Towards Better Character-based Word Vectors

Luoshu Wang
luoshu@stanford.edu, Stanford University

Overview

Motivation
- In recent years, a lot of work has been done to update word vectors by combining context information, e.g., BERT, ELMo. More exploration is needed to improve word representation using its internal structure.
- Current character-based word vectors, such as FastText method, implicitly assume that every morpheme is equally important, but in reality, that is not the case.

Key Contributions
- Design an unsupervised word segmentation model to split words into morphemes.
- Propose to represent words using these segmented subwords.
- Propose to adopt FastText Framework or simple logistic regression model as general model to train word embeddings.
- Conduct an extensive experimental study.

Data
- For build vocabulary:
  - Wiki latest pages articles (22G)
  - Enwik9 dataset (~700M)
  - Pre-trained FastText vectors (~1M words)
- Train Embedding:
  - enwik9 dataset
- Evaluation:
  - WordSim353 dataset
  - Rare Word dataset

Model and Embedding Creation Example

Build Dictionary: Extract (word, weight) pairs from articles or pre-trained vectors

Byte Pair Encoding

Optional

Get weight from Zipf's distribution if word rank info available.

FastText Framework with modifying subwords representation (skipgram + negative sampling)

Logistic regression model to fit pre-trained word vectors (for fast iteration)

Word Embeddings!

Get the most frequent subwords pairs, sorted by frequency

Be able to segment out-of-vocabulary words using most frequent subword pairs

Word Segmentation Examples

- deliveryman
- dissociatives
- childishness

Results

Human Similarity Judgment Task

<table>
<thead>
<tr>
<th>Model</th>
<th>Dictionary</th>
<th>WS353</th>
<th>Rare Word</th>
</tr>
</thead>
<tbody>
<tr>
<td>FastText</td>
<td>enwik9</td>
<td>74</td>
<td>45</td>
</tr>
<tr>
<td>Our model</td>
<td>enwik9</td>
<td>76</td>
<td>41</td>
</tr>
<tr>
<td>Our model</td>
<td>large wiki data</td>
<td>76</td>
<td>41</td>
</tr>
</tbody>
</table>

Analysis

- Our model can capture the morphemes within one word as expected.
- Our model outperforms FastText in WS353 dataset: there is more common words in WordSim353 dataset, where our model might be benefited from the morphological decomposition of words.
- The loss in the Rare Word dataset might come from BPE can only split word to non-duplicated subwords, so if the segmentation is imperfect, it might lose important information. (BPS might help with this issue.)

Important Features
- Word Weight
- Subword Frequency
- Begin-Of-Word / End-Of-Word

Conclusion
- Our model can capture the morphemes really well, and works better for words that can be decomposed to subwords.
- Our model training is around 2x faster than baseline enwik9 dataset, which is quite efficiently.

Future Work
- Learn word vectors in larger wikipedia dataset (22 G).
- Run more evals for Byte Pair Search method. BPS Example: Input: low 5, lower 2, newest 6, widest 3. Both (s, t) and (e, s) pair has subword pair frequency 9, but in the every iteration of BPE, BPE will only merge one pair. Merge order might influence subwords vocabulary and final word segmentation results. To address this issue, BPS will go through all the possibilities and returning more relevant subwords.

Additional Information

Mentor: Peng Qi (pengqi@cs.stanford.edu)

References