To Parse or Not to Parse: An Experimental Comparison of RNTNs and CNNs for Sentiment Analysis

Zahra Ahmadi[1], Aleksandrs Stier[1], Marcin Skowron[2], and Stefan Kramer[1]

[1] Institut für Informatik, Johannes Gutenberg-Universität, Mainz, Germany
[2] Austrian Research Institute for Artificial Intelligence, Vienna, Austria
zaahmadi@uni-mainz.de, stier@students.uni-mainz.de,
marcin.skowron@ofai.at, kramer@informatik.uni-mainz.de

Abstract. Recent years have seen a variety of different deep learning architectures for sentiment analysis. However, little is known about their comparative performance and merits on a common ground, across a variety of datasets, and on the same level of optimization. In this paper, we provide such a comparison for two popular architectures, Recursive Neural Tensor Networks (RNTNs) and Convolutional Neural Networks (CNNs). Although RNTNs have been shown to work well in many cases, they require intensive manual labeling due to the so-called vanishing gradient problem. To enable an extensive comparison of the two architectures, this paper employs two methods to automatically label the internal nodes: a rule-based method and (this time as part of the RNTN method) a convolutional neural network. This enables us to compare these RNTN models to a relatively simple CNN architecture. On almost all benchmark datasets the CNN architecture outperforms the variants of RNTNs tested in the paper. These results suggest that CNNs already offer good predictive performance and, at the same time, more research on RNTNs would be needed to further exploit sentence structure.

1 Introduction

The advent of social media such as twitter, blogs, ratings and reviews has created a surge of research on the task of sentiment analysis especially for short texts such as sentences [17, 16]. However, a single sentence has a limited amount of contextual data which makes its sentiment prediction challenging. To effectively solve this problem, one may model sentences to analyze and represent their semantic content. Neural network based sentence modeling approaches have been increasingly considered [12, 19, 6] for their significant advantages of removed requirements for feature engineering and preservation of the word order and syntactic structures, in contrast to the traditional bag-of-words model, where sentences are encoded as unordered collections of words.

Most existing neural network models in the context of sentence classification fall into one of two groups: Recursive Neural Networks (RecNNs) and Convolutional Neural Networks (CNNs). RecNNs have shown excellent abilities to model word combinations in a sentence. However, they depend on well-performing parsers to provide the topological structure. These are not available for many languages and do not perform well in noisy domains. Further, they often require labeling of all phrases in sentences to reduce the so-called *vanishing gradient problem* [5]. On the other hand, CNN models apply a convolution operator sequentially on word vectors using sliding windows. Each sentence is treated individually as a bag of n-grams, and long-range dependency information spanning multiple sliding windows is therefore lost [20]. Another limitation of CNN models is their requirement for the exact specification of their architecture and hyperparameters [21].

We conducted extensive experiments over a range of benchmark datasets to compare the two network architectures: RNTNs and CNNs. Our goal is to provide an in-depth analysis on how these models perform across different settings. Such a comparison is missing in the literature, likely because recursive networks often require labor-intensive manual labeling of phrases. Such annotations are unavailable for many benchmark datasets. We propose two methods to label the internal phrases automatically and also investigate whether there is an effect of using constituency parsing instead of dependency parsing in the RNTN model. In this way, we aim to contribute to a better understanding of the limitations of the two network models and how to improve them.

The remainder of this paper is organized as follows: A brief review on the related literature is presented in Section 2. Section 3 explains the details of network architectures. In Section 4, results of the experiments on common benchmarks are discussed, and finally, Section 5 concludes the paper.

2 Related Work

Neural network approaches which are used in sentiment analysis range from basic Neural Bag-of-Words (NBoW) to more representative compositional approaches such as RecNNs [18, 4], CNNs [7, 6], and LSTM models [10, 22].

Recursive neural networks [15, 3] work by feeding an external parse tree to the network. At every node in the tree, the composition is done in a bottom-up fashion by a weight matrix shared over all nodes of the tree. Recurrent Neural Networks (RNN) are a special case of recursive networks where their structure is linear instead of a tree [12]. An in-depth comparison of RecNN and RNN showed that when long-distance semantic dependencies play a role, recursive models offer useful power [10]. Yet RecNNs implicitly model the interaction among input vectors, whereas Recursive Neural Tensor Networks (RNTNs) have been proposed to allow more explicit interactions [19].

CNNs, as the alternative models for predicting sentiment, apply one-dimensional convolution kernels in sequential order to extract local features. Recently, new architectures have been proposed to resolve the limitation of CNNs in losing long-range dependency information [11, 20], or to overcome the fixed structure of CNNs for one input length [6].

3 Method

In this section we first present our approach to the automatic labeling of RNTNs and then explain our proposed architecture for the CNN.

$$p_3 = f\left(\begin{bmatrix}a_1\\p_2\end{bmatrix}^T \mathbf{V}^{[1:d]}\begin{bmatrix}a_1\\p_2\end{bmatrix} + \mathbf{W}\begin{bmatrix}a_1\\p_2\end{bmatrix}\right)$$

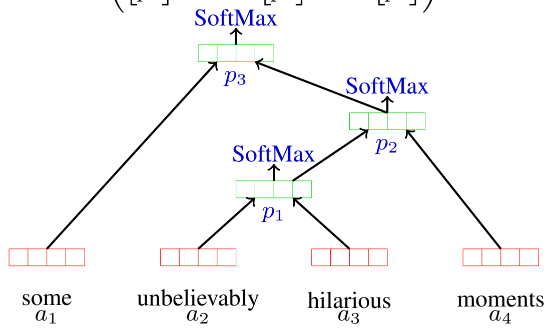

Fig. 1: An example of an RNTN architecture with word vector dimension of size 4 for sentiment classification of a given input sequence, which is parsed by a constituency parser. $\mathbf{V}$ and $\mathbf{W}$ are the tensor matrix and the recursive weight matrix, respectively.

Recursive Neural Tensor Network Architecture. RNTNs [19] are a generalization of RecNNs where the interactions among input vectors are encoded in a single composition function (Figure 1). Here, we propose two methods to make the labeling process automatic:

- **Rule-based method:** The RNTN model was first proposed for sentiment analysis purposes. Hence, our first approach uses a rule-based method to determine the opinion of a phrase. We use four types of dictionaries: A dictionary of sentiments consisting of $6,360$ entries with a sentiment range of $[-3, +3]$, a negation dictionary consisting of 28 entries, a dictionary of intensifier terms consisting of 47 words with a weight range of $[1, 3]$, and a dictionary of diminishers consisting of 26 entries with a weight range of $[-3, -1]$. For any phrase, we start analyzing from the end backwards to the beginning: If any sentiment term found, we search for an intensifier/diminisher term to increase/decrease the absolute value of the sentiment. Then we search for a negation term. If one is found and there is no intensifier/diminisher before the sentiment term, the sentiment is reversed; otherwise if the phrase includes both the negation term and an intensifier/diminisher, the sentiment is set to weak negative.
- **CNN-based method:** A more general approach for labeling the phrases is to use a pre-trained CNN model. We use the architecture proposed here (see below for the description) to train a model on the sentence level, and use the resulting model to label the internal phrases for the RNTN. In this way, we could apply the RNTN to domains other than sentiment classification as well.

Convolutional Neural Network Architecture. Deep convolutional neural networks have led to a series of breakthrough results in image classification. Although recent evidence shows that network depth is of crucial importance to obtain better results [2], most of the models in the sentiment analysis and sentence modeling literature use a simple architecture (e.g. [7] uses a one-layer CNN). Inspired by the success of CNNs in image classification, our goal is to expand the convolution and Max-Pooling layers in order to

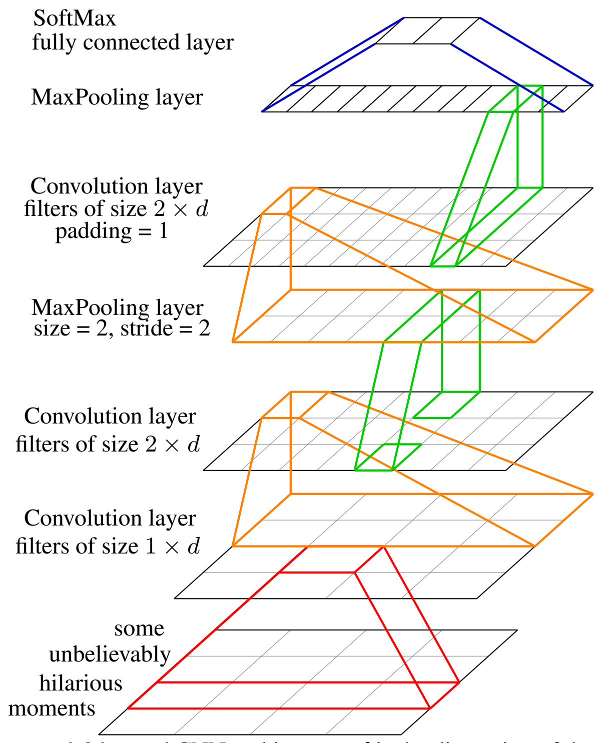

Fig. 2: Our proposed 6-layered CNN architecture. d is the dimension of the word vector.

achieve better performance by deepening the models and adding higher non-linearity to the structure. However, deeper models are also more difficult to train [2]. To reduce the computational complexity, we choose small filter sizes. In our experiments, we have come up with a simple CNN model that consists of six layers (Figure 2): The first layer applies $1 \times d$ filters on the word vectors, where d is the word vector dimension. The essence of adding such a layer to the network is to derive more meaningful features from word vectors for every single word before feeding them to the rest of the network. This helps us achieving better performance since the original word vectors capture only sparse information about the words' labels. In contrast to our proposed layer, [7] uses a so-called *non-static* approach to modify the word vectors through the training phase.

The second layer of our CNN model is again a convolution layer with the filters of size $2 \times d$. The output of this layer is fed into a Max-Pooling layer with pooling size and stride 2. The reason for applying such a Max-Pooling layer in the middle layers of the network is to reduce the dimensionality and to speed up the training phase. This layer does not have notable effect on the accuracy of the resulting model. Next, on the fourth layer, convolving filters of size $2 \times d$ with a padding size 1 are again applied to the output of previous layer. Padding preserves the original input size. The next layer applies Max-Pooling to the whole input at once. Using bigger pooling sizes leads to better results [21]. Finally, the last layer is a fully connected SoftMax layer which outputs the probability distribution over the labels.

Table 1: Summary statistics for the datasets. c, N_{tr}, N_{tu} and N_{ts} indicate the number of labels, number of training sentences, number of tuning sentences and the number of test sentences, respectively.

Dataset	c	N_{tr}/N_{tu}	N_{ts}
MR	2	10662	CV
SST-2	2	6920/872	1821
SST-5	5	8544/1101	2210
SemEval-2016	3	12644/3001	20632

Table 2: Performance comparison on all datasets. Accuracy and F-measure is averaged over all the classes. n/a indicates non-defined cases as one of the classes was misclassified completely. If an experiment was not applicable, the cell is left with a dash: SST-2 internal phrases belong to three class (negative, neutral, positive), however, the output of the corresponding CNN model has only two labels.

Dataset	RNTN								CNN		Rule-based	
	Constituency parser				Dependency parser							
	Rule		CNN		Rule		CNN					
	Acc.	F1	Acc.	F1	Acc.	F1	Acc.	F1	Acc.	F1	Acc.	F1
MR	0.63	0.63	0.70	0.70	0.50	0.50	0.49	0.49	**0.71**	**0.71**	0.64	0.64
SST-2	0.70	0.70	-	-	0.56	0.56	-	-	**0.77**	**0.77**	0.69	0.69
SST-5	0.30	0.28	0.34	0.21	0.30	**0.29**	0.30	n/a	**0.37**	0.26	0.31	**0.29**
SemEval-2016	0.53	0.45	0.52	0.51	0.52	0.45	0.50	0.49	**0.56**	**0.56**	0.53	0.52

4 Experiments

In this section, we first introduce the benchmark datasets and experimental settings, then we will investigate the variants of RNTNs and compare their performance to the proposed CNN architecture.

4.1 Datasets

We compare the models on a set of commonly applied benchmark datasets (Table 1): The Movie Review (**MR**) dataset[3] was extracted from Rotten Tomato reviews [13], where the reviews can be positive or negative. As MR dataset does not have a separate test set, we use 10-fold cross-validation in the experiments. An extended version of MR dataset relabeled by Socher et al. [19] in the Stanford Sentiment Treebank (**SST-5**)[4] has five fine-grained labels: negative, somewhat negative, neutral, somewhat positive and positive. A binary version of the SST-5 dataset (**SST-2**) was created by removing the neutral sentences and assigning the remaining to two classes: negative or positive. The

[3] `https://www.cs.cornell.edu/people/pabo/movie-review-data/`

[4] `http://nlp.stanford.edu/sentiment/Data`

Table 3: Confusion matrix of rule-based RNTN (left) vs. CNN-based RNTN (right) on the SST-5 phrase set.

		Predicted labels				
		-2	-1	0	+1	+2
Actual	-2	3239	4112	2360	1402	177
	-1	8936	12409	20399	5447	888
	0	9107	11462	249850	15662	1965
	+1	2763	3994	26119	23413	4526
	+2	691	909	2521	9550	3766

		Predicted labels				
		-2	-1	0	+1	+2
Actual	-2	1154	7755	248	2084	49
	-1	2514	30581	4455	10153	376
	0	1511	132379	106202	44987	2967
	+1	232	18540	9280	26838	5925
	+2	12	2578	1889	8864	4094

SemEval-2016[5] dataset is a set of tweets and was provided by the SemEval contest. Tweets were labeled by one of the three labels: negative, neutral and positive.

4.2 Experimental Settings

In our experiments, we use the pre-trained Glove [14] word vector models[6]: On the SemEval-2016 dataset, we use Twitter specific word vectors that were trained on 2 billion tweets. On other datasets, we use the model trained on the web data from Common Crawl which contains a case-sensitive vocabulary of size 2.2 million. In all the experiments, the size of the word vector, the minibatch and the epochs were set to 25, 20 and 100, respectively. We use $f = tanh$ and a learning rate of 0.01 in all the RNTN models. In CNN models, the number of filters in the convolutional layers are set to 100, 200 and 300, respectively; and the maximum length of the sentences is 32. For shorter sentences, they are padded with zero vectors. In RNTN models which use constituency parsers, we use the Stanford parser [8]. For those models which use dependency parsers, we use the Tweebo parser [9] – a dependency parser specifically developed for Twitter data – for the SemEval-2016 dataset and on the rest of the datasets, we use the Stanford neural network dependency parser [1].

4.3 Results

In this section, we present the results of automatic labeling of phrases, the effect of the selected parser type, and describe the overall evaluation results for the presented RNTN and CNN models. Finally we discuss the effect of automatic labeling on the performance of the RNTN.

- **Comparison of automatic labeling methods:** We first use the manually labeled SST-5 dataset to test the effectiveness of our automatic labeling methods. We extract all the possible phrases of the whole dataset with respect to their parse trees and use our rule-based method to label them. The accuracy of the rule-based method is 69% and its confusion matrix is reported in Table 3 (left). In the next step we train a CNN model on the training sentences and use the resulting model to label the phrases. The

[5] `http://alt.qcri.org/semeval2016/task4/`
[6] `http://nlp.stanford.edu/projects/glove/`

Table 4: Performance comparison of various labeling methods for the RNTN on the SST-5 dataset.

	Manual labeling	Rule-based method	CNN-based method
Accuracy	**0.41**	0.30	0.34
F1-measure	**0.32**	0.27	0.21

Table 5: Confusion matrix of the manually labeled RNTN (left) vs. the CNN model (right) on the SST-5 test set.

		Predicted labels				
		-2	-1	0	+1	+2
Actual	-2	1	237	21	19	1
	-1	0	455	70	103	5
	0	1	190	72	113	13
	+1	0	129	47	267	67
	+2	0	38	26	218	117

		Predicted labels				
		-2	-1	0	+1	+2
Actual	-2	15	187	0	77	0
	-1	6	426	1	198	2
	0	0	215	7	164	3
	+1	0	165	0	315	30
	+2	0	65	0	277	57

accuracy of the CNN model labeling is 40% and the corresponding confusion matrix is presented in Table 3 (right). Although the accuracy of the CNN model is far lower than that of the rule-based method, we observe that the CNN is a better model to correctly classify positive and negative classes than the rule-based method. In turn, the rule-based method is superior in the classification of the neutral class.

- **Constituency parser vs. dependency parser:** When analyzing the effect of using a dependency parser instead of a constituency parser in RNTNs (Table 2), for some datasets (e.g. MR) a significant loss of performance is visible. This is particularly noticeable when the labeling method is CNN (e.g. 70% to 49% in MR). The reason for this observation could be the difference of the word order resulting from a dependency parser compared to the n-gram features extracted by the CNN.
- **RNTN vs. CNN:** Table 2 shows a detailed comparison of the RNTN version to the CNN model and the rule-based method. With the same settings of parameters, we see a better performance of the CNN model on all the datasets, with the exception of the SST-5 dataset. The largest performance (in terms of F-measure) improvement can be observed on the SST-2 and SemEval-2016 datasets, 0.70 and 0.51; and 0.77 and 0.56, respectively, for the best performing RNTN and CNN approaches. The possible reasons may be related to the enormously large number of parameters that have to be optimized in the tensor and the effects of the applied automatic labeling of phrases used on the RNTN. Therefore, a future research direction could try to reduce this space and find a better initialization.
- **Effect of automatic labeling on RNTN performance:** Table 4 presents the performance of different versions of the RNTN trained on the manually labeled SST-5 dataset versus the rule-based and CNN-based automatic labeling variants. As we can see, automatic labeling will result in a significant degradation of performance on SST-5. Comparing the results with the CNN model in Table 2 shows that the manually labeled RNTN outperforms the CNN architecture in terms of overall accuracy

and F-measure. Looking into the confusion matrix of both methods (Table 5) indicates that the RNTN is better at predicting neutral and positive labels while the CNN is better at classifying negative and more positive labels. Unfortunately, currently there is no other dataset that is manually labeled at the phrase level. A future direction could be further evaluating the impacts of the phrase labeling accuracy on various datasets.

5 Conclusions

In this paper we proposed two methods to automatically label the internal nodes of recursive networks to reduce the labor-intensive task of manually labeling the phrases in predicting the sentiment of the sentences. We then conducted an in-depth study of the RNTN model and compared the model to a relatively simple CNN architecture. Experimental results demonstrate that the proposed CNN model outperforms the RNTN variants. The findings also show that there is still room for improvement of RNTNs in terms of determining tensor functions in a more informed manner.

Acknowledgement

The authors thank PRIME Research for supporting the first author during her research time. The third author is supported by the Austrian Science Fund (FWF): P27530-N15.

References

1. Chen, D., Manning, C.D.: A fast and accurate dependency parser using neural networks. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing. pp. 740–750 (2014)
2. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: IEEE Conference on Computer Vision and Pattern Recognition. pp. 770–778 (2016)
3. Hermann, K.M., Blunsom, P.: The role of syntax in vector space models of compositional semantics. In: Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics. pp. 894–904 (2013)
4. Irsoy, O., Cardie, C.: Deep recursive neural networks for compositionality in language. In: Advances in Neural Information Processing Systems. pp. 2096–2104 (2014)
5. Iyyer, M., Manjunatha, V., Boyd-Graber, J., Daumé III, H.: Deep unordered composition rivals syntactic methods for text classification. In: Proceedings of 53rd Annual Meeting of the Association for Computational Linguistics. pp. 1681–1691 (2015)
6. Kalchbrenner, N., Grefenstette, E., Blunsom, P.: A convolutional neural network for modelling sentences. In: Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics. pp. 655–665 (2014)
7. Kim, Y.: Convolutional neural networks for sentence classification. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing. pp. 1746–1751 (2014)
8. Klein, D., Manning, C.D.: Accurate unlexicalized parsing. In: Proceedings of the 41st Annual Meeting on Association for Computational Linguistics. pp. 423–430 (2003)

9. Kong, L., Schneider, N., Swayamdipta, S., Bhatia, A., Dyer, C., Smith, N.A.: A dependency parser for tweets. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing. pp. 1001–1012 (2014)
10. Li, J., Luong, M.T., Jurafsky, D., Hovy, E.: When are tree structures necessary for deep learning of representations? In: Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning. pp. 2304–2314 (2015)
11. Ma, M., Huang, L., Xiang, B., Zhou, B.: Dependency-based convolutional neural networks for sentence embedding. In: Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing. pp. 174–179 (2015)
12. Mikolov, T., Karafiát, M., Burget, L., Cernockỳ, J., Khudanpur, S.: Recurrent neural network based language model. In: 11th Annual Conference of the International Speech Communication Association. pp. 1045–1048 (2010)
13. Pang, B., Lee, L.: Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In: Proceedings of the 43rd annual meeting on association for computational linguistics. pp. 115–124 (2005)
14. Pennington, J., Socher, R., Manning, C.D.: Glove: Global vectors for word representation. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing. pp. 1532–1543 (2014)
15. Pollack, J.B.: Recursive distributed representations. Artificial Intelligence 46(1), 77–105 (1990)
16. Reforgiato Recupero, D., Presutti, V., Consoli, S., Gangemi, A., Nuzzolese, A.G.: Sentilo: Frame-based sentiment analysis. Cognitive Computation 7(2), 211–225 (2015)
17. Rexha, A., Kröll, M., Dragoni, M., Kern, R.: Polarity classification for target phrases in tweets: A word2vec approach. In: Sack, H., Rizzo, G., Steinmetz, N., Mladenić, D., Auer, S., Lange, C. (eds.) The Semantic Web: ESWC 2016 Satellite Events. pp. 217–223 (2016)
18. Socher, R., Huval, B., Manning, C.D., Ng, A.Y.: Semantic compositionality through recursive matrix-vector spaces. In: Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning. pp. 1201–1211 (2012)
19. Socher, R., Perelygin, A., Wu, J.Y., Chuang, J., Manning, C.D., Ng, A.Y., Potts, C.P.: Recursive deep models for semantic compositionality over a sentiment treebank. In: Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing. pp. 1631–1642 (2013)
20. Zhang, R., Lee, H., Radev, D.: Dependency sensitive convolutional neural networks for modeling sentences and documents. In: The 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. pp. 1512–1521 (2016)
21. Zhang, Y., Wallace, B.: A sensitivity analysis of (and practitioners' guide to) convolutional neural networks for sentence classification. CoRR abs/1510.03820 (2015)
22. Zhu, X., Sobhani, P., Guo, H.: Long short-term memory over recursive structures. In: Proceedings of the 32nd International Conference on Machine Learning. pp. 1604–1612 (2015)