Fast Script Word Recognition
With Very Large Lexica

Dr. Marc-Peter Schambach
Fast Script Word Recognition with Very Large Lexica
Overview

- What is script word recognition?
- What are very large lexica?
- What means fast?

- Standard algorithm
- New algorithm 1 (approximation)
- New algorithm 2 (exact)
Fast Script Word Recognition with Very Large Lexica

Introduction

We are talking about script word recognition

Mail sorting applications
Address reading: postal code, city, streets
Fast Script Word Recognition with Very Large Lexica

What are “Very Large Lexica”?

Berlin / Germany

Lexicon size: Up to 500,000
Fast Script Word Recognition with Very Large Lexica
What means “fast”?

2-Level-Sorter

Time for each document: max. 2 sec all together!
Fast Script Word Recognition with Very Large Lexica
HMM Standard Algorithm

The lexicon is a word list:
- Build HMM as trie from the lexicon
- Calculate
- Assign probabilities on the leaves to the lexicon

The lexicon is a pattern:
- Transform pattern into HMM
- Calculate
- Calculate best matching path, assign probability
- Calculate n-best matching paths
Fast Script Word Recognition with Very Large Lexica
2-step Algorithm
Fast Script Word Recognition with Very Large Lexica
Pruning algorithms

Forward Step
- Difference of node likelihood to best likelihood (at each vector)

\[ p_{\text{min}} = p_{\text{max}} / f_{\text{forw}} \]

Backward Step
- Difference of node probability to best likelihood (at each tree depth)

\[ p_{\text{min}} = p_{\text{max}} / f_{\text{back}} \quad (f_{\text{back}} \leq f_{\text{forw}}) \]

- Maximum number of active nodes
- Estimation of word length
Fast Script Word Recognition with Very Large Lexica
Approximation

Example: Wwwwwtown, liiitown
Solution: 3rd pass exact calculation on trie (best results)
## Fast Script Word Recognition with Very Large Lexica

### Results

<table>
<thead>
<tr>
<th></th>
<th>Postal codes</th>
<th>City names</th>
<th>Street names</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Images</strong></td>
<td>580</td>
<td>1047</td>
<td>1007</td>
</tr>
<tr>
<td><strong>Lexicon = 100</strong></td>
<td>93.6 %</td>
<td>93.6 %</td>
<td>94.9 %</td>
</tr>
<tr>
<td><strong>Lexicon = 1000</strong></td>
<td>88.1 %</td>
<td>87.9 %</td>
<td>92.2 %</td>
</tr>
<tr>
<td><strong>Lexicon size</strong></td>
<td>27540</td>
<td>21757</td>
<td>478123</td>
</tr>
<tr>
<td><strong>Pattern</strong></td>
<td>[0-9] {5}</td>
<td>[a-Z] {1-15}</td>
<td>[a-Z] {1-25}</td>
</tr>
<tr>
<td><strong>Exact (trie)</strong></td>
<td>54.1 %</td>
<td>63.1 %</td>
<td>59.1 %</td>
</tr>
<tr>
<td></td>
<td>37.8 ms</td>
<td>178.3 ms</td>
<td>2288 ms</td>
</tr>
<tr>
<td><strong>Nominal (pattern)</strong></td>
<td>44.1 %</td>
<td>4.3 %</td>
<td>2.5 %</td>
</tr>
<tr>
<td></td>
<td>3.5 ms</td>
<td>18.7 ms</td>
<td>26.4 ms</td>
</tr>
<tr>
<td><strong>Approximation (pattern &amp; trie)</strong></td>
<td>53.8 %</td>
<td>55.1 %</td>
<td>48.1 %</td>
</tr>
<tr>
<td></td>
<td>4.3 ms</td>
<td>44.2 ms</td>
<td>135.7 ms</td>
</tr>
<tr>
<td><strong>Approximation &amp; postprocessing</strong></td>
<td>54.2 %</td>
<td>60.0 %</td>
<td>51.4 %</td>
</tr>
<tr>
<td></td>
<td>5.7 ms</td>
<td>52.0 ms</td>
<td>145.2 ms</td>
</tr>
</tbody>
</table>
Fast Script Word Recognition with Very Large Lexica

Conclusion

- Slightly lower recognition rates
- Significantly reduced computation times
- New application areas
Fast Script Word Recognition with Very Large Lexica

Summary

- Recognition with very large lexica
  Tested with lexicon size up to 500000

- Presentation of a 2-step algorithm
  Calculate probability trellis on pattern
  Extract results with lexicon trie

- Speedup / accuracy trade-off
  Gains for densely populated patterns

- Usage for new application domains
  Address recognition with missing postal codes
Thank you!