Practical 2 - Technical Report

Jochem Brandsema
14546620
jochem.brandsema@student.uva.nl

Joan Velja
14950480
joan.velja@student.uva.nl

1 Introduction

Sentiment analysis, a key component of understanding textual data, has widespread applications in several domains such as social media monitoring (Badjatiya et al., 2017) and Political sentiment orientation (Mullen and Malouf, 2006). This report focuses on the comparison of various neural network models for sentence representation in sentiment analysis, specifically using the Stanford Sentiment Treebank dataset [1].

Research Questions: Our study is guided by 5 key questions:

1. What is the role of word order in sentiment classification tasks?
2. Does incorporating tree structures enhance model accuracy?
3. How does model performance vary with sentence length?
4. What is the impact of supervising sentiment at each node in the tree?
5. How do the Child-Sum Tree-LSTM and the N-ary Tree-LSTM compare on this task?

Motivation: These questions are pivotal for a good understanding of the models we will implement. Understanding how these factors interact with the task can significantly improve the accuracy and applicability of sentiment classification models in real-world scenarios.

Expectations: We hypothesize that models sensitive to word order and tree structures will demonstrate superior performance, attributing to their ability to capture more contextual and structural information in text.

Methodology: Previous studies have explored various models for sentiment analysis. Our report includes the evaluation of models such as BOW, CBOW, Deep CBOW, LSTM, Binary Tree-LSTM and Child-Sum Tree-LSTM to address the posed research questions. Our approach involves evaluating the models in a comparative fashion, examining their performance on longer and shorter sentences, and observing how performance of the Tree-LSTM models changes when supervised at each node in the tree.

Summary of Findings: This study confirms the importance of word order in sentiment classification, as models from the LSTM family outperformed their BOW counterparts. Also tree structure proves to be useful, though the performance boost is quite modest. Tree-LSTMs however do prove to be more robust regarding sentence length compared to the regular LSTM model, the performance of which suffers when trying to classify longer sentences. Supervision at each tree node appears disadvantageous to the classification accuracy of Tree-LSTMs with our initial batch size, but actually improved performance for a higher batch size. Somewhat surprisingly, the Binary Tree-LSTM, optimized for the binary tree structure provided by the data, only marginally exceeds the more general Child-Sum Tree-LSTM.

2 Background

2.1 Bag of Words and Its Variants

The Bag of Words (BOW) model is a fundamental concept in NLP. It represents text by the frequency of words within it, ignoring grammar and word order (Harris, 1954). While BOW is widely used due to its simplicity, it fails to capture the semantic relationships between words, limiting its effectiveness in complex NLP tasks.

The Continuous Bag of Words (CBOW) leverages arbitrarily sized embeddings. It learns a linear layer to perform classification. These embeddings allow CBOW to capture some degree of semantic

[1] https://nlp.stanford.edu/sentiment/

meanings, but like BOW, it still treats text as an unordered collection of words.

Deep CBOW extends the above model by stacking multiple layers and non-linear activations, allowing the model to capture complex, non-linear relationships. However, the increased complexity also increases the risk of overfitting, particularly on smaller datasets. Finally, we can leverage pre-trained word embeddings, providing a richer representation of text (Mikolov et al., 2013a).

2.2 Word Embeddings

Word embeddings are a foundational aspect of modern NLP. They provide a way to represent words as vectors in a high-dimensional space, capturing semantic relationships between words. Techniques like GloVe (Global Vectors for Word Representation) and Word2Vec are prevalent, offering unique advantages in capturing word meanings (Pennington et al., 2014; Mikolov et al., 2013b). Embeddings are particularly beneficial in sentiment analysis for encoding contexual informations in vectors.

2.3 LSTM and Tree-LSTM

Long Short-Term Memory (LSTM) networks, a special kind of RNN, are adept at processing sequences of data, like text (Hochreiter and Schmidhuber, 1997). Unlike standard RNNs, LSTMs can capture long-term dependencies in data, making them suitable for understanding the context in sentences or paragraphs. This ability is critical in sentiment analysis where the sentiment often depends on the broader context of a sentence.

Building on the concept of LSTMs, Tree-LSTM incorporates tree structures into the network, allowing it to process data with hierarchical relationships (Tai et al., 2015; Le and Zuidema, 2015; Zhu et al., 2015). This is particularly relevant for our project, as the Stanford Sentiment Treebank provides sentences with an associated binary tree structure.

2.4 Relation Between Techniques

BOW and its variants provide foundational techniques for text representation, but their effectiveness is dramatically influenced by the complexity of the task and the richness of the dataset. Word embeddings provide the input layer for models of both some of the BOW and LSTM families, and Tree-LSTMs extend LSTMs to handle semantical structure in data. Understanding these relationships is key to appreciating the advancements in sentiment analysis models.

3 Models

3.1 Bag of Words (BOW) Model

Unlike traditional BOW models which rely on word frequency, our neural BOW model associates each word with a multi-dimensional vector, representing the sentiment conveyed by the word, that can be learned using gradient based methods. Each entry in a vector corresponds to one of the output classes. To classify a sentence, we sum the vectors of all words in the sentence along with a bias vector. This process, while straightforward, results in the loss of word order information. The resulting vector contains the logits for each class, so the final sentiment is determined by the *argmax* of this vector.

3.2 Continuous Bag of Words (CBOW) Model

In the CBOW setting, word embeddings have an arbitrary size, allowing the model to capture a larger share of information about each word. The technique is the same as BOW, and the summed vector of these embeddings is transformed to the output class size using a learned parameter matrix W.

3.3 Deep CBOW Model

The Deep CBOW model extends the CBOW architecture by adding two linear layers (for a total of three) and tanh activation functions between them.

3.4 Deep CBOW Model with Pre-Trained Embeddings

This variant of Deep CBOW, just like all LSTM models described hereafter, uses the GloVe word representations as static pre-trained embeddings, rather than learning the embeddings from scratch.

3.5 Long Short-Term Memory (LSTM) Model

Our LSTM model processes text data sequentially, capturing long-term dependencies that are crucial for understanding context. A linear layer with dropout was added to this and the other LSTM architectures to allow classification.

3.6 Binary Tree-LSTM Model

The Binary Tree-LSTM model leverages the binary tree structure provided by the Stanford Sentiment Treebank. It extends the capabilities of the standard LSTM to handle structured data, potentially offering a more accurate understanding of the sentiment expressed in different parts of a sentence.

3.7 Child-Sum Tree-LSTM Model

The Child-Sum Tree-LSTM model is a Tree-LSTM model that can handle any number of children per node, where the Binary Tree-LSTM only functions on binary trees. A drawback of this model is that it cannot perceive the order of a node's children.

4 Experiments

4.1 Task and Data Description

The task in our experiments is sentiment classification using the Stanford Sentiment Treebank (SST) dataset. The SST dataset consists of sentences, their binary tree structure, and fine-grained sentiment scores. A review consists of a single sentence, and we have a sentiment score for each node in the binary tree that makes up the sentence, including the root node (i.e., we still have an overall sentiment score for the entire review). The sentiment scores range from 0 (very negative) to 4 (very positive). We have 8544 training sentences, 1101 validation sentences and finally 2210 test sentences. Additionally, we employ an extended training set where every node from the original training set is treated as one sample. We also split the test set in two; one with shorter sentences and one with longer sentences, as to address question number 3.

4.2 Training and hyperparameters

The models are trained training set using the Adam optimization algorithm and the cross-entropy loss. We use a batch size of 1 for the BOW variants and 25 for the LSTM variants. All BOW models are trained for 30000 iterations with learning rate 0.0005 and all LSTM models for 3000 iterations with learning rate $2e{-}4$, with models being evaluated on the validation set every 1000 and 100 iterations respectively. After training, the version with the best accuracy on the validation set is returned as final model for further evaluation. This procedure was repeated with 5 different seeds.

All models use word embeddings of size 300, except for the vanilla BOW model, where the embedding size must equal the output size of 5. The used hidden sizes were 100 for Deep CBOW, 168 for LSTM and 150 for both Tree-LSTM models.

4.3 Evaluation

For evaluating the models, we use accuracy. The performance is calculated based on the models' ability to correctly classify the sentiment of the sentences in the test set of the SST dataset and in the two splits of this test set as mentioned before. As each model is trained and evaluated for every random seed, the mean test accuracy and the standard deviation across these runs is reported.

Model	Acc. (%)	Std. Dev.	Δ (%)
BOW	25.35	1.54	-
CBOW	34.53	1.78	+9.18
DCBOW	36.96	1.45	+2.43
PT DCBOW	43.32	0.51	+6.36
LSTM	46.13	0.45	+2.81
CS-TLSTM	46.66	0.63	+0.53
BTLSTM	47.62	0.43	+0.96

Table 1: Performance of different models on the dataset. Δ indicates the absolute percentage improvement.

5 Results

As it can be seen in table 1, there is a trend where more complex models that are able to capture more contextual information and dependencies in text (like LSTM, CS-TLSTM, BTLSTM) perform better than simpler models (like BOW and CBOW). As expected, BOW and CBOW, due do their underlying assumptions, fail to go above 26% and 35% accuracy respectively. Their approach to text representation ignores order and context, which is likely the reason for their poor performance. This is to be expected, as these two models do not even leverage nonlinearities. Results get slightly better with DeepCBOW, leveraging some degree of non-linearities, but biggest jump is provided by the implementation of pre-trained word embeddings, increasing the accuracy by 6%. This is once more to be expected, as GloVe provides the model with semantic information from the beginning, giving it a head start. To our surprise, LSTMs do not yield the improvement we expected: we only obtain a roughly a 3% absolute increase in accuracy going from the PT Deep CBOW to the Vanilla LSTM.

These results allow us to already confirm our initial claim on the importance of word order to the task. Models like LSTMs are specifically designed to address this aspect. Subsequently, tree structures allow models to capture not just the linear sequence of words, but also the hierarchical structure of language, as parsed in syntactic trees. This specific feature is helpful because for instance, it can help differentiate the scope of negation, although the observed improvement is very modest.

Tackling the issue of sequence length, we pro-

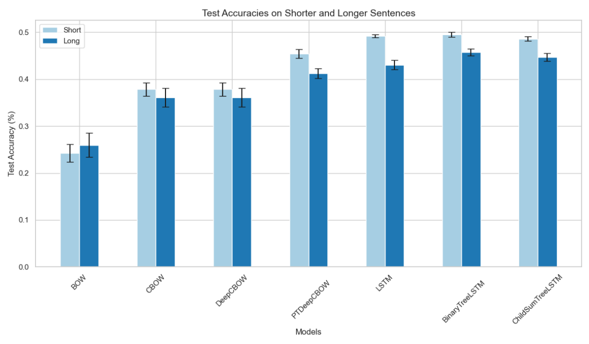

Figure 1: Model accuracies on shorter and longer sentences.

vide results in figure 1. The results show that the models have more difficulty with longer sentences compared to shorter sentences. This effect is more pronounced in the LSTM models than in the BOW models. Out of the LSTMs, the vanilla LSTM model shows the steepest decline in accuracy, where the effect seems to be less strong in the Tree-LSTM models. In regular LSTMs, the input signal has to travel through many cells, which can cause the model to have 'forgotten' input from the beginning by the time it gets to the end. This effect is mitigated in Tree-LSTMs, as the input doesn't have to go through as many cells. This explains why the regular LSTM may have more difficulty with longer sentences than the Tree-LSTMs.

As expected, BOW models show less variation in performance across different sentence lengths. As long as a sentence contains enough words indicating the correct sentiment relative to words pointing to other sentiments, sentence length is not be of major influence. The more surprising is the substantial drop in performance of the Deep CBOW with GloVe embeddings, where this is not as pronounced for the Deep CBOW where the embeddings were randomly initialized. We don't have a ready explanation for this.

Moving on to supervision at each node in the tree, we show results from training on the extended training set in table 2. It stands out that tree based models, when trained on all nodes in the trees under the same set of hyperparameters, perform worse than those trained on the regular training set. Initially we thought this might have been because the distribution of the training set had shifted: more and more shorter sentences were added in, while the test set had remained unchanged, meaning that the train and test set now have different distributions.

Model	Acc. (%)	Std. Dev.	Δ (%)
CS-TLSTM	45.51	1.68	-1.15
BTLSTM	45.30	0.58	-2.32

Table 2: Performance of tree-based models trained on the extended training set. Δ indicates the absolute percentage improvement compared to training on the original training set.

Model	Acc. (%)	Std. Dev.
CS-TLSTM	46.34	1.68
BTLSTM	48.81	0.88

Table 3: Performance of tree-based models trained on the extended training set, with batch size = 128

After some hyperparameter tuning to address increase in size of our training dataset, we found that training with `batch_size = 25` as suggested in the notebook was very detrimental: in academia (Ahmed et al., 2019), it is widely accepted to use batches in ranges 32-256 to leverage stable gradient computations, so we ran a further experiment with `batch_size = 128`. The results are provided at table 3. We find that, contrary to our belief, supervising at each node proves to improve model performance for both the Child Sum Tree LSTM and the Binary Tree LSTM: we claim that, by training on each node, we enhance granularity of learning, as the model effectively gets more data points, leading to more robust learning. Each subphrase (node), in this case, acts as an additional training example, thus leading to better generalization.

Finally, we address the performance of Child-Sum Tree-LSTM versus Binary Tree-LSTM. Since the former is designed to handle trees with a variable number of children and the latter is tailored toward binary trees, we expected the Binary Tree-LSTM to perform better. Table 1 shows this is indeed the case, but only marginally so. This is somewhat surprising, also given that the Binary Tree-LSTM has substantially more parameters than the Child-Sum Tree-LSTM.

6 Conclusion

Our experiments have reinforced the critical role of word order and sentence structure in sentiment analysis, highlighting that models which account for these elements outperform more traditional approaches. Furthermore, the integration of pre-trained word embeddings, has substantially enhanced model efficacy (and efficiency) by leveraging a pre-existing semantic framework. Additionally, our study has revealed that tree-based models demonstrate robustness in managing variable sentence lengths, endorsing the integration of linguistic structures into model design to better reflect the hierarchical nature of human language.

Our findings align with current literature, corroborating the established understanding that deep learning models, particularly those that incorporate sequential and contextual information, are generally superior for sentiment analysis tasks. The literature also consistently highlights the effectiveness of pre-trained embeddings in capturing latent semantic relationships that are not immediately evident from a surface-level analysis.

Unexpectedly, our experiments revealed a significant decline in LSTM performance with increased sentence length, suggesting a possible limitation in the model's ability to handle long-term dependencies without additional supportive mechanisms. We suspect that allowing for finetuning of the embeddings could prove beneficial to the improvement of the model. This is a potential direction for future research.

Based on our insights, we recommend further exploration into the implementation of attention mechanisms as to address performance dips in longer sentences (Ahmed et al., 2019).

References

Mahtab Ahmed, Muhammad Rifayat Samee, and Robert E. Mercer. 2019. Improving tree-lstm with tree attention. *CoRR*, abs/1901.00066.

Pinkesh Badjatiya, Shashank Gupta, Manish Gupta, and Vasudeva Varma. 2017. Deep learning for hate speech detection in tweets. *CoRR*, abs/1706.00188.

Zellig S Harris. 1954. Distributional structure. *Word*, 10(2-3):146–162.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Phong Le and Willem Zuidema. 2015. Compositional distributional semantics with long short term memory.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013a. Efficient estimation of word representations in vector space. Cite arxiv:1301.3781.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013b. Distributed representations of words and phrases and their compositionality.

Tony Mullen and Robert Malouf. 2006. A preliminary investigation into sentiment analysis of informal political discourse. In *AAAI spring symposium: computational approaches to analyzing weblogs*, pages 159–162.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Kai Sheng Tai, Richard Socher, and Christopher D. Manning. 2015. Improved semantic representations from tree-structured long short-term memory networks.

Xiaodan Zhu, Parinaz Sobihani, and Hongyu Guo. 2015. Long short-term memory over recursive structures. In *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 1604–1612, Lille, France. PMLR.