CHARACTER WORD EMBEDDING AND POS TAGGING FOR INDIAN LANGUAGES

Anirban Majumdar
Amit Kumar
October 15, 2015

Indian Institute of Technology Kanpur
MOTIVATION
· Distributed word representations are proven to be a powerful tool.
· Word embeddings captures syntactic and semantic information about word.
· In task like POS Tagging intra-word information could be very useful which is ignored in word embeddings.
· Character embeddings can be use to capture the intra-word information [1].
· Why not enhance the word embedding to use intra-word information by using character embedding.
- Learning Character-level Representations by Santos et al.
- Some results on english language

<table>
<thead>
<tr>
<th>RELATED WORK</th>
</tr>
</thead>
<tbody>
<tr>
<td>· Learning Character-level Representations by Santos et al.</td>
</tr>
<tr>
<td>· Some results on english language</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>INCONSIDERABLE</th>
<th>83-YEAR-OLD</th>
<th>SHEEP-LIKE</th>
<th>DOMESTICALLY</th>
<th>UNSTEADINESS</th>
</tr>
</thead>
<tbody>
<tr>
<td>INCONCEIVABLE</td>
<td>43-YEAR-OLD</td>
<td>ROCKET-LIKE</td>
<td>FINANCIALLY</td>
<td>UNEASINESS</td>
</tr>
<tr>
<td>INDISTINGUISHABLE</td>
<td>63-YEAR-OLD</td>
<td>FERN-LIKE</td>
<td>ESSENTIALLY</td>
<td>UNHAPPINESS</td>
</tr>
<tr>
<td>INNUMERABLE</td>
<td>73-YEAR-OLD</td>
<td>SLIVER-LIKE</td>
<td>GENERALLY</td>
<td>UNPLEASANTNESS</td>
</tr>
<tr>
<td>INCOMPATIBLE</td>
<td>49-YEAR-OLD</td>
<td>BUSINESS-LIKE</td>
<td>IRONICALLY</td>
<td>BUSINESS</td>
</tr>
<tr>
<td>INCOMPREHENSIBLE</td>
<td>53-YEAR-OLD</td>
<td>WAR-LIKE</td>
<td>SPECIALLY</td>
<td>UNWILLINGNESS</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>INCONSIDERABLE</th>
<th>00-YEAR-OLD</th>
<th>SHEEP-LIKE</th>
<th>DOMESTICALLY</th>
<th>UNSTEADINESS</th>
</tr>
</thead>
<tbody>
<tr>
<td>INSIGNIFICANT</td>
<td>SEVENTEEN-YEAR-OLD</td>
<td>BURROWER</td>
<td>WORLDWIDE 000,000,000</td>
<td>PARESTHESIA 0.000000</td>
</tr>
<tr>
<td>INORDINATE</td>
<td>SIXTEEN-YEAR-OLD</td>
<td>CRUSTACEAN-LIKE</td>
<td>HYPEROSALVATION 0.000</td>
<td></td>
</tr>
<tr>
<td>ASSUREDLY</td>
<td>FOURTEEN-YEAR-OLD</td>
<td>TROLL-LIKE</td>
<td>DROWSINESS 0.000000</td>
<td></td>
</tr>
<tr>
<td>UNDESERVED</td>
<td>NINTEEN-YEAR-OLD</td>
<td>SCORPION-LIKE</td>
<td>SALES</td>
<td></td>
</tr>
<tr>
<td>SCRUPLE</td>
<td>FIFTEEN-YEAR-OLD</td>
<td>UROHIDROSIS</td>
<td>RETAILS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>BREATHELESSNESS -0.00</td>
</tr>
</tbody>
</table>
GOAL
∙ Learning intra-word feature extraction of words using character embedding.
∙ Enhancing word embedding using the character embedding of the word.
∙ Using enhanced word embedding to perform task like POS Tagging.
CHALLENGES
CHALLENGES

- Character embedding relatively new field.
- Extracting the morphological information from character embedding
- Use of Enhanced word vectors for NLP tasks such as POS tagging in Indian Languages like Hindi, Bengali
ROADMAP
DATA SET

- Wikipedia english corpus (16 million words, Vocab Size: 70k)
- Training data for POS tagger: wikipedia hindi corpus (200 MB)
- Wikipedia Corpus for Bengali (100 MB)
DATA COLLECTION

- Cleaning english and hindi wikipedia corpus
- Collecting dataset for hindi
- Wiki Extractor for cleaning up the corpus
  github.com/bwbaugh/wikipedia-extractor
Figure: Position based character embeddings
Character Embedding captures the syntactic features
Can improve the result of tasks like POS tagging and NER
But how to join the char-level embedding with the word-level one ??
· **Options:**
  
  · Average addition to the word embeddings
  
  · Using CNN approach to get a char-level embedding for a word from the characters of that word
  
  · More on we can use syllables or affixes instead of character to get the joint embedding
ENHANCED WORD EMBEDDINGS

- Enhancing Word embedding to use intra-word information
- Word embedding from composition of character embeddings
  - Average Addition [2] character embedding vector without feature extraction
    \[ x_j = \frac{1}{2}(w_j + \frac{1}{N_j} \sum_{k=1}^{N_j} c_k) \]
  - Feature Extraction using CNN and adding information to word embeddings
- Using the joint learned embedding for the purpose like POS tagging
Table 3: Evaluation accuracies (%) on analogical reasoning.

<table>
<thead>
<tr>
<th>Method</th>
<th>Total</th>
<th>Capital</th>
<th>State</th>
<th>Family</th>
</tr>
</thead>
<tbody>
<tr>
<td>CBOB</td>
<td>54.85</td>
<td>51.40</td>
<td>66.29</td>
<td>62.92</td>
</tr>
<tr>
<td></td>
<td>58.24</td>
<td>53.32</td>
<td>66.29</td>
<td>70.00</td>
</tr>
<tr>
<td></td>
<td>60.07</td>
<td>54.36</td>
<td>66.29</td>
<td>73.75</td>
</tr>
<tr>
<td>Skip-Gram</td>
<td>69.14</td>
<td>62.78</td>
<td>82.29</td>
<td>80.83</td>
</tr>
<tr>
<td></td>
<td>68.04</td>
<td>63.66</td>
<td>81.14</td>
<td>78.75</td>
</tr>
<tr>
<td></td>
<td>72.07</td>
<td>65.44</td>
<td><strong>84.00</strong></td>
<td><strong>84.58</strong></td>
</tr>
<tr>
<td>GloVe</td>
<td>67.44</td>
<td>69.22</td>
<td>58.05</td>
<td>69.25</td>
</tr>
<tr>
<td></td>
<td>70.42</td>
<td>70.01</td>
<td>64.00</td>
<td>76.25</td>
</tr>
<tr>
<td></td>
<td><strong>72.99</strong></td>
<td><strong>73.26</strong></td>
<td>65.71</td>
<td>81.25</td>
</tr>
</tbody>
</table>
- Extracting character embeddings for the given corpus
- Feature extraction from character embeddings using CNN
· Previous work for POS tagging is mostly based on Statistical or Rule Based Model

· Can improve the results using the joint embedding

· Advantage: Less hand-crafted features

· primarily: mainly (11.726825), mostly (12.344781), principally (15.456143), chiefly (15.708947), largely (15.779496), and (16.920006), secondarily (17.022827)

Zhiyuan Liu Maosong Sun
Huanbo Luan Xinxiang Chen Lei Xu. “Joint Learning of Character and Word Embeddings”. In: (2015).
QUESTIONS?
CHAR-LEVEL EMBEDDING USING CNN - DETAILS

- Produces local features around each character of the word
- Combines them to get a fixed size character-level embedding
- Given a word $w$ composed of $M$ characters $c_1, c_2, ..., c_M$, each $c_M$ is transformed into a character embedding $r_{\text{chr}}^M$. The input to the convolution layer is the sequence of character embedding of $M$ characters.
· Window of size $k_{chr}$ (character context window) of successive windows in the sequence of $r_{chr}^1, r_{chr}^2, \ldots, r_{chr}^M$

· The vector $z_m$ (concatenation of character embedding $m$) for each character embedding is defined as follows:

$$z_m = (r_{chr}^{(m-(k_{chr}-1)/2)}, \ldots, r_{chr}^{(m+(k_{chr}-1)/2)})^T$$
Char-level embedding using CNN - Details

- Convolutional layer computes the jth element of the character embedding \( r_{wch} \) of the word \( w \) as follows:
  \[
  [r_{wch}]_j = \max_{1 < m < M}[W^0z_m + b^0]_j
  \]

- Matrix \( W^0 \) is used to extract local features around each character window of the given word.

- Global fixed-sized feature vector is obtained using max operator over each character window.
CHAR-LEVEL EMBEDDING USING CNN - DETAILS

- Parameter to be learned:
  - $W^{\text{chr}}, W^0, \text{and } b^0$

- Hyper-parameters:
  - $d^{\text{chr}}$: the size of the character vector
  - $c^{\text{lu}}$: the size of the convolution unit
    (also the size of the character-level embedding)
  - $k^{\text{chr}}$: the size of the character context window