical overlap
   - move from left to right one pixel at a time
Restricted Boltzmann Machine

- Generative Stochastic Artificial Neural Network (ANN)
- Learn probability distribution over the inputs
- Trained with Contrastive Divergence
  - Similarly to gradient descent techniques
  - As an autoencoder
- Can reconstruct the features ($h$) from the input ($v$)
  - And the other way around
Convolutional RBM

- The layers are connected by convolution
- Input and outputs are matrices
  - 2D Image with $C$ channels as input
  - $K$ 2D feature maps as output
  - $N_W \times N_W$ pixels per patch
  - $[C \times K \times N_W \times N_W]$ weights
- The training principles are the same as for the RBM
Feature Extractor

- Two CRBM are stacked to form a Convolutional Deep Belief Network

- Max Pooling after each CRBM
  - To improve robustness of features
  - To reduce the number of features

- Normalization of the final features
  - Each feature group is one-sum normalized
  - Each feature is zero-mean and unit variance normalized
**Input:**
- A “target” keyword image $K$
- A “candidate” word image $X$

**Decision:** Does the candidate image matches with the keyword?
- Decided with a dissimilarity measure and a threshold
- If $ds(K, X) < T$ then accept the candidate $X$
Find an optimal alignment between two sequences of different length
- Warped non-linearly to match each other
- The cost of an alignment is the sum of the distances of aligned pairs
  - Normalized w.r.t. the warping path

Sakoe-Chiba band is used to improve the results
- Constrain the search within a band around the shortest path

Source: Wikimedia
Hidden Markov Model (HMM)

- Based on: Fischer et al. “HMM-based word spotting in handwritten documents using subword models”, ICPR 2010

1. One \( m \)-state HMM per character, left-right topology
2. Keyword model \( K \) is created by connecting character HMMs
3. A filler model \( F \) (unconstrained) is created in the same way

The dissimilarity is computed with both log-likelihoods measures

\[
ds(X, K) = \frac{\log p(X|F) - \log p(X|K)}{L_K}
\]
Experimental Evaluation

- Evaluated on three datasets
  - GW: 4894 word images, 1755, English, single-writer
  - PAR: 23485 word images, 13th Century, ancient German, single-writer
  - IAM: 70871 word images, modern English, multiple-writer
Experimental Evaluation

- Evaluated against three baselines
  - Marti2001: 9 heuristic features per column of the image
  - Rodriguez2008: local gradient histogram features (128-dimensional)
  - Terasawa2009: slit-style Histogram Of Gradients (HOG) features (384-dimensional)

- Performance is assessed using two measures:
  - Average Precision (AP): one global threshold
  - Mean Average Precision (MAP): one threshold per keyword

- The number of filters is the only parameter tuned for each data set
  - All other parameters are kept the same under all configurations

- Parameters of the classifiers are the same for all systems
  - Taken from: Fischer et al. “HMM-based word spotting in handwritten documents using subword models”, ICPR 2010
### DTW Results

<table>
<thead>
<tr>
<th>System</th>
<th>GW</th>
<th>PAR</th>
<th>IAM</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>AP</td>
<td>MAP</td>
<td>AP</td>
</tr>
<tr>
<td>Marti2001</td>
<td>33.24</td>
<td>45.26</td>
<td>50.67</td>
</tr>
<tr>
<td>Rodriguez2008</td>
<td>41.20</td>
<td>63.39</td>
<td>55.82</td>
</tr>
<tr>
<td>Terasawa2009</td>
<td>43.76</td>
<td>64.80</td>
<td>69.10</td>
</tr>
<tr>
<td>Proposed</td>
<td><strong>56.98</strong></td>
<td><strong>68.64</strong></td>
<td><strong>72.71</strong></td>
</tr>
<tr>
<td>Relative Improvement</td>
<td>23.20%</td>
<td>5.59%</td>
<td>4.96%</td>
</tr>
</tbody>
</table>

- **Results**
  - Better on GW than all the baselines
  - Comparable perf on PAR with best baseline (Terasawa2009)
  - IAM results can be ignored
    - DTW template matching is failing with different writing styles
### HMM Results

<table>
<thead>
<tr>
<th>System</th>
<th>GW</th>
<th>PAR</th>
<th>IAM</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>AP</td>
<td>MAP</td>
<td>AP</td>
</tr>
<tr>
<td>Marti2001</td>
<td>48.80</td>
<td>69.42</td>
<td>69.47</td>
</tr>
<tr>
<td>Rodriguez2008</td>
<td>32.60</td>
<td>59.40</td>
<td>25.43</td>
</tr>
<tr>
<td>Terasawa2009</td>
<td>68.01</td>
<td>79.49</td>
<td>90.50</td>
</tr>
<tr>
<td>Proposed</td>
<td><strong>71.21</strong></td>
<td><strong>85.06</strong></td>
<td><strong>92.34</strong></td>
</tr>
<tr>
<td>Relative Improvement</td>
<td>4.49%</td>
<td>6.54%</td>
<td>1.99%</td>
</tr>
<tr>
<td></td>
<td>7.76%</td>
<td>1.06%</td>
<td></td>
</tr>
</tbody>
</table>

- Outperforms every baseline in all tested situations

\[\text{GW:} \{48.80, 69.42, 69.47, 77.98\} \quad \text{PAR:} \{32.60, 59.40, 25.43, 32.53\} \quad \text{IAM:} \{68.01, 79.49, 90.50, 90.53\} \]

\[\text{AP:} \{16.67, 5.47, 59.66, 64.68\} \quad \text{MAP:} \{49.24, 21.11, 71.59, 72.36\} \]
System Optimization

- Optimization of the system has been challenging
  - Large number of parameters
  - Rather different datasets

- Training parameters
  - 25 epochs of Contrastive Divergence
  - **Sparsity for binary units**

- Architecture parameters
  - Two-layer models proved best
  - Sliding window of **20 pixels width**
  - **Number of filters**: 8 (GW) and 12 (PAR/IAM)
    - Very important for DTW
  - Units: Binary (GW) and ReLU (PAR/IAM)
Conclusion

- Proposed system outperforms 3 baselines on 3 data sets
  - Robust performance under all tested conditions
  - With purely unsupervised feature learning
  - Improvements on two different classifiers: DTW and HMMs
- Optimizing the model is non-trivial
  - Large number of parameters
  - DTW is “constraining” about the features
  - Still room for improvement
Future Work - Implementation

Future works

- Use grayscale normalized images
- Augment dataset with distortions
- Find a better configuration specific for HMM
- Score words with potentially better classifiers such as LSTM
- Compare with other auto-encoder types

Implementation

- Freely available online
- Keyword Spotting System (kws), C++
  - https://github.com/wichtounet/word_spotting
- Deep Learning Library (DLL), C++
  - https://github.com/wichtounet/dll

*URLs present in the paper*
Questions