Adaptive Recursive Neural Network for Target-dependent Twitter Sentiment Classification

Li Dong[†*] **Furu Wei**[‡] **Chuanqi Tan**[†*] **Duyu Tang**[ᵀ*] **Ming Zhou**[‡] **Ke Xu**[†]
[†]Beihang University, Beijing, China
[‡]Microsoft Research, Beijing, China
[ᵀ]Harbin Institute of Technology, Harbin, China
donglixp@gmail.com fuwei@microsoft.com {ysjtcq,tangduyu}@gmail.com
mingzhou@microsoft.com kexu@nlsde.buaa.edu.cn

Abstract

We propose Adaptive Recursive Neural Network (AdaRNN) for target-dependent Twitter sentiment classification. AdaRNN adaptively propagates the sentiments of words to target depending on the context and syntactic relationships between them. It consists of more than one composition functions, and we model the adaptive sentiment propagations as distributions over these composition functions. The experimental studies illustrate that AdaRNN improves the baseline methods. Furthermore, we introduce a manually annotated dataset for target-dependent Twitter sentiment analysis.

1 Introduction

Twitter becomes one of the most popular social networking sites, which allows the users to read and post messages (i.e. tweets) up to 140 characters. Among the great varieties of topics, people in Twitter tend to express their opinions for the brands, celebrities, products and public events. As a result, it attracts much attention to estimate the crowd's sentiments in Twitter.

For the tweets, our task is to classify their sentiments for a given target as positive, negative, and neutral. People may mention several entities (or targets) in one tweet, which affects the availabilities for most of existing methods. For example, the tweet "*@ballmer: windows phone is better than ios!*" has three targets (*@ballmer*, *windows phone*, and *ios*). The user expresses neutral, positive, and negative sentiments for them, respectively. If target information is ignored, it is difficult to obtain the correct sentiment for a specified target. For target-dependent sentiment classification, the manual evaluation of Jiang et al. (2011) show that about 40% of errors are caused by not considering the targets in classification.

The features used in traditional learning-based methods (Pang et al., 2002; Nakagawa et al., 2010) are independent to the targets, hence the results are computed despite what the targets are. Hu and Liu (2004) regard the features of products as targets, and sentiments for them are heuristically determined by the dominant opinion words. Jiang et al. (2011) combine the target-independent features (content and lexicon) and target-dependent features (rules based on the dependency parsing results) together in subjectivity classification and polarity classification for tweets.

In this paper, we mainly focus on integrating target information with Recursive Neural Network (RNN) to leverage the ability of deep learning models. The neural models use distributed representation (Hinton, 1986; Rumelhart et al., 1986; Bengio et al., 2003) to automatically learn features for target-dependent sentiment classification. RNN utilizes the recursive structure of text, and it has achieved state-of-the-art sentiment analysis results for movie review dataset (Socher et al., 2012; Socher et al., 2013). The recursive neural models employ the semantic composition functions, which enables them to handle the complex compositionalities in sentiment analysis.

Specifically, we propose a framework which learns to propagate the sentiments of words towards the target depending on context and syntactic structure. We employ a novel adaptive multi-compositionality layer in recursive neural network, which is named as *AdaRNN* (Dong et al., 2014). It consists of more than one composition functions, and we model the adaptive sentiment propagations as learning distributions over these composition functions. We automatically learn the composition functions and how to select them from supervisions, instead of choosing them heuristically or by hand-crafted rules. AdaRNN

[*]Contribution during internship at Microsoft Research.

Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Short Papers), pages 49–54,
Baltimore, Maryland, USA, June 23-25 2014.

determines how to propagate the sentiments towards the target and handles the negation or intensification phenomena (Taboada et al., 2011) in sentiment analysis. In addition, we introduce a manually annotated dataset, and conduct extensive experiments on it. The experimental results suggest that our approach yields better performances than the baseline methods.

2 RNN: Recursive Neural Network

RNN (Socher et al., 2011) represents the phrases and words as D-dimensional vectors. It performs compositions based on the binary trees, and obtain the vector representations in a bottom-up way.

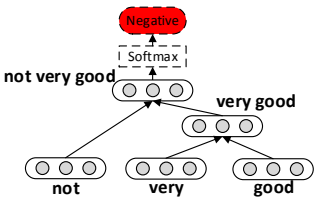

Figure 1: The composition process for "*not very good*" in Recursive Neural Network.

As illustrated in Figure 1, we obtain the representation of "*very good*" by the composition of "*very*" and "*good*", and the representation of trigram "*not very good*" is recursively obtained by the vectors of "*not*" and "*very good*". The dimensions of parent node are calculated by linear combination of the child vectors' dimensions. The vector representation $\mathbf{v}$ is obtained via:

$$\mathbf{v} = f\left(g\left(\mathbf{v}_l, \mathbf{v}_r\right)\right) = f\left(\mathbf{W}\begin{bmatrix}\mathbf{v}_l \\ \mathbf{v}_r\end{bmatrix} + \mathbf{b}\right) \quad (1)$$

where $\mathbf{v}_l, \mathbf{v}_r$ are the vectors of its left and right child, g is the composition function, f is the nonlinearity function (such as tanh, sigmoid, softsign, etc.), $\mathbf{W} \in \mathbb{R}^{D\times 2D}$ is the composition matrix, and $\mathbf{b}$ is the bias vector. The dimension of $\mathbf{v}$ is the same as its child vectors, and it is recursively used in the next step. Notably, the word vectors in the leaf nodes are regarded as the parameters, and will be updated according to the supervisions.

The vector representation of root node is then fed into a softmax classifier to predict the label. The k-th element of $\text{softmax}(\mathbf{x})$ is $\frac{\exp\{\mathbf{x}_k\}}{\sum_j \exp\{\mathbf{x}_j\}}$. For a vector, the softmax obtains the distribution over K classes. Specifically, the predicted distribution is $\mathbf{y} = \text{softmax}\left(\mathbf{U}\mathbf{v}\right)$, where $\mathbf{y}$ is the predicted distribution, $\mathbf{U} \in \mathbb{R}^{K\times D}$ is the classification matrix, and $\mathbf{v}$ is the vector representation of node.

3 Our Approach

We use the dependency parsing results to find the words syntactically connected with the interested target. Adaptive Recursive Neural Network is proposed to propagate the sentiments of words to the target node. We model the adaptive sentiment propagations as semantic compositions. The computation process is conducted in a bottom-up manner, and the vector representations are computed recursively. After we obtain the representation of target node, a classifier is used to predict the sentiment label according to the vector.

In Section 3.1, we show how to build recursive structure for target using the dependency parsing results. In Section 3.2, we propose Adaptive Recursive Neural Network and use it for target-dependent sentiment analysis.

3.1 Build Recursive Structure

The dependency tree indicates the dependency relations between words. As described above, we propagate the sentiments of words to the target. Hence the target is placed at the root node to combine with its connected words recursively. The dependency relation types are remained to guide the sentiment propagations in our model.

Algorithm 1 Convert Dependency Tree

Input: Target node, Dependency tree
Output: Converted tree

```
function CONV(r)
    E_r ← SORT(dep edges connected with r)
    v ← r
    for (r -t-> u/u -t-> r) in E_r do
        if r is head of u then
            w ← node with CONV(u), v as children
        else
            w ← node with v, CONV(u) as children
        v ← w
    return v
Call CONV(target node) to get converted tree
```

As illustrated in the Algorithm 1, we recursively convert the dependency tree starting from the target node. We find all the words connected to the target, and these words are combined with target node by certain order. Every combination is considered as once propagation of sentiments. If the target is head of the connected words, the target vector is combined as the right node; if otherwise, it is combined as the left node. This ensures the

child nodes in a certain order. We use two rules to determine the order of combinations: (1) the words whose head is the target in dependency tree are first combined, and then the rest of connected words are combined; (2) if the first rule cannot determine the order, the connected words are sorted by their positions in sentence from right to left. Notably, the conversion is performed recursively for the connected words and the dependency relation types are remained. Figure 2 shows the converted results for different targets in one sentence.

3.2 AdaRNN: Adaptive Recursive Neural Network

RNN employs one global matrix to linearly combine the elements of vectors. Sometimes it is challenging to obtain a single powerful function to model the semantic composition, which motivates us to propose AdaRNN. The basic idea of AdaRNN is to use more than one composition functions and adaptively select them depending on the linguistic tags and the combined vectors. The model learns to propagate the sentiments of words by using the different composition functions.

Figure 2 shows the computation process for the example sentence "*windows is better than ios*", where the user expresses positive sentiment towards *windows* and negative sentiment to *ios*. For the targets, the order of compositions and the dependency types are different. AdaRNN adaptively selects the composition functions $g_1 \dots g_C$ depending on the child vectors and the linguistic types. Thus it is able to determine how to propagate the sentiments of words towards the target.

Based on RNN described in Section 2, we define the composition result $\mathbf{v}$ in AdaRNN as:

$$\mathbf{v} = f\left(\sum_{h=1}^{C} P\left(g_h|\mathbf{v}_l, \mathbf{v}_r, \mathbf{e}\right) g_h\left(\mathbf{v}_l, \mathbf{v}_r\right)\right) \quad (2)$$

where $g_1, \dots, g_C$ are the composition functions, $P\left(g_h|\mathbf{v}_l, \mathbf{v}_r, \mathbf{e}\right)$ is the probability of employing g_h given the child vectors $\mathbf{v}_l, \mathbf{v}_r$ and external feature vector $\mathbf{e}$, and f is the nonlinearity function. For the composition functions, we use the same forms as in Equation (1), i.e., we have C composition matrices $W_1 \dots W_C$. We define the distribution over these composition functions as:

$$\begin{bmatrix} P\left(g_1|\mathbf{v}_l, \mathbf{v}_r, \mathbf{e}\right) \\ \vdots \\ P\left(g_C|\mathbf{v}_l, \mathbf{v}_r, \mathbf{e}\right) \end{bmatrix} = \text{softmax}\left(\beta S \begin{bmatrix} \mathbf{v}_l \\ \mathbf{v}_r \\ \mathbf{e} \end{bmatrix}\right) \quad (3)$$

where β is the hyper-parameter, $S \in \mathbb{R}^{C \times (2D+|\mathbf{e}|)}$ is the matrix used to determine which composition function we use, $\mathbf{v}_l, \mathbf{v}_r$ are the left and right child vectors, and $\mathbf{e}$ are external feature vector. In this work, $\mathbf{e}$ is a one-hot binary feature vector which indicates what the dependency type is. If relation is the k-th type, we set $\mathbf{e}_k$ to 1 and the others to 0.

Adding β in softmax function is a widely used parametrization method in statistical mechanics, which is known as Boltzmann distribution and Gibbs measure (Georgii, 2011). When $\beta = 0$, this function produces a uniform distribution; when $\beta = 1$, it is the same as softmax function; when $\beta \to \infty$, it only activates the dimension with maximum weight, and sets its probability to 1.

3.3 Model Training

We use the representation of root node as the features, and feed them into the softmax classifier to predict the distribution over classes. We define the ground truth vector $\mathbf{t}$ as a binary vector. If the k-th class is the label, only $\mathbf{t}_k$ is 1 and the others are 0. Our goal is to minimize the cross-entropy error between the predicted distribution $\mathbf{y}$ and ground truth distribution $\mathbf{t}$. For each training instance, we define the objective function as:

$$\min_{\Theta} \; -\sum_{j} \mathbf{t}_j \log \mathbf{y}_j + \sum_{\theta \in \Theta} \lambda_\theta \|\theta\|_2^2 \quad (4)$$

where Θ represents the parameters, and the L_2-regularization penalty is used.

Based on the converted tree, we employ backpropagation algorithm (Rumelhart et al., 1986) to propagate the errors from root node to the leaf nodes. We calculate the derivatives to update the parameters. The AdaGrad (Duchi et al., 2011) is employed to solve this optimization problem.

4 Experiments

As people tend to post comments for the celebrities, products, and companies, we use these keywords (such as "*bill gates*", "*taylor swift*", "*xbox*", "*windows 7*", "*google*") to query the Twitter API. After obtaining the tweets, we manually annotate the sentiment labels (negative, neutral, positive) for these targets. In order to eliminate the effects of data imbalance problem, we randomly sample the tweets and make the data balanced. The negative, neutral, positive classes account for 25%, 50%, 25%, respectively. Training data consists of 6,248 tweets, and testing data has 692

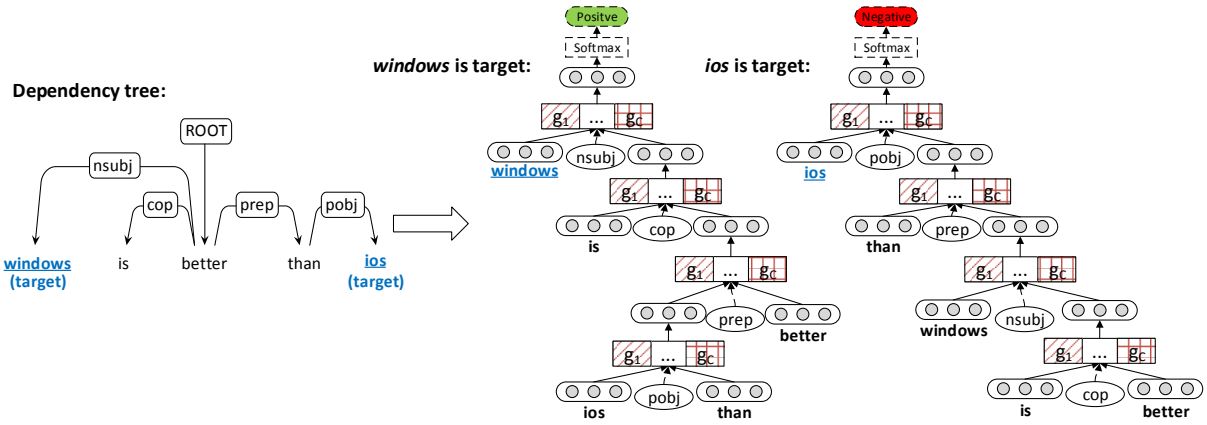

Figure 2: For the sentence "*windows is better than ios*", we convert its dependency tree for the different targets (*windows* and *ios*). AdaRNN performs semantic compositions in bottom-up manner and forward propagates sentiment information to the target node. The $g_1, \ldots, g_C$ are different composition functions, and the combined vectors and dependency types are used to select them adaptively. These composition functions decide how to propagate the sentiments to the target.

tweets. We randomly sample some tweets, and they are assigned with sentiment labels by two annotators. About 82.5% of them have the same labels. The agreement percentage of polarity classification is higher than subjectivity classification. To the best of our knowledge, this is the largest target-dependent Twitter sentiment classification dataset which is annotated manually. We make the dataset publicly available [1] for research purposes.

We preprocess the tweets by replacing the targets with *T* and setting their POS tags to *NN*. Liblinear (Fan et al., 2008) is used for baselines. A tweet-specific tokenizer (Gimpel et al., 2011) is employed, and the dependency parsing results are computed by Stanford Parser (Klein and Manning, 2003). The hyper-parameters are chosen by cross-validation on the training split, and the test accuracy and macro-average F1-score score are reported. For recursive neural models, the dimension of word vector is set to 25, and $f = \tanh$ is used as the nonlinearity function. We employ 10 composition matrices in AdaRNN. The parameters are randomly initialized. Notably, the word vectors will also be updated.

SVM-indep: It uses the uni-gram, bi-gram, punctuations, emoticons, and #hashtags as the content features, and the numbers of positive or negative words in General Inquirer as lexicon features. These features are all target-independent.

SVM-dep: We re-implement the method proposed by Jiang et al. (2011). It combines both the target-independent (SVM-indep) and target-dependent features and uses SVM as the classifier. There are seven rules to extract target-sensitive features. We do not implement the social graph optimization and target expansion tricks in it.

SVM-conn: The words, punctuations, emoticons, and #hashtags included in the converted dependency tree are used as the features for SVM.

RNN: It is performed on the converted dependency tree without adaptive composition selection.

AdaRNN-w/oE: Our approach without using the dependency types as features in adaptive selection for the composition functions.

AdaRNN-w/E: Our approach with employing the dependency types as features in adaptive selection for the composition functions.

AdaRNN-comb: We combine the root vectors obtained by AdaRNN-w/E with the uni/bi-gram features, and they are fed into a SVM classifier.

Method	Accuracy	Macro-F1
SVM-indep	62.7	60.2
SVM-dep	63.4	63.3
SVM-conn	60.0	59.6
RNN	63.0	62.8
AdaRNN-w/oE	64.9	64.4
AdaRNN-w/E	65.8	65.5
AdaRNN-comb	**66.3**	**65.9**

Table 1: Evaluation results on target-dependent Twitter sentiment classification dataset. Our approach outperforms the baseline methods.

[1] http://goo.gl/5Enpu7

As shown in the Table 1, AdaRNN achieves better results than the baselines. Specifically, we find that the performances of SVM-dep increase than SVM-indep. It indicates that target-dependent features help improve the results. However, the accuracy and F1-score do not gain significantly. This is caused by mismatch of the rules (Jiang et al., 2011) used to extract the target-dependent features. The POS tagging and dependency parsing results are not precise enough for the Twitter data, so these hand-crafted rules are rarely matched. Further, the results of SVM-conn illustrate that using the words which have paths to target as bag-of-words features does not perform well.

RNN is also based on the converted dependency tree. It outperforms SVM-indep, and is comparable with SVM-dep. The performances of AdaRNN-w/oE are better than the above baselines. It shows that multiple composition functions and adaptive selection help improve the results. AdaRNN provides more powerful composition ability, so that it achieves better semantic composition for recursive neural models. AdaRNN-w/E obtains best performances among the above methods. Its macro-average F1-score rises by 5.3% than the target-independent method SVM-indep. It employs dependency types as binary features to select the composition functions adaptively. The results illustrate that the syntactic tags are helpful to guide the model propagate sentiments of words towards target. Although the dependency results are also not precise enough, the composition selection is automatically learned from data. Hence AdaRNN is more robust for the imprecision of parsing results than the hand-crafted rules. The performances become better after adding the unigram and bi-gram features (target-independent).

4.1 Effects of β

We compare different β for AdaRNN defined in Equation (3) in this section. Different parameter β leads to different composition selection schemes.

As illustrated in Figure 3, the AdaRNN-w/oE and AdaRNN-w/E achieve the best accuracies at $\beta = 2$, and they have a similar trend. Specifically, $\beta = 0$ obtains a uniform distribution over the composition functions which does not help improve performances. $\beta \to \infty$ results in a maximum probability selection algorithm, i.e., only the composition function which has the maximum probability is used. This selection scheme makes the optimization instable. The performances of $\beta = 1, 2$ are similar and they are better than other settings. It indicates that adaptive selection method is useful to model the compositions. The hyper-parameter β makes trade-offs between uniform selection and maximum selection. It adjusts the effects of these two perspectives.

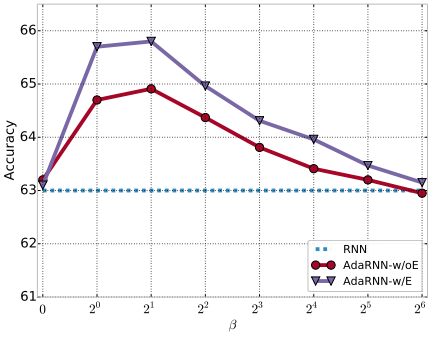

Figure 3: The curve shows the accuracy as the hyper-parameter $\beta = 0, 2^0, 2^1, \ldots, 2^6$ increases. AdaRNN achieves the best results at $\beta = 2^1$.

5 Conclusion

We propose Adaptive Recursive Neural Network (AdaRNN) for the target-dependent Twitter sentiment classification. AdaRNN employs more than one composition functions and adaptively chooses them depending on the context and linguistic tags. For a given tweet, we first convert its dependency tree for the interested target. Next, the AdaRNN learns how to adaptively propagate the sentiments of words to the target node. AdaRNN enables the sentiment propagations to be sensitive to both linguistic and semantic categories by using different compositions. The experimental results illustrate that AdaRNN improves the baselines without hand-crafted rules.

Acknowledgments

This research was partly supported by the National 863 Program of China (No. 2012AA011005), the fund of SKLSDE (Grant No. SKLSDE-2013ZX-06), and Research Fund for the Doctoral Program of Higher Education of China (Grant No. 20111102110019).

References

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Janvin. 2003. A neural probabilistic language model. *JMLR*, 3:1137–1155, March.

Li Dong, Furu Wei, Ming Zhou, and Ke Xu. 2014. Adaptive multi-compositionality for recursive neural models with applications to sentiment analysis. In *Twenty-Eighth AAAI Conference on Artificial Intelligence (AAAI)*. AAAI.

John Duchi, Elad Hazan, and Yoram Singer. 2011. Adaptive subgradient methods for online learning and stochastic optimization. *JMLR*, 12:2121–2159, July.

Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. 2008. Liblinear: A library for large linear classification. *J. Mach. Learn. Res.*, 9:1871–1874, June.

H.O. Georgii. 2011. *Gibbs Measures and Phase Transitions*. De Gruyter studies in mathematics. De Gruyter.

Kevin Gimpel, Nathan Schneider, Brendan O'Connor, Dipanjan Das, Daniel Mills, Jacob Eisenstein, Michael Heilman, Dani Yogatama, Jeffrey Flanigan, and Noah A. Smith. 2011. Part-of-speech tagging for twitter: Annotation, features, and experiments. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: Short Papers - Volume 2*, HLT '11, pages 42–47, Stroudsburg, PA, USA. Association for Computational Linguistics.

Geoffrey E. Hinton. 1986. Learning distributed representations of concepts. In *Proceedings of the Eighth Annual Conference of the Cognitive Science Society*, pages 1–12. Hillsdale, NJ: Erlbaum.

Minqing Hu and Bing Liu. 2004. Mining and summarizing customer reviews. In *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '04, pages 168–177, New York, NY, USA. ACM.

Long Jiang, Mo Yu, Ming Zhou, Xiaohua Liu, and Tiejun Zhao. 2011. Target-dependent twitter sentiment classification. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, HLT '11, pages 151–160, Stroudsburg, PA, USA. Association for Computational Linguistics.

Dan Klein and Christopher D. Manning. 2003. Accurate unlexicalized parsing. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics - Volume 1*, ACL '03, pages 423–430, Stroudsburg, PA, USA. Association for Computational Linguistics.

Tetsuji Nakagawa, Kentaro Inui, and Sadao Kurohashi. 2010. Dependency tree-based sentiment classification using crfs with hidden variables. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 786–794. Association for Computational Linguistics.

Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. 2002. Thumbs up?: sentiment classification using machine learning techniques. In *Proceedings of the ACL-02 conference on Empirical methods in natural language processing-Volume 10*, pages 79–86. Association for Computational Linguistics.

D.E. Rumelhart, G.E. Hinton, and R.J. Williams. 1986. Learning representations by back-propagating errors. *Nature*, 323(6088):533–536.

Richard Socher, Cliff C. Lin, Andrew Y. Ng, and Christopher D. Manning. 2011. Parsing Natural Scenes and Natural Language with Recursive Neural Networks. In *ICML*.

Richard Socher, Brody Huval, Christopher D. Manning, and Andrew Y. Ng. 2012. Semantic compositionality through recursive matrix-vector spaces. In *EMNLP-CoNLL*, pages 1201–1211.

Richard Socher, Alex Perelygin, Jean Y Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. 2013. Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank. In *EMNLP*, pages 1631–1642.

Maite Taboada, Julian Brooke, Milan Tofiloski, Kimberly Voll, and Manfred Stede. 2011. Lexicon-based methods for sentiment analysis. *Comput. Linguist.*, 37(2):267–307, June.