Appendix I: Experimental Setup

6.1 Data statistics

We consider a wide range of text-representation-based tasks in this paper, including document categorization, text sequence matching and (short) sentence classification. For document classification tasks, we use the same data splits in (Zhang et al., 2015b) (downloaded from https://goo.gl/QaRpr7); for short sentence classification, we employ the same training/testing data and preprocessing procedure with (Kim, 2014). The statistics and corresponding types of these datasets are summarized in Table 9

<table>
<thead>
<tr>
<th>Datasets</th>
<th>#w</th>
<th>#c</th>
<th>Train</th>
<th>Types</th>
</tr>
</thead>
<tbody>
<tr>
<td>Yahoo</td>
<td>104</td>
<td>10</td>
<td>1,400K</td>
<td>Topic categorization</td>
</tr>
<tr>
<td>AG News</td>
<td>43</td>
<td>4</td>
<td>120K</td>
<td>Topic categorization</td>
</tr>
<tr>
<td>Yelp P.</td>
<td>138</td>
<td>2</td>
<td>560K</td>
<td>Sentiment analysis</td>
</tr>
<tr>
<td>Yelp F.</td>
<td>152</td>
<td>5</td>
<td>650K</td>
<td>Sentiment analysis</td>
</tr>
<tr>
<td>DBpedia</td>
<td>57</td>
<td>14</td>
<td>560K</td>
<td>Ontology classification</td>
</tr>
<tr>
<td>SNLI</td>
<td>11</td>
<td>6</td>
<td>549K</td>
<td>Textual Entailment</td>
</tr>
<tr>
<td>MultiNLI</td>
<td>21</td>
<td>13</td>
<td>393K</td>
<td>Textual Entailment</td>
</tr>
<tr>
<td>WikiQA</td>
<td>7</td>
<td>26</td>
<td>20K</td>
<td>Question answering</td>
</tr>
<tr>
<td>Quora</td>
<td>13</td>
<td>13</td>
<td>384K</td>
<td>Paraphrase identification</td>
</tr>
<tr>
<td>MSRP</td>
<td>23</td>
<td>23</td>
<td>4K</td>
<td>Paraphrase identification</td>
</tr>
<tr>
<td>MR</td>
<td>20</td>
<td>2</td>
<td>11K</td>
<td>Sentiment analysis</td>
</tr>
<tr>
<td>SST-1</td>
<td>18</td>
<td>5</td>
<td>12K</td>
<td>Sentiment analysis</td>
</tr>
<tr>
<td>SST-2</td>
<td>19</td>
<td>2</td>
<td>10K</td>
<td>Sentiment analysis</td>
</tr>
<tr>
<td>Subj</td>
<td>23</td>
<td>2</td>
<td>10K</td>
<td>Subjectivity classification</td>
</tr>
<tr>
<td>TREC</td>
<td>10</td>
<td>6</td>
<td>6K</td>
<td>Question classification</td>
</tr>
</tbody>
</table>

Table 9: Data Statistics. Where #w, #c and Train denote the average number of words, the number of classes and the size of training set, respectively. For sentence matching datasets, #w stands for the average length for the two corresponding sentences.

6.2 Sequence Tagging Results

<table>
<thead>
<tr>
<th>Datasets</th>
<th>CoNLL2000</th>
<th>CoNLL2003</th>
</tr>
</thead>
<tbody>
<tr>
<td>CNN-CRF</td>
<td>94.32</td>
<td>89.59</td>
</tr>
<tr>
<td>BI-LSTM-CRF</td>
<td>94.46</td>
<td>90.10</td>
</tr>
<tr>
<td>SWEM-CRF</td>
<td>90.34</td>
<td>86.28</td>
</tr>
</tbody>
</table>

Table 10: The results (F1 score) on sequence tagging tasks.

SWEM-CRF indicates that CRF is directly operated on top of the word embedding layer and make predictions for each word (there is no contextual/word-order information before CRF layer, compared to CNN-CRF or BI-LSTM-CRF). As shown above, CNN-CRF and BI-LSTM-CRF consistently outperform SWEM-CRF on both sequence tagging tasks, although the training takes around 4 to 5 times longer (for BI-LSTM-CRF) than SWEM-CRF. This suggests that for chunking and NER, compositional functions such as LSTM or CNN are very necessary, because of the sequential (order-sensitive) nature of sequence tagging tasks.

6.3 What are the key words used for predictions?

Given the sparsity of word embeddings, one natural question would be: What are those key words that are leveraged by the model to make predictions? To this end, after training SWEM-max on Yahoo! Answer dataset, we selected the top-10 words (with the maximum values in that dimension) for every word embedding dimension. The results are visualized in Figure 3. These words are indeed very predictive since they are likely to occur in documents with a specific topic, as discussed above. Another interesting observation is that the frequencies of these words are actually quite low in the training set (e.g. colston: 320, repubs: 255 win32: 276), considering the large size of the training set (1,400K). This suggests that the model is utilizing those relatively rare, yet representative words of each topic for the final predictions.

6.4 How many word embedding dimensions are needed?

Since there are no compositional parameters in SWEM, the component that contains the semantic
Figure 4: The test accuracy comparisons between SWEM and CNN/LSTM models on (a) Yahoo! Answers dataset and (b) SNLI dataset, with different proportions of training data (ranging from 0.1% to 100%).

Table 11: Test accuracy of SWEM on Yahoo dataset with a wide range of word embedding dimensions.

<table>
<thead>
<tr>
<th># Dim.</th>
<th>3</th>
<th>10</th>
<th>30</th>
<th>100</th>
<th>300</th>
<th>1000</th>
</tr>
</thead>
<tbody>
<tr>
<td>Yahoo</td>
<td>64.05</td>
<td>72.62</td>
<td>73.13</td>
<td>73.12</td>
<td>73.24</td>
<td>73.31</td>
</tr>
</tbody>
</table>

Surprisingly, SWEM consistently outperforms CNN and LSTM models by a large margin, on a wide range of training data proportions. For instance, with 0.1% of the training samples from Yahoo dataset (around 1.4K labeled data), SWEM achieves an accuracy of 56.10%, which is much better than that of models with CNN (25.32%) or LSTM (42.37%). On the SNLI dataset, we also noticed the same trend that the SWEM architecture result in much better accuracies, with a fraction of training data. This observation indicates that overfitting issues in CNN or LSTM-based models on text data mainly stems from over-complicated compositional functions, rather than the word embedding layer. More importantly, SWEM tends to be a far more robust model when only limited data are available for training.

6.5 Sensitivity of compositional functions to sample size

To explore the robustness of different compositional functions, we consider another application scenario, where we only have a limited number of training data, e.g., when labeled data are expensive to obtain. To investigate this, we re-run the experiments on Yahoo and SNLI datasets, while employing increasing proportions of the original training set. Specifically, we use 0.1%, 0.2%, 0.6%, 1.0%, 10%, 100% for comparison; the corresponding results are shown in Figure 4.