at SemEval-2019 Task 9: Semi-supervised Domain Adaptation using Tri-training for Suggestion Mining
Sai Prasanna, Sri Ananda Seelan
saiprasanna.r, anandaseelan.in@zohocorp.com

Suggestion Mining

• Mining sentences that contain suggestions in online discussions and reviews.

• Example Suggestion: “An electric kettle would have been a good addition to the room.”

• Subtask A: Domain specific sentence classification with training data from Microsoft Windows developer platform.

• Subtask B: Cross-domain classification on hotel reviews dataset.

Objective

• Evaluate recent advancements from semi-supervised and transfer learning literature to come up with a system for suggestion mining.

• Subtask A
  • Relatively small dataset
  • Class Imbalance

  • Transfer Learning: Use pre-trained language models and transfer it to downstream tasks.

• Subtask B
  • No hand labelled training data.

  • Domain transfer using Semi-Supervised Learning: Bootstrapping a model to come up with labels for data in a new domain and use it for training.

Tri-Training

Following the work of Ruder and Plank (2018) to apply classic tri-training, a semi-supervised learning technique for domain adaptation.

Algorithm 1 Tri-training
1: \( L \leftarrow \) Labelled Data, \( |L| = m \)
2: \( U \leftarrow \) Unlabelled Data, \( |U| = n \)
3: for \( i \leftarrow 1, 2, 3 \) do
4: \( l_i \leftarrow \) BootstrapSamples(\( L \))
5: end for
6: repeat
7: for \( i \leftarrow 1, 2, 3 \) do
8: \( M_i \leftarrow \) Train(\( l_i \))
9: end for
10: for \( i \leftarrow 1, 2, 3, \) do
11: \( l_i \leftarrow \) \( L \)
12: for \( j \leftarrow 1, n \) do
13: if \( M_p(\{U_j\}) \) where \( p, q \neq i \) then
14: \( l_i \leftarrow l_i + \{U_j, M_p(\{U_j\})\} \)
15: end if
16: end for
17: end for
18: until no improvement in validation metrics

Model Architecture

Baseline: GloVe (Pennington et al., 2014) + Deep Averaging Net (Iyyer et al., 2015)
Final: BERT (Devlin et al., 2018) + CNN (Kim et al., 2014)

Results

<table>
<thead>
<tr>
<th>Models/Experiments</th>
<th>Subtask - A</th>
<th>Subtask - B</th>
</tr>
</thead>
<tbody>
<tr>
<td>Organizer Baseline</td>
<td>26.80</td>
<td>73.21</td>
</tr>
<tr>
<td>DAN + GloVe</td>
<td>38.84 ± 3.10</td>
<td>56.35 ± 4.71</td>
</tr>
<tr>
<td>DAN + BERT</td>
<td>60.82 ± 3.99</td>
<td>70.49 ± 4.09</td>
</tr>
<tr>
<td>CNN + BERT</td>
<td>64.81 ± 4.86</td>
<td>64.31 ± 6.72</td>
</tr>
<tr>
<td>CNN + BERT w/o Upsampling</td>
<td>70.58 ± 4.24</td>
<td>58.66 ± 7.79</td>
</tr>
<tr>
<td>CNN + BERT + Tritrain (Test set)</td>
<td>66.81 ± 1.90</td>
<td>82.19 ± 1.03</td>
</tr>
<tr>
<td>CNN + BERT + Tritrain (Yelp)</td>
<td>NA</td>
<td>81.98 ± 2.05</td>
</tr>
</tbody>
</table>

Table 1: F1-scores of different models/experiments
Confidence interval over 5 seeds.
DAN – Deep Averaging Network

<table>
<thead>
<tr>
<th>Subtask</th>
<th>Model A</th>
<th>Model B</th>
<th>p-value</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>DAN + glove (Baseline)</td>
<td>DAN + BERT</td>
<td>( \approx 0 )</td>
</tr>
<tr>
<td>A</td>
<td>DAN + BERT</td>
<td>CNN + BERT</td>
<td>0.046</td>
</tr>
<tr>
<td>B</td>
<td>CNN + BERT</td>
<td>CNN + BERT + Tritrain (Test set)</td>
<td>3.25e-08</td>
</tr>
</tbody>
</table>

Table 2: Pairwise comparison of various models using the McNemar’s Test
\( p \leq 0.05 \) indicates a significant difference between the model performance.

References


Built with Pytorch & AllenNLP