The Illegal Character Problem

What are illegal characters?
- emojis, special punctuation marks, full-width characters, mathematical symbols, Greek letters, scientific notations...
- depends on task, language and domain

Why remove them from training data?
- drop noisy sentences
- reduce vocabulary size

How is it done so far?
- tiresome and repetitive scripting in a case-by-case fashion
- all WMT18 corpus filtering systems are rule-based

Modeling Clean Data

Why BGMM?
- classic and powerful algorithm to describe data and find clusters
- infer number of components $K$ from data
- good convergence with feature vectors in reduced dimensions $B' 
  \begin{align*}
  p(e) &= \sum_{k=1}^{K} \pi_k \mathcal{N}(e; \mu_k, \Sigma_k) \\
  \pi_k &\sim \mathcal{DP}(\alpha) \\
  \mu_k &\sim \mathcal{N}(\mu_k; \sqrt{\frac{1}{K}}) \\
  \Sigma_k &\sim \mathcal{W}(\Sigma_k; \nu, \nu)
  \end{align*}

Why clustering instead of classification?
- difficult to obtain negatively labelled data
  - unrealistic to cover all types of noises with synthetic data
  - not appropriate to deem all sentences in a noisy corpus dirty
- easy to obtain positively labelled data
- development set is often available
- can be deemed clean with relatively high confidence

Scoring and Filtering Corpus

How are the scores used for filtering?
- the weighted log probability densities are directly used as scores
- versatile filtering on monolingual or parallel data, with e.g.
  - absolute thresholds
  - relative thresholds
  - the minimum scores seen during training
  - linear combinations of scores across languages

Experiments

Sentence Analysis on STS

<table>
<thead>
<tr>
<th>Language Modeling on WMT19</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Test Accuracy [%])</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>filtered</th>
<th>zh</th>
<th>en</th>
<th>ru</th>
</tr>
</thead>
<tbody>
<tr>
<td>0%</td>
<td>40.9</td>
<td>135.0</td>
<td>181.0</td>
<td></td>
</tr>
<tr>
<td>10%</td>
<td>36.9</td>
<td>133.5</td>
<td>176.7</td>
<td></td>
</tr>
<tr>
<td>20%</td>
<td>37.1</td>
<td>132.9</td>
<td>180.6</td>
<td></td>
</tr>
<tr>
<td>30%</td>
<td>38.7</td>
<td>131.4</td>
<td>178.6</td>
<td></td>
</tr>
</tbody>
</table>

Machine Translation on WMT19

<table>
<thead>
<tr>
<th>(Case-sensitive BLEU [%])</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th></th>
<th>zh-en</th>
<th>en-zh</th>
<th>ru-en</th>
<th>en-ru</th>
</tr>
</thead>
<tbody>
<tr>
<td>filtered test17 test18</td>
<td>25.0</td>
<td>25.4</td>
<td>30.1</td>
<td>33.0</td>
</tr>
<tr>
<td>0%</td>
<td>25.2</td>
<td>25.6</td>
<td>30.9</td>
<td>33.1</td>
</tr>
<tr>
<td>10%</td>
<td>24.3</td>
<td>24.8</td>
<td>30.2</td>
<td>33.0</td>
</tr>
</tbody>
</table>

Takeaways?
- a method to get meaningful scores for "character legality"
- an open-sourced tool for corpus filtering

Constructing Feature Vectors

Unicode block?
- Unicode -> de facto standard for character encoding
- characters with similar origins or functions are grouped in blocks

How exactly is a feature vector constructed?
- consider "我爱NLP!"
  - "我" -> CJK Unified Ideographs block
  - "爱" -> Basic Latin block
  - fixed width count vector \( [4, 0, 0, ...] \)
  - dimensionality reduction \( [4, 2] \)
  - normalize the counts \( [0.67, 0.33] \)
  - supports custom block ranges: e.g. ASCII digits and punctuations
  - additional features: e.g. total character count, word count

Acknowledgments

This work has received funding from the European Research Council (ERC) (under the European Union’s Horizon 2020 research and innovation programme, grant agreement No 645578) project "SIDELINE" and the Deutsche Forschungsgemeinschaft (DFG, grant agreement NE 572/8-1) project "CoreTec". The GPU cluster used for the experiments was partially funded by DFG (Grant WE 1958/ 1).