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

Datasets	#w	#c	Train	Types
Yahoo	104	10	1,400K	Topic categorization
AG News	43	4	120K	Topic categorization
Yelp P.	138	2	560K	Sentiment analysis
Yelp F.	152	5	650K	Sentiment analysis
DBpedia	57	14	560K	Ontology classification
SNLI	11 / 6	3	549K	Textual Entailment
MultiNLI	21/11	3	393K	Textual Entailment
WikiQA	7 / 26	2	20K	Question answering
Quora	13 / 13	2	384K	Paraphrase identification
MSRP	23 / 23	2	4K	Paraphrase identification
MR	20	2	11K	Sentiment analysis
SST-1	18	5	12K	Sentiment analysis
SST-2	19	2	10K	Sentiment analysis
Subj	23	2	10K	Subjectivity classification
TREC	10	6	6K	Question classification

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

Datasets	CoNLL2000	CoNLL2003
CNN-CRF	94.32	89.59
BI-LSTM-CRF	94.46	90.10
SWEM-CRF	90.34	86.28

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 How many word embedding dimensions are needed?

Since there are no compositional parameters in SWEM, the component that contains the semantic information of a text sequence is the word embedding. Thus, it is of interest to see how many word embedding dimensions are needed for a SWEM architecture to perform well. To this end, we vary the dimension from 3 to 1000 and train a SWEM-*concat* model on the Yahoo dataset. For fair comparison, the word embeddings are randomly initialized in this experiment, since there are no pre-trained word vectors, such as GloVe (Pennington et al., 2014), for some dimensions we consider. As shown in Table 11, the model exhibits higher accuracy with larger word embedding dimensions. This is not surprising since with more embedding dimensions, more semantic features could be potentially encapsulated. However, we also observe that even with only 10 dimensions, SWEM demonstrates comparable results relative to the case with 1000 dimensions, suggesting that word embeddings are very efficient at abstracting semantic information into fixed-length vectors. This property indicates that we may further reduce the number of model parameters with lower-dimensional word embeddings, while still achieving competitive results.

6.4 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 3.

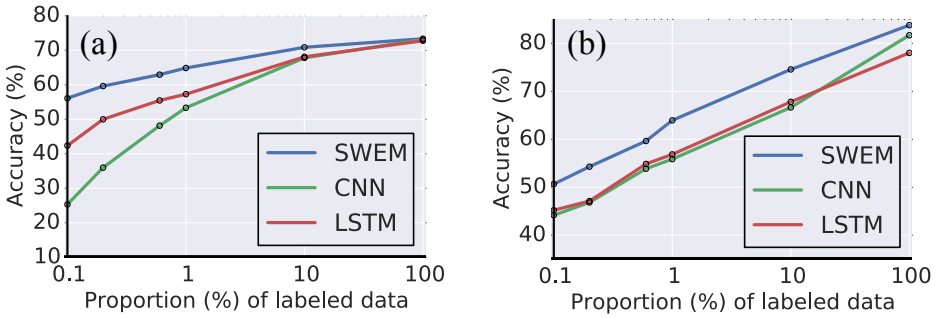

Figure 3: The test accuracy comparisons between SWEM and CNN/LSTM models on (a) Yahoo! Answers dataset and (b) SNLI dataset, with different proportions of training data (ranging from 0.1% to 100%).

# Dim.	3	10	30	100	300	1000
Yahoo	64.05	**72.62**	73.13	73.12	73.24	73.31

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

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.