Learning Sentiment-Specific Word Embedding for Twitter Sentiment Classification

Duyu Tang\textsuperscript{1}, Furu Wei\textsuperscript{2}, Nan Yang\textsuperscript{2}, Ming Zhou\textsuperscript{2}, Ting Liu\textsuperscript{1}, Bing Qin\textsuperscript{1}

\textsuperscript{1} Harbin Institute of Technology
\textsuperscript{2} Microsoft Research
Roadmap

• Motivation

• The Proposed Method

• Experiments

• Conclusion
Twitter sentiment classification

• Input: A tweet
• Output: Sentiment polarity of the tweet
  – Positive / Negative / Neutral
Top-system in SemEval-2013 Task 2(B)

- *NRC-Canada* [Mohammad 2013]
  - Feature engineering
    - Hand-crafted features
    - Sentiment lexicons

- How about learning feature automatically from data for Twitter sentiment classification?
Word Representation (Embedding)

• Word embedding is important
  – Compositionality
  – [Yessenalina11; Socher13]

• Word Embedding

\[
\text{linguistic} = \begin{pmatrix}
1.045 \\
0.912 \\
-0.894 \\
-1.053 \\
0.459
\end{pmatrix}
\]
Is It Enough for Sentiment Analysis?

- Existing embedding learning algorithms typically use the syntactic contexts of words.

\[
\text{he formed the } \textbf{good} \text{ habit of } \ldots \\
\text{he formed the } \textbf{bad} \text{ habit of } \ldots
\]

The words with similar contexts but opposite sentiment polarity are mapped into close vectors.
Roadmap

• Motivation

• The Proposed Method

• Experiments

• Conclusion
Twitter Sentiment Classification

Input

Embedding Layer

Layer $f$

Layer $f'$

average

max

min

concatenate

Training Data

Learning Algorithm

Sentiment Classifier
Collobert and Weston (C&W) 2011

**Loss Function**

\[
\max(0, 1 - f^{cw}(t) + f^{cw}(t^r))
\]

**HardTanh**

\[
y = \begin{cases} 
-1 & \text{if } x < -1 \\
x & \text{if } -1 \leq x \leq 1 \\
1 & \text{if } x > 1 
\end{cases}
\]

**Linear**

\[
Y_j = W_{ij} \times X_i + b_j
\]
Collobert and Weston (C&W) 2011

Loss Function
\[ \max(0, 1 - f^{cw}(t) + f^{cw}(t^r)) \]

HardTanh
\[
\frac{\partial y}{\partial x} = \begin{cases} 
0 & \text{if } x < -1 \\
1 & \text{if } -1 \leq x \leq 1 \\
0 & \text{if } x > 1 
\end{cases}
\]

Linear
\[
\frac{\partial L}{\partial W_{ij}} = \frac{\partial L}{\partial Y_j} \times \frac{\partial Y_j}{\partial W_{ij}} = \frac{\partial L}{\partial Y_j} \times X_i \\
\frac{\partial L}{\partial X_i} = \sum_j \frac{\partial L}{\partial Y_j} \times \frac{\partial Y_j}{\partial X_i} = \sum_j \frac{\partial L}{\partial Y_j} \times W_{ij} \\
\frac{\partial L}{\partial b_j} = \frac{\partial L}{\partial Y_j} \times \frac{\partial Y_j}{\partial b_j} = \frac{\partial L}{\partial Y_j} \times 1
\]
Model 1: SSWE Hard

- Intuition
  - Use the *sentiment polarity of sentences (e.g. tweets)* to learn the sentiment-specific word embedding (*SSWE*)

- Solution
  - Predict the sentiment polarity of text

\[
\begin{array}{ll}
\text{Positive} & [1, 0] \\
\text{Negative} & [0, 1]
\end{array}
\]
it is so coool :)

Loss Function

\[ - \sum_{k = \{0, 1\}} f_k^g(t) \cdot \log(f_k^h(t)) \]

Gold Distribution

Predicted Distribution

Softmax

\[ Y_i = \frac{\exp(X_i)}{Z} \quad Z = \sum_{i'} \exp(X_{i'}) \]
Model 2: SSWE Soft

• Intuition
  – Use the sentiment polarity of sentences to learn the sentiment-specific word embedding

– Solution
  • Soften the hard constrains of Model 1

Positive  [1, 0]  P > N

Negative  [0, 1]  P < N
Loss Function

\[
\max (0, 1 - \delta_s(t) f_0^r(t) + \delta_s(t) f_1^r(t))
\]

Indicator Function

\[
\delta_s(t) = \begin{cases} 
1 & \text{if } f^g(t) = [1, 0] \\
-1 & \text{if } f^g(t) = [0, 1]
\end{cases}
\]
Model 3: SSWE Unified

• Intuition
  – Use both the syntactic contexts of words and the sentiment polarity of sentences to learn the sentiment-specific word embedding

  – Solution
    • A hybrid approach by capturing both information

  

  he formed the good habit of

  Context

  Sentiment
Loss Function
\[ \alpha \cdot \text{loss}_{cw}(t, t^r) + (1 - \alpha) \cdot \text{loss}_{us}(t, t^r) \]

Sentiment Loss
\[ \max(0, 1 - \delta_s(t) f_1^u(t) + \delta_s(t) f_1^u(t^r)) \]
Embedding Training

• Data
  – Tweets contains positive/negative emoticons
    | Positive | :) | :) | :- | :D | =) |
    |Negative  | :( | :( | :( |    |    |
  – 5M positive, 5M negative tweets from April, 2013

• Back-propagation + AdaGrad [Duchi 2011]
  – Embedding length = 50
  – Window size = 3
  – Learning rate = 0.1
Roadmap

• Motivation
• The Proposed Method
• Experiments
  – Twitter Sentiment Classification
  – Word Similarity of Sentiment Lexicons

• Conclusion
Twitter Sentiment Classification

• Setting
  – Data
    • Twitter Sentiment Classification Track in Semantic Evaluation 2013 (message-level)
    • Positive VS negative classification

  – Evaluation metric
    • Macro-F1 of positive VS negative
Results

• Comparison with Different Embeddings
Results

• Comparison with Twitter Sentiment Classification Algorithms

<table>
<thead>
<tr>
<th>Method</th>
<th>Macro-F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>DistSuper + unigram</td>
<td>61.74</td>
</tr>
<tr>
<td>DistSuper + uni/bi/tri-gram</td>
<td>63.84</td>
</tr>
<tr>
<td>SVM + unigram</td>
<td>74.50</td>
</tr>
<tr>
<td>SVM + uni/bi/tri-gram</td>
<td>75.06</td>
</tr>
<tr>
<td>NBSVM</td>
<td>75.28</td>
</tr>
<tr>
<td>RAE</td>
<td>75.12</td>
</tr>
<tr>
<td><strong>NRC (Top System in SemEval)</strong></td>
<td><strong>84.73</strong></td>
</tr>
<tr>
<td>NRC - ngram</td>
<td>84.17</td>
</tr>
<tr>
<td><strong>SSWE_u</strong></td>
<td><strong>84.98</strong></td>
</tr>
<tr>
<td><strong>SSWE_u+NRC</strong></td>
<td><strong>86.58</strong></td>
</tr>
<tr>
<td><strong>SSWE_u+NRC-ngram</strong></td>
<td><strong>86.48</strong></td>
</tr>
</tbody>
</table>
SemEval 2014 Task 9 (b)

- **Coooollll**: A deep learning system for Twitter sentiment classification

![Diagram showing the process of training data, feature representation, learning algorithm, and sentiment classifier.]
SemEval 2014 Task 9 (b)

• Results
  – Our system **Coooollll** is ranked 2\textsuperscript{nd} among 45 systems on Twitter2014 test set.
SemEval 2014 Task 9 (b)

• Results

  – Our system **Coooolll** is ranked 2\textsuperscript{nd} among 45 systems on Twitter2014 test set.

<table>
<thead>
<tr>
<th>Method</th>
<th>Positive/Negative/Neutral</th>
<th>Positive/Negative</th>
</tr>
</thead>
<tbody>
<tr>
<td>SSWE</td>
<td>T1 70.49  T2 64.29  T3 68.69  T4 66.86  T5 50.00</td>
<td>T1 84.51  T2 85.19  T3 85.06  T4 86.14  T5 62.02</td>
</tr>
<tr>
<td>Coooolll</td>
<td>T1 72.90  T2 67.68  T3 70.40  T4 70.14  T5 46.66</td>
<td>T1 86.46  T2 85.32  T3 86.01  T4 87.61  T5 56.55</td>
</tr>
<tr>
<td>STATE</td>
<td>T1 71.48  T2 65.43  T3 66.18  T4 67.07  T5 44.89</td>
<td>T1 83.96  T2 82.82  T3 84.39  T4 86.16  T5 58.27</td>
</tr>
<tr>
<td>W2V</td>
<td>T1 55.19  T2 52.98  T3 52.33  T4 50.58  T5 49.63</td>
<td>T1 68.87  T2 71.89  T3 74.50  T4 71.52  T5 61.60</td>
</tr>
<tr>
<td>Top</td>
<td>T1 74.84  T2 70.28  T3 72.12  T4 70.96  T5 58.16</td>
<td>- - - - - - - - - -</td>
</tr>
<tr>
<td>Average</td>
<td>T1 63.52  T2 55.63  T3 59.78  T4 60.41  T5 45.44</td>
<td>- - - - - - - - - -</td>
</tr>
</tbody>
</table>
Roadmap

• Motivation
• The Proposed Method
• Experiments
  – Twitter Sentiment Classification
  – Word Similarity of Sentiment Lexicons
• Conclusion
Experiment Settings

• Setting
  – Evaluation metric

\[ \text{Accuracy} = \frac{\sum_{i=1}^{\#Lex} \sum_{j=1}^{N} \beta(w_i, c_{ij})}{\#Lex \times N} \]

– Data

<table>
<thead>
<tr>
<th>Lexicon</th>
<th>Positive</th>
<th>Negative</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>HL</td>
<td>1,331</td>
<td>2,647</td>
<td>3,978</td>
</tr>
<tr>
<td>MPQA</td>
<td>1,932</td>
<td>2,817</td>
<td>4,749</td>
</tr>
<tr>
<td>Joint</td>
<td>1,051</td>
<td>2,024</td>
<td>3,075</td>
</tr>
</tbody>
</table>

Accuracy = 8/10 = 80%

{cool, awesome, great, bad, nice, interesting, fantastic, fantastic, love, excellent, terrible}
Results

• Evaluation
  – Word similarity of sentiment lexicons

<table>
<thead>
<tr>
<th>Embedding</th>
<th>HL</th>
<th>MPQA</th>
<th>Joint</th>
</tr>
</thead>
<tbody>
<tr>
<td>Random</td>
<td>50.00</td>
<td>50.00</td>
<td>50.00</td>
</tr>
<tr>
<td>C&amp;W</td>
<td>63.10</td>
<td>58.13</td>
<td>62.58</td>
</tr>
<tr>
<td>Word2vec</td>
<td>66.22</td>
<td>60.72</td>
<td>65.59</td>
</tr>
<tr>
<td>ReEmb(C&amp;W)</td>
<td>64.81</td>
<td>59.76</td>
<td>64.09</td>
</tr>
<tr>
<td>ReEmb(w2v)</td>
<td>67.16</td>
<td>61.81</td>
<td>66.39</td>
</tr>
<tr>
<td>WVSA</td>
<td>68.14</td>
<td>64.07</td>
<td>67.12</td>
</tr>
<tr>
<td>SSWE_{\text{h}}</td>
<td>74.17</td>
<td>68.36</td>
<td>74.03</td>
</tr>
<tr>
<td>SSWE_{\text{r}}</td>
<td>73.65</td>
<td>68.02</td>
<td>73.14</td>
</tr>
<tr>
<td>SSWE_{\text{u}}</td>
<td>77.30</td>
<td>71.74</td>
<td>77.33</td>
</tr>
</tbody>
</table>
Conclusion

• Learn continuous representation of words for Twitter sentiment classification.

• Develop three neural networks for learning sentiment-specific word embedding (SSWE) from massive tweets without manual annotation.

• The effectiveness of SSWE has been verified in Twitter sentiment classification and word similarity judgement.
Thanks