Natural Language Processing 1, Practical II: Encoding Sentences with Neural Models

Tim van Loenhout
Student no: 10741577

Ard Snijders
Student no: 12854913

1 Introduction

Many of the recent developments within natural language processing (NLP) build on the concept of distributed representations of words, or word embeddings. Research has shown that word embeddings can serve as reliable representations for tasks that require single-word encodings (Mikolov et al., 2013), (Pennington et al., 2014). What if we want to encode phrases, or sentences? This task is non-trivial, as we cannot simply extend distributional semantics to account for the meaning of phrases and sentences - given the infinite possibilities of constructing sentences, even with a limited vocabulary it is not feasible to learn distributed representations for all possible sentences. Rather, it would be more appropriate to learn sentence representations via a more robust method, allowing for improved generalization on unseen phrases.

The task of obtaining such sentence representations remains an open question in NLP research. Early work by Mitchel & Lapata (2010) suggests a vector mixture model framework, where word combinations can be modelled as linear transformations (additive/multiplicative) between vectors, whereas more recently, researchers have begun employing neural architectures such as recurrent neural networks (RNNs), long short-term memory (LSTMs) (Palangi et al., 2016) and Tree-LSTMs (Tai et al., 2015) (Phong Le, 2015) (Xiaodan Zhu, 2015).

In this work, we aim to provide an experimental overview of prevalent methods to obtain sentence representations. We consider a number of neural compositional semantic models: supervised machine learning models which are typically trained for applications in specific downstream NLP tasks. For this assessment, we will evaluate models on the task of sentiment classification. We start with a simplistic bag-of-words approach, and gradually introduce more complex models with the aim of working towards a sentence representation which captures most of the aspects of sentence meaning. Specifically, we assess and compare the performance of a several neural bag-of-words (BOW) models and LSTM networks. For the latter, we investigate the Binary Tree-LSTM network as formulated by Tai, Socher & Manning (2015) and evaluate its performance against the conventional LSTM architecture. Furthermore, we implement a Tree-LSTM approach with sentiment supervision at both the node and sentence level. Lastly, we explore how varying sentences lengths affects performance across models.

Contrary to our expectations, we find that the comparatively complex LSTM-based networks do not offer significantly higher classification performance compared to deep continuous bag-of-words models with pre-trained word embeddings. Furthermore, we do not observe significant differences in model performance for varying sentence lengths. Finally, we do not find evidence that supervising sentiment at node and sentence level improves model performance, although this approach appears less prone to overfitting.

2 Background

All of the models covered in this work build on the concept of word embeddings, where words are represented by real-valued vectors which are distributed in a continuous space. The neural bag-of-words model (BOW) returns a sentence representation through summation of individual word representations. As this is a commutative approach, BOW-based models do not take word order into account and fail to capture syntactic information. We can also compute a sentence representation using a recurrent neural network (RNN), where a sentence is fed through a network as a linear sequence of word representations, and where a hidden state is used to capture the sequential relationships in the data. This approach preserves word order but suffers from a phenomenon known as the vanishing gradient: the error signal recursively diminishes as we backpropagate through time, which can cause the early layers of a network to stop learning. This is problematic in language modelling, as it prevents the RNN from adequately capturing important long-distance dependencies.

Long short-term memory networks (Hochreiter & Schmidhuber, 1997) partially address this problem by incorporating a series of novel concepts called gates, which regulate the flow of information through the network, deciding what information should be kept, and what should be forgotten. This allows the network to more accurately capture relations between words that appear further apart. However, a problem with applying conventional LSTM network architectures to language modelling is that sentences are encoded as strictly linear sequences of tokens. While this method accounts for word order, we cannot capture the true syntactic structure of a sentence. This problem can be addressed by using a Tree-LSTM (Tai et al., 2015) (Phong Le, 2015) (Xiaodan Zhu, 2015), a tree-based network architecture which encodes sentences not as a linear sequence, but according to the structure of their syntactic parse trees. Previous research suggests that binary Tree-LSTMs outperform conventional LSTM networks on the task of sentiment analysis (Tai et al., 2015).

3 Models

For our first model, we use a simple BOW-approach where each word in the vocabulary is mapped to a $5-$dimensional vector indicating its sentiment class. A sentence representation is obtained by a summation over the vectors of its words, after wich a softmax operation is performed to convert the real-valued sentence vector to a vector with probabilities. Subsequently, an argmax function is used to obtain the sentiment prediction. For all models, we use cross-entropy loss to measure the model error. Next, we consider a continuous BOW (CBOW) model. Though similar to the simple BOW, word embeddings can now take an arbitrary size. We initialize word embeddings with a size of 300. We then learn a parameter matrix W by incorporating a linear layer, such that we can project the summed 300-dimensional sentence vector (along with a bias term b) onto a 5-dimensional sentiment space. In an attempt to make the CBOW model more powerful, we can extend it to a Deep CBOW (DCBOW) by adding more layers and several non-linear activations. Specifically, we linearly transform our 300-dimensional word embeddings to a hidden layer of size 100 and perform a tanh activation. Next, another linear transformation and tanh activation are applied, after which the layer of size 100 is projected to 5-dimensional space.

In the (deep) CBOW models, our word-embeddings were randomly initialized, and the network's ability to learn largely relied on the sentiment prediction error signal. For the final BOW model, we enrich the DCBOW by using pre-trained embeddings (PTDCBOW). These high-quality word embeddings more closely approximate how words relate to one another, allowing the model to start with representations that already encapsulate some useful information.

For our basic LSTM model, we follow the conventional implementation as formulated by Hochreiter & Schmidhuber. Furthermore, we experiment whether mini-batching (where multiple sentences are fed through the network simultaneously) affects the model performance. For the binary Tree-LSTM model, we adhere to the implementation given by Tai et al.. For this model, we first explore an architecture where sentences are encoded according to their syntactic structure. We also assess the effect of supervising sentiment at a node-level.

4 Experiments

We evaluate the effectiveness of the different models by performing sentiment analysis classification on a dataset of movie reviews, for which we use the Stanford Sentiment Treebank. This dataset comprises a collection of $11,855$ sentences with annotated sentiment classes ranging between 0 (very negative) and 4 (very positive). The dataset also includes binary parse trees per sentence, with sentiment scores at each node, which we will use when applying the Tree LSTM models. Prior to training, all examples are shuffled randomly after which we split our data into a $8544/1101/2210$ train/dev/test split. All bag-of-words models are trained for 50000 iterations, with a learning rate of $5e^{-3}$. We use GloVe embeddings for the pre-trained BOW model and all subsequent LSTM models (Pennington et al., 2014). Of the 21701 unique words in our dataset, 978 words do not appear in the GloVe embeddings. We use a randomly initialized $< UNK >$ vector to account for this. We train the LSTM models for 10000 iterations (preliminary experiments indicated that this is sufficient for model convergence) with a learning rate of $2e^{-4}$.

For the conventional LSTM networks, we experiment with single-sentence batches and mini-batches of 16 sentences. Padding vectors (all entries equal to 1) are used to compensate for varying sentence lengths. We employ dropout regularization with $p = 0.5$ for all LSTM networks. This method entails randomly 'dropping' nodes from the network to prevent the model from becoming too dependent on certain co-dependencies, forcing it to employ more of its parameters, which can induce improved model generalization. All models are optimized using Adam, an advanced stochastic gradient descent algorithm. Subsequently, model performance is evaluated by the average test set accuracy over three different seeds. Comparisons between the models are tested for significance using a sign-test over all the predictions on the test set.

We first compare all BOW-based and LSTM models separately. We then compare the best-performing submodels of both classes. We investigate the effect of sentence length by tracking performance across models for fixed sentence lengths, within a range of 2 to 50 tokens. Furthermore, we assess differences in accuracy between conventional LSTM networks and tree-based networks. Lastly, we explore the effect of supervising sentiment in tree-based LSTM networks at the node-level. We do so by extracting the children nodes and subtrees that form the parse tree associated with each sentence, and then appending these words and phrases to our dataset, along with their annotated sentiment scores. Consequently, the model is trained both on entire sentences and their constituent parts.

5 Results

For the final test accuracies, see figure 1. An overview of significance tests can be found in figure 5 in the appendix. Enriching the simple bag-of-words with larger word embeddings (CBOW) yields a significant improvement in performance. Possibly, using larger embeddings allows the model to learn more fine-grained aspects of each word. The Deep CBOW model (DCBOW) appears to converge slightly faster compared to the CBOW model, which could be caused by its ability to learn non-linear relations between inputs and outputs by use of non-linear activations. However, we do not find evidence that the Deep CBOW model (DCBOW) offers superior accuracy - possibly, the DCBOW is still bottle-necked by the commutative BOW sentence representation. Moreover, the model's use of randomly initialized vectors, and the small size of the dataset, might hamper its ability to learn good word embeddings, negatively affecting its performance. Indeed, when we initialize the same model with pre-trained GloVe embeddings (PTDCBOW), we report a significant improvement in accuracy, which underlines the importance of high-quality word embeddings.

For the LSTM models, we first assess performance of a conventional LSTM architecture. With single-sentence batches, it does not perform significantly better than the PTDCBOW. Training the model with mini-batch gradient descent (MLSTM) appears to speed up convergence and reduce variability in validation accuracy. Possibly, averaging the gradient over multiple batches causes the weight updates to be more stable, which causes steadier convergence.

Contrary to the evidence presented by Tai et al. (2015), we do not find that encoding sentences with a Tree LSTM (TLSTM) leads to significant improvements in accuracy compared to the conventional LSTM. Similary, Tree-LSTMs with supervision of sentiment scores at node and sentence-level (STLSTM) do not yield significant improvements in sentiment classification accuracy. However, the STLSTM seem less susceptible to overfitting compared to the other LSTMs, which could be caused by the increased size of the dataset. As can be seen from the validation accuracy curves in figure 3, the training loss for both the MLSTM and TLSTM continues to fall as the number of iterations grows, but this higher training accuracy does not translate to better performance on the validation set. Conversely, the training loss for the STLSTM remains largely stationary as the number of iterations increases, which is consistent with its non-decreasing validation accuracy, even after many iterations.

Finally, we assess the performance of all models for varying sentence lengths - see figure 4 for results. The performances between models appears to largely follow the model accuracies discussed in figure 1, with the PTDCBOW and LSTM models offering the best overall performance. However, it appears that none of these models clearly outperform one another for varying sentence length. Although it might appear that certain models become more uncertain for longer sentences (> 40 tokens), it should be noted that the majority of sentences have lengths within the range of approximately 6 and 29 tokens - because of this, it would be difficult to draw any meaningful inferences for model performance sentence lengths outside this range.

5.1 Discussion

While the TLSTM model offers the best performance, it does not offer significant improvements compared to the other models. Moreover, it appears that for the task of sentiment classification, the comparatively simple PTDCBOW model performs almost as well as all of the LSTM models. Theoretically, we would expect the LSTM models to outperform BOW-based models as LSTMs encode more aspects of a sentence (word order for conventional LSTMs, and syntax for TLSTMs), thus rendering them more appropriate for representing a sentence. However, our results suggest that the models produced comparable sentence representations.

One possible explanation for this is that in the context of sentiment analysis, word order and syntax do not necessarily account for much semantically-relevant information. However, this seems unlikely, given the evidence cited earlier. Rather, it is more probable that for this particular dataset, word order and syntax are not important determinants for sentiment. Similarly, we expected the various LSTM models to perform better over longer sentences opposed to the BOW models: by design, LSTMs preserve state over sequences of data (Tai et al., 2015), which in principle should render them more capable of capturing important long-distance dependencies.

6 Conclusion

Taking the underlying theories of the assessed models into consideration, we would expect to see starker differences in performance as we introduce models that encode more aspects of sentences. In order to draw stronger conclusions on the role of word order for different models, further work could focus on explicitly testing for word order - for instance, by shuffling words within sentences and comparing model performance with unaltered sentences. Moreover, it could be investigated whether LSTMs can compose superior representations for longer sentences, though a dataset with more longer sentences would be required for this. It should also be considered that this work only focused on model performance on sentiment analysis: it is possible that certain models might outperform one another on other tasks. In that respect, it could be appropriate to compare the same set of models across a number of different tasks to ensure a fairer comparison.

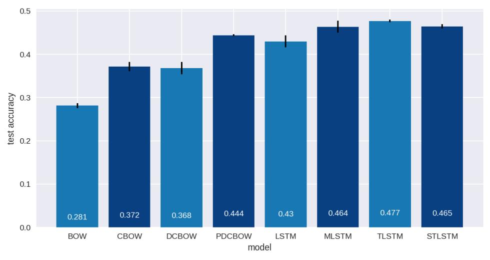

Figure 1: Average test accuracy over three seeds. The black bars indicate standard deviations.

References

Hochreiter, S. & Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, *9*, 1735–1780.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., & Dean, J. (2013). Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems*, 3111–3119.

Mitchell, J. & Lapata, M. (2010). Composition in distributional models of semantics. *Cognitive Science: A Multidisciplinary Journal*, *34*, 1551–6709.

Palangi, H., Deng, L., Shen, Y., Gao, J., He, X., Chen, J., Song, X., & Ward, R. (2016). Deep sentence embedding using long short-term memory networks: Analysis and application to information retrieval. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*.

Pennington, J., Socher, R., & Manning, C. D. (2014). Glove: Global vectors for word representation. *EMNLP*, *14*, 1532–1542.

Phong Le, W. Z. (2015). Compositional distributional semantics with long short term memory. *Proceedings of the Fourth Joint Conference on Lexical and Computational Semantics*, 10–19.

Tai, K. S., Socher*, R., & Manning, C. D. (2015). Improved semantic representations from tree-structured long short-term memory networks. *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing*.

Xiaodan Zhu, Parinaz Sobhani, H. G. (2015). Long short-term memory over recursive structures. *Proceedings of the 32nd International Conference on Machine Learning*.

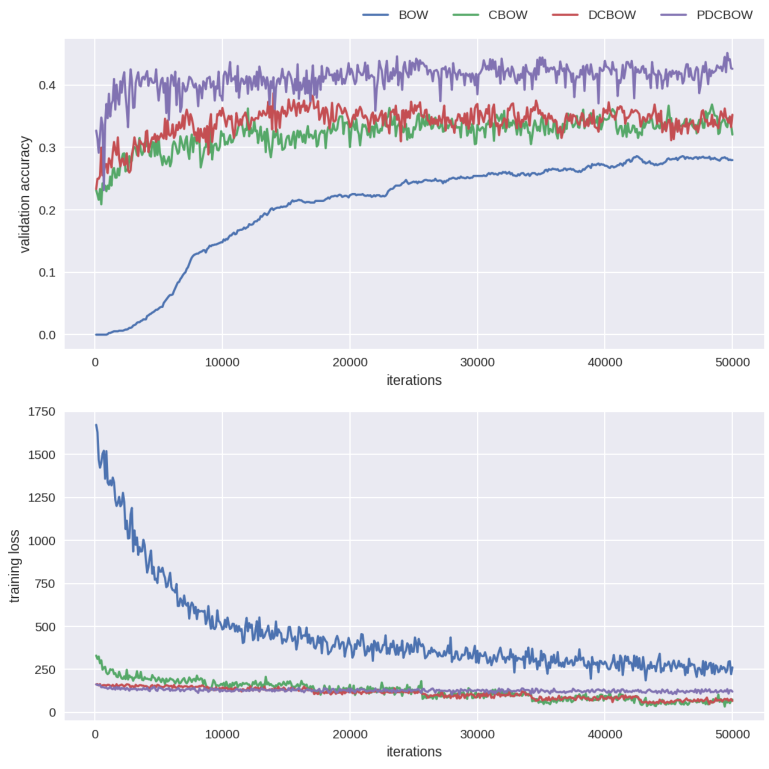

Figure 2: Loss and accuracy curves of the BOW models.

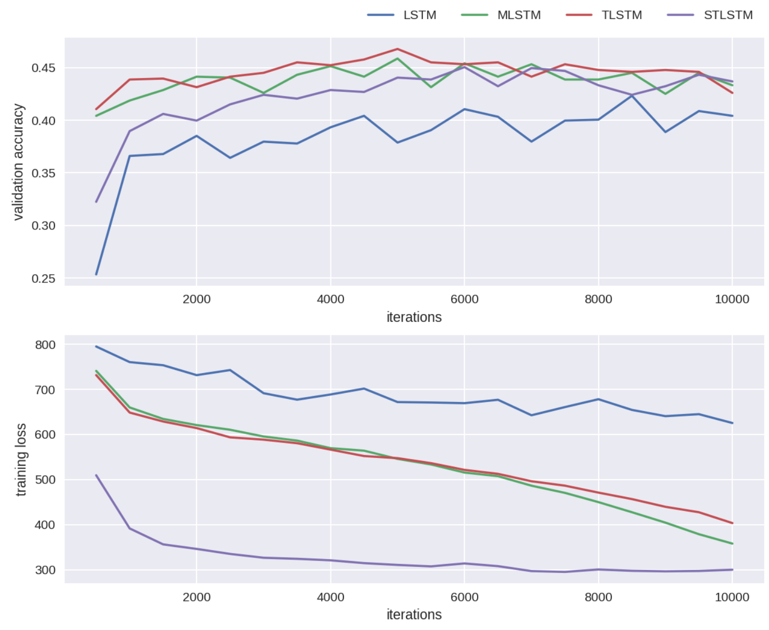

Figure 3: Loss and accuracy curves of the LSTM models.

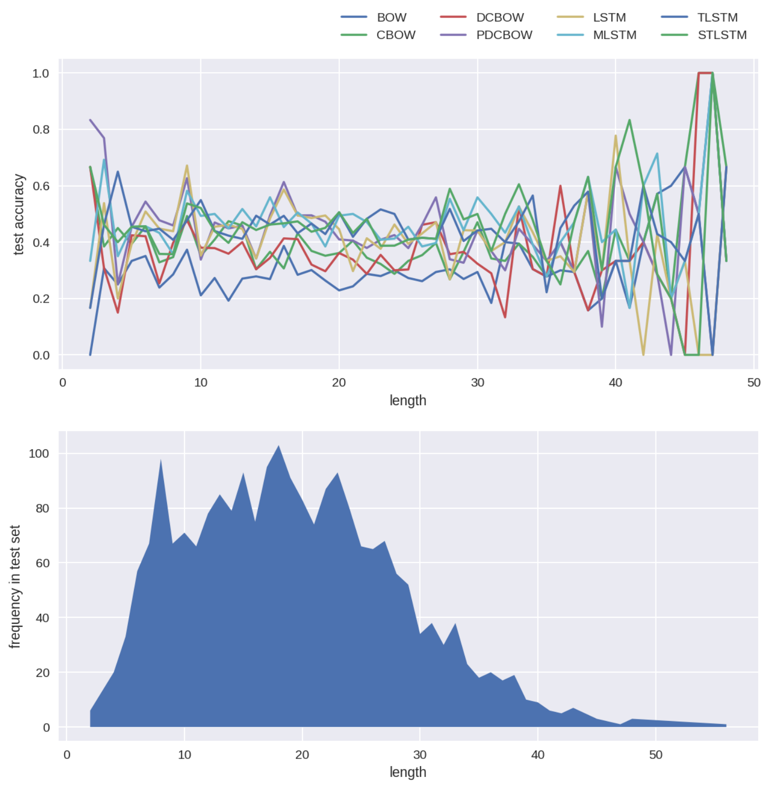

Figure 4: Accuracy and occurence frequency per sentence length.

BOW							
CBOW	$1.67e^{-13}$						
DCBOW	$1.90e^{-12}$	$7.50e^{-1}$					
PDCBOW	$3.57e^{-40}$	$4.61e^{-9}$	$5.93e^{-10}$				
LSTM	$9.81e^{-34}$	$2.54e^{-6}$	$4.74e^{-7}$	$2.53e^{-1}$			
MLSTM	$4.72e^{-50}$	$8.66e^{-14}$	$6.62e^{-15}$	1.13^{e-1}	6.16^{e-3}		
TLSTM	$1.76e^{-57}$	$1.18e^{-17}$	$6.22e^{-19}$	$7.42e^{-3}$	$1.27e^{-4}$	$2.80e^{-1}$	
STLSTM	$1.55e^{-50}$	$4.93e^{-14}$	3.68^{e-15}	$9.73e^{-2}$	$4.91e^{-3}$	$9.51e^{-1}$	$3.14e^{-1}$
model	BOW	CBOW	DCBOW	PDCBOW	LSTM	MLSTM	TLSTM

Figure 5: Significance tests between the models, with $p \leq 0.05$ indicating a significant difference in performance.