Neural Networks for Integrating Compositional and Non-compositional Sentiment in Sentiment Composition

Xiaodan Zhu & Hongyu Guo
National Research Council Canada
1200 Montreal Road, M50
Ottawa, ON K1A 0R6, Canada
{xiaodan.zhu,hongyu.guo}@nrc-cnrc.gc.ca

Parinaz Sobhani
EECS, University of Ottawa
800 King Edward Avenue
Ottawa, ON K1N 6N5, Canada
psobh090@uottawa.ca

Abstract

This paper proposes neural networks for integrating compositional and non-compositional sentiment in the process of *sentiment composition*, a type of semantic composition that optimizes a sentiment objective. We enable individual composition operations in a recursive process to possess the capability of choosing and merging information from these two types of sources. We propose our models in neural network frameworks with structures, in which the merging parameters can be learned in a principled way to optimize a well-defined objective. We conduct experiments on the Stanford Sentiment Treebank and show that the proposed models achieve better results over the model that lacks this ability.

1 Introduction

Automatically determining the sentiment of a phrase, a sentence, or even a longer piece of text is still a challenging problem. Data sparseness encountered in such tasks often requires to factorize the problem to consider smaller pieces of component words or phrases, for which much research has been performed on bag-of-words or bag-of-phrases models (Pang and Lee, 2008; Liu and Zhang, 2012). More recent work has started to model *sentiment composition* (Moilanen and Pulman, 2007; Choi and Cardie, 2008; Socher et al., 2012; Socher et al., 2013), a type of semantic composition that optimizes a sentiment objective. In general, the composition process is critical in the formation of the sentiment of a span of text, which has not been well modeled yet and there is still scope for future work.

Compositionality, or non-compositionality, of the senses of text spans is important for language understanding. Sentiment, as one of the major semantic differential categories (Osgood et al., 1957), faces the problem as well. For example, the phrase *must see* or *must try* in a movie or restaurant review often indicates a positive sentiment, which, however, may be hard to learn from the component words. More extreme examples, e.g., slangs like *bad ass*, are not rare in social media text. This particular example can actually convey a very positive sentiment even though its component words are very negative. In brief, a sentiment composition framework that can consider both compositional and non-compositional sentiment is theoretically interesting.

From a more pragmatical viewpoint, if one is able to reliably learn the sentiment of a text span (e.g., an ngram) holistically, it would be desirable that a composition model has the ability to decide the sources of knowledge it trusts more: the composition from the component words, the non-compositional source, or a *soft* combination of them. In such a situation, whether the text span is actually composable may be blur or may not be a concern.

In general, the composition of sentiment is a rather complicated process. As a glimpse of evidence, the effect of negation words on changing sentiment of their scopes appears to be a complicated function (Zhu et al., 2014). The recently proposed neural networks (Socher et al., 2013; Socher et al., 2011) are promising, for their capability of modeling complicated functions (Mitchell, 1997) in

general, handling data sparseness by learning low-dimensional embeddings at each layer of composition, and providing a framework to optimize the composition process in principled way.

This paper proposes neural networks for integrating compositional and non-compositional sentiment in the process of sentiment composition. To achieve this, we enable individual composition operations in a recursive process to possess the capability of choosing and merging information from these two types of sources. We propose our models in neural network frameworks with structures (Socher et al., 2013), in which the merging parameters can be learned in a principled way to optimize a well-defined objective. We conduct experiments on the Stanford Sentiment Treebank and show that the proposed models achieve better results over the model that does not consider this property.

2 Related work

Composition of sentiment Early work on modeling sentiment does not examine semantic composition closely (Pang and Lee, 2008; Liu and Zhang, 2012), as mentioned above. Recent work has considered sentiment-oriented semantic composition (Moilanen and Pulman, 2007; Choi and Cardie, 2008; Socher et al., 2012; Socher et al., 2013), or simply called sentiment composition in this paper. For example, Moilanen and Pulman (2007) used a collection of handwritten compositional rules to assign sentiment values to different granularities of text spans. Choi and Cardie (2008) proposed a learning-based framework. The more recent work of (Socher et al., 2013) proposed models based on neural networks that do not rely on any heuristic rules. Such models work in a bottom-up fashion over a tree to infer the sentiment label of a phrase or sentence as a composition of the sentiment expressed by its constituting parts. The approach leverages a principled method, the forward and backward propagation, to optimize the system performance. In this paper, we follow the neural network approach to integrate compositional and non-compositional sentiment in sentiment composition.

Prior knowledge of sentiment Integrating non-compositional sentiment into the composition process can be viewed as introducing some prior sentiment knowledge, as in general the sentiment of a word or a phrase perceived independent of its context is often referred to as *prior* sentiment. Word-level prior sentiment is typically annotated in manual sentiment lexicons (Wilson et al., 2005; Hu and Liu, 2004; Mohammad and Turney, 2010), or learned in an unsupervised or semisupervised way (Hatzivassiloglou and McKeown, 1997; Esuli and Sebastiani, 2006; Turney and Littman, 2003; Mohammad et al., 2009). More recently, sentiment indicators, such as emoticons and hashtags, are utilized (Go et al., 2009; Davidov et al., 2010; Kouloumpis et al., 2011; Mohammad, 2012; Mohammad et al., 2013a). With enough data, such freely available (but noisy) annotation can be used to learn the sentiment of ngrams. In our study, we will investigate in the proposed composition models the effect of automatically learned sentimental ngrams.

3 Prior-enriched semantic networks

In this paper, we propose several neural networks that enable each composition operation to possess the ability of choosing and merging sentiment from lower-level composition and that from non-compositional sources. We call the networks Prior-Enriched Semantic Networks (PESN). We present several specific implementations based on RNTN (Socher et al., 2013); the latter has showed to be a state-of-the-art sentiment composition framework. However, the realization of a PESN node is not necessarily only tied with RNTN.

Figure 1 shows a piece of PESN. Each of the three big nodes, i.e., N_1, N_2, and N_3, corresponds to a node in a constituency parse tree; e.g., N_3 may correspond to the phrase *not a must try*, where N_1 and N_2 are *not* and *a must try*, respectively. We extend each of the nodes to possess the ability to consider sentiment from lower-level composition and non-compositional sources. In node N_3, knowledge from the lower-level composition is represented in the hidden vector i_3, which is merged with non-compositional knowledge represented in e_3, and the merged information is saved in m_3. The black box in the center performs the actual merging, which integrates the two knowledge sources in order to min-

imize an overall objective function that we will discuss in detail later. The recursive neural networks and the forward-backward propagation over structures (Socher et al., 2013; Goller and Kchler, 1996) provide a principled way to optimize the whole network.

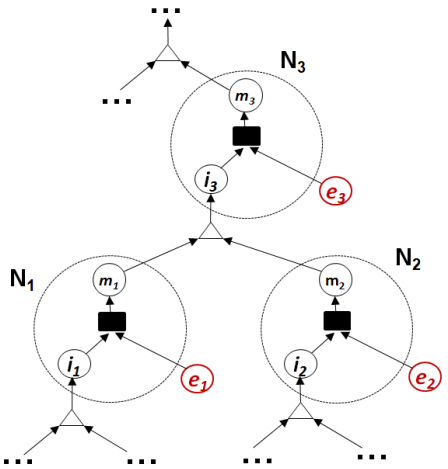

Figure 1: A prior-enriched semantic network (PESN) for sentiment composition. The three nodes, N_1, N_2, and N_3, correspond to three nodes in a constituency parse tree, and each of them consider sentiment from lower-level composition (i_1, i_2, i_3) and from non-compositional sentiment (e_1, e_2, e_3).

3.1 Regular bilinear merging

The most straightforward way of implementing a PESN node is probably through a regular bilinear merging. Take node N_3 in Figure 1 as an example; the node vector m_3 will be simply merged from i_3 and e_3 as follows:

$$m_3 = tanh(W_m \begin{bmatrix} i_3 \\ e_3 \end{bmatrix} + b_m) \quad (1)$$

Again, vector i_3 contains the knowledge from the lower-level composition; e_3 is a vector representing non-compositional sentiment information, which can be either from human annotation or automatically learned resources. Note that in the network, all hidden vectors m and i (including word embedding vectors) have the same dimensionality d, but the non-compositional nodes, i.e., the nodes e , do not necessarily have to have the same number of elements, and we let l be their dimensionality. The merging matrix W_m is *d-by-(d+l)*.

As in this paper we discuss PESN in the framework of RNTN, computation outside the nodes N_1, N_2, N_3 follows that for the standard three-way tensors in RNTN. That is, the hidden vector i_3 is computed with the following formula:

$$i_3 = tanh(\begin{bmatrix} m_1 \\ m_2 \end{bmatrix}^T V_r^{[1:d]} \begin{bmatrix} m_1 \\ m_2 \end{bmatrix} + W_r \begin{bmatrix} m_1 \\ m_2 \end{bmatrix}) \quad (2)$$

where, $W_r \in \mathbb{R}^{d \times (d+d)}$ and $V_r \in \mathbb{R}^{(d+d) \times (d+d) \times d}$ are the matrix and tensor of the composition function used in RNTN, respectively, each of which is shared over the whole tree in computing vectors i_1, i_2, and i_3.

3.2 Explicitly gated merging

Compared to the regular bilinear merging model, we here further explicitly control the input of the compositional and non-compositional semantics. Explicitly gating neural network has been studied in the literature. For example, the long short-term memory (LSTM) utilizes *input gates*, together with *output gates* and *forget gates*, to guide memory blocks to remember/forget history (Hochreiter and Schmidhuber, 1997).

For our purpose here, we explore an *input gate* to explicitly control the two different input sources. As shown in Figure 2, an additional gating layer g_3 is used to control i_3, e_3 explicitly.

$$g_3 = \sigma(\begin{bmatrix} W_{g_e} e_3 \\ W_{g_i} i_3 \end{bmatrix} + b_g) \quad (3)$$

$$m_3 = tanh(W_m(g_3 \otimes \begin{bmatrix} i_3 \\ e_3 \end{bmatrix}) + b_m) \quad (4)$$

The sign $\otimes$ is a Hadamard product; σ is a logistic sigmoid function instead of a $tanh$ activation, which makes the gating signal g_3 to be in the range of [0, 1] and serve as a soft switch (not a hard binary

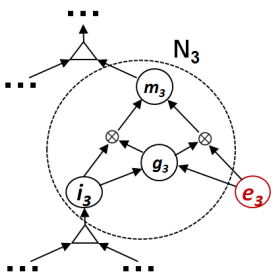

Figure 2: An input-gated network that explicitly controls the compositional and non-compositional sentiment input.

0/1 switch) to explicitly gate i_3 and e_3. Note that elsewhere in the network, we still use $tanh$ as our activation function. In addition, $W_{g_e} \in \mathbb{R}^{d \times l}$ and $W_{g_i} \in \mathbb{R}^{l \times d}$ are the weight matrices used to calculate the gate vector.

3.3 Confined-tensor-based merging

The third approach we use for merging compositional and non-compositional knowledge employs tensors, which are able to explore multiplicative combination among variables. Tensors have already been successfully used in a wide range of NLP tasks in capturing high-order interactions among variables. The forward computation of m_3 follows:

$$m_3 = tanh(\begin{bmatrix} i_3 \\ e_3 \end{bmatrix}^T V_m^{[1:d]} \begin{bmatrix} i_3 \\ e_3 \end{bmatrix} + W_m \begin{bmatrix} i_3 \\ e_3 \end{bmatrix}) \quad (5)$$

where $V_m^{[1:d]} \in R^{(d+l)\times(d+l)\times d}$ is the tensor m that defines multiple bilinear forms, and the matrix W_m is as defined in the previous models.

As we focus on the interaction between i_3 and e_3, we force each slice of tensor, e.g. $V_m^{[k]}$, to have zero-valued blocks. More specifically, the top-right *d-by-l* block of the piece matrix $V_m^{[k]}$ ($k \in \{1...d\}$) and the bottom-left *l-by-d* block are non-zero parameters, used to capture multiplicative, element-pair interactions between i_3 and e_3, while the rest block are set to be zero, to ignore interactions between those variables within i_3 and those within e_3. This does not only make the model focus on the interaction between vector i and e, it also helps significantly reduce the number of parameters to estimate, which, otherwise, could potentially lead to overfitting. We call this model confined-tensor-based merging.

3.4 Learning and inference

Objective The overall objective function in learning PESN, following (Socher et al., 2013), minimizes the cross-entropy error between the predicted distribution $y^{sen_i} \in \mathbb{R}^{c\times 1}$ at a node i and the target distribution $t^i \in \mathbb{R}^{c\times 1}$ at that node, where c is the number of sentiment categories. PESN learns the parameters that are used to merge the compositional and non-compositional sentiment so that the merging operations integrate the two sources in minimizing prediction loss. The neural network over structures provides a principled framework to optimize these parameters.

More specifically, the error over an entire sentence is calculated as a regularized sum:

$$E(\theta) = \sum_i \sum_j t_j^i \log y^{sen_i}{}_j + \lambda \left\|\theta\right\|^2 \quad (6)$$

where, λ is the regularization parameter, $j \in c$ denotes the j-th element of the multinomial target distribution, θ are model parameters that will be discussed below, and i iterates over all nodes i_x (e.g., i_1, i_2, and i_3) in Figure 1, where the model predicts sentiment labels.

Backpropagation over the structures To minimize $E(\theta)$, the gradient of the objective function with respect to each of the parameters in θ is calculated efficiently via backpropagation through structure (Socher et al., 2013; Goller and Kchler, 1996), after computing the prediction errors in forward propagation with formulas described above.

Regular bilinear merging The PESN implemented with simple bilinear merging has the following model parameters: $\theta = (V_r, W_r, W_m, W_{label}, L)$. As discussed above, V_r and W_r are the tensor and matrix in RNTN; W_m is the weight matrix for merging the compositional and non-compositional sentiment vectors. L denotes the vector representations of the word dictionary, and W_{label} is sentiment classification matrix used to predict sentiment label at a

node. Backpropagation on the regular bilinear merging node follows a standard derivative computation in a regular feed-forward network, which we skip here.

Explicitly gated merging In this model, in addition to W_m, we further learn two weight matrices W_{g_i} and W_{g_e}, as introduced in Formula 3 and 4 above. Consider Figure 2 and let δ^{m_3} denote the error messages passed down to node m_3. The error messages are passed back to i_3 directly through the Hadamard product and also through the gate node g_3. The former, denoted as $\delta^{i_3,dir}$, is calculated with:

$$\delta^{i_3,dir} = (\delta^{m_3} \otimes g_3)[1:d] \tag{7}$$

where, g_3 is calculated with Formula 3 above in the forward process; $[1:d]$ means taking the first d elements of the vector yielded by the Hadamard product; the rest $[d+1:d+l]$ elements of the Hadamard production are discarded, as we do not update e_3, which is given as our prior knowledge.

The error messages passed down to gate vector g_3 is computed with

$$\delta^{g_3} = \delta^{m_3} \otimes \begin{bmatrix} i_3 \\ e_3 \end{bmatrix} \otimes s^{'}(g_3) \tag{8}$$

where, $s^{'}(.)$ is the element-wise derivative of logistic function, which can be calculated only using $s(.)$, as $s(.)(1-s(.))$. The derivative of W_{g_i} can be calculated with:

$$\frac{\partial E^{g_3}}{W_{g_e}} = (\delta^{g_3}[1:d])e_3^T \tag{9}$$

Similarly, partial derivatives over W_{g_i} can be calculated. These values will be summed to the total derivative of W_{g_i} and W_{g_e}, respectively. With these notations, the error messages passed down to i_3 through the gate can then be computed with:

$$\delta^{i_3,gate} = W_{g_i}^T(\delta^{g_3}[d+1:d+l]) \tag{10}$$

and the total error messages to node i_3 is then:

$$\delta^{i_3,total} = (\delta^{i_3,dir} + \delta^{i_3,gate} + \delta^{i_3,local}) \otimes f^{'}(i_3) \tag{11}$$

where $\delta^{i_3,local}$ is the local error message from the sentiment prediction errors performed at the node i_3 itself to obtain the total error message for i_3, which is in turn passed down through regular RNTN tensor to the lower levels. $f^{'}(.)$ is the element-wise derivative of $tanh$ function.

Confined-tensor-based merging In confined-tensor-based merging, the error messages passed to the two children i_3 and e_3 is computed with:

$$\delta^{i_3,e_3} = (W_m^T\delta^{m_3}) \otimes f^{'}(\begin{bmatrix} i_3 \\ e_3 \end{bmatrix}) + \delta^{tns} \tag{12}$$

where,

$$\delta^{tns} = \sum_{k=1}^{d} \delta_k^{m_3}(V_m^{[k]} + (V_m^{[k]})^T) \begin{bmatrix} i_3 \\ e_3 \end{bmatrix} \otimes f^{'}(\begin{bmatrix} i_3 \\ e_3 \end{bmatrix}) \tag{13}$$

where the error messages to i_3 are the first d numbers of elements of δ^{i_3,e_3}. The rest elements of δ^{i_3,e_3} are discarded; as mentioned above, we do not update e_3 as it is given as the prior knowledge. We skip the derivative for the W_{m_3}. While the derivative of each slice $k(k = 1,\ldots,d)$ of the tensor V is calculated with:

$$\frac{\partial E^{m_3}}{V_m^{[k]}} = \delta_k^{m_3,down} \begin{bmatrix} i_3 \\ e_3 \end{bmatrix} \begin{bmatrix} i_3 \\ e_3 \end{bmatrix}^T \tag{14}$$

Again, the full derivative for V_m and W_m is the sum of their derivatives over the trees. After the error message passing from m_3 to i_3 is obtained, it can be summed up with the local error message from the sentiment prediction errors at the node i_3 itself to obtain the total error message for i_3, which is in turn used to calculate the error messages passed down as well as the derivative in the lower-level tree.

4 Experiments

4.1 Data

We use the Stanford Sentiment Treebank (Socher et al., 2013) in our experiments. The data contain about 11,800 sentences from the movie reviews that were originally collected by Pang and Lee (2005).

The sentences were parsed with the Stanford parser (Klein and Manning, 2003). Phrases at all the tree nodes were manually annotated with sentiment values. We use the same split of the training and test data as in (Socher et al., 2013) to predict the sentiment categories of the roots (sentences) and the phrases, and use the same evaluation metric, *classification accuracy*, to measure the performances.

4.2 Obtaining non-compositional sentiment

In our experiments, we explore in sentiment composition the effect of two different types of non-compositional sentiment: (1) sentiment of ngrams automatically learned from an external, much larger corpus, and (2) sentiment of ngrams assigned by human annotators.

Following the method proposed in (Mohammad et al., 2013b), we learn sentimental ngrams from Tweets. The unsupervised approach utilizes *hashtags*, which can be regarded as conveying freely available (but noisy) human annotation of sentiment. More specifically, certain words in tweets are specially marked with the hash character (#) to indicate the topic, sentiment polarity, or emotions such as joy, sadness, angry, and surprised. With enough data, such artificial annotation can be used to learn the sentiment of ngrams by their likelihood of co-occurring with such hashtagged words.

More specifically, a collection of 78 seed hashtags closely related to *positive* and *negative* such as *#good, #excellent, #bad,* and *#terrible* were used (32 positive and 36 negative). These terms were chosen from entries for *positive* and *negative* in the Roget's Thesaurus. A set of 775,000 tweets that contain at least a positive hashtag or a negative hashtag were used as the learning corpus. A tweet was considered positive if it had one of the 32 positive seed hashtags, and negative if it had one of the 36 negative seed hashtags. The association score for an ngram w was calculated from these pseudo-labeled tweets as follows:

$$score(w) = PMI(w, positive) - PMI(w, negative) \quad (15)$$

where PMI stands for pointwise mutual information, and the two terms in the formula calculate the PMI between the target ngram and the pseudo-labeled positive tweets as well as that between the ngram and the negative tweets, respectively. Accordingly, a positive *score(.)* indicates association with positive sentiment, whereas a negative score indicates association with negative sentiment.

We use in our experiments the bigrams and trigrams learned from the dataset with the occurrences higher than 5. We assign these ngrams into one of the 5 bins according to their sentiment scores obtained with Formula 15: $(-\infty, -2]$, $(-2, -1]$, $(-1, 1)$, $[1, 2)$, and $[2, +\infty)$. Each ngram is now given a one-hot vector, indicating the polarity and strength of its sentiment. For example, a bigram with a score of -1.5 will be assigned a 5-dimensional vector $[0, 1, 0, 0, 0]$, indicating a weak negative. Note that PESN can also take into other forms of sentiment embeddings, such as those learned in (Tang et al., 2014).

In addition, the Stanford Sentiment Treebank contains manually annotated sentiment for each individual phrase in a parse tree, so we use such annotation but not other manual lexicons, by assuming such annotation fits the corpus itself the best. Specifically, we use bigram and trigram annotation in the treebank. Note that even longer ngrams are much sparser and probably less useful in general, one may learn sentiment for multi-word expressions of a larger length, which we will leave as future work.

4.3 Results

Overall prediction performance Table 1 shows the accuracies of different models on Stanford Sentiment Treebank. We evaluate the models on 5-category sentiment prediction at both the sentence (root) level and at all nodes (including roots).[1] The results reported in Table 1 are all based on the version 3.3.0 of the Stanford CoreNLP[2] and our implementation of PESN on it. The CoreNLP includes a java implementation of RNTN.[3] To make the results reported in the table comparable, we trained the

[1]The package only gives approximate accuracies for 2-category sentiment, which are not included here in the table.

[2]http://nlp.stanford.edu/sentiment/code.html

[3]The matlab code used in (Socher et al., 2013) is not published.

Models	sentence-level (roots)	all phrases (all nodes)
(1) RNTN	42.44	79.95
(2) Regular-bilinear (auto)	42.37	79.97
(3) Regular-bilinear (manu)	42.98	80.14
(4) Explicitly-gated (auto)	42.58	80.06
(5) Explicitly-gated (manu)	43.21	80.21
(6) Confined-tensor (auto)	42.99	80.49
(7) Confined-tensor (manu)	**43.75**†	**80.66**†

Table 1: Model performances (accuracies) on predicting 5-category sentiment at the sentence (root) level and phrase-level on Stanford Sentiment Treebank. The numbers in the bold font are the best performances achieved on the two tasks. Both results are statistically significantly better ($p < 0.05$) than the corresponding RNTN results.

RNTN models with the default parameter[4] and run the training from 5 different random initializations, and report the best results we observed.

The rows in the table marked with *auto* are models using the automatically learned ngrams, and those marked with *manu* using manually annotated sentiment for bigrams and trigrams. Note that the non-compositional sentiment of a node is only used to predict the sentiment of phrases above it in the tree. For example, in Figure 1 discussed earlier, the effect of e_1 and e_2 will be used to predict the sentiment of i_3 and other node i above, but not that of i_1 and i_2 themselves, avoiding the concern of using the annotation of a tree node to predict the sentiment of itself.

The models in general benefit from incorporating the non-compositional knowledge. The numbers in the bold font are the best performance achieved on the two tasks. While using the simple regular bilinear merging shows some gains, the more complicated models achieve further improvement.

Above we have seen the general performance of the models. Below, we take a closer look at the prediction errors at different depths of the sentiment treebank. The *depth* here is defined as the longest distance between a tree node and its descendant leafs. In Figure 3, the x-axis corresponds to different depths and y-axis is the accuracy. The figure was drawn with the RNTN and the model (7) in Table 1, so as to study the compositional property in the ideal situation where the lexical has a full coverage of bigrams and trigrams.

[4]java -mx8g edu.stanford.nlp.sentiment.SentimentTraining -numHid 25 -trainPath train.txt -devPath dev.txt -train -model model.ser.gz

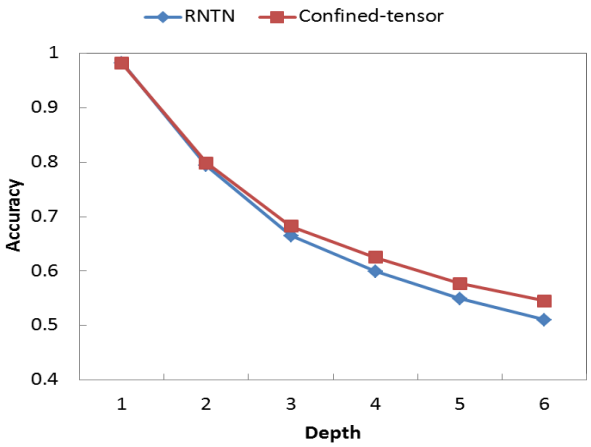

Figure 3: Errors made at different depths in the sentiment tree bank.

The figure shows that using the confined tensor to combine holistic sentiment information outperforms the original RNTN model that does not consider this, starting from depth 3, showing the benefit of using holistic bigram sentiment. The improvement increases at depth 4 (indicating the benefit of using trigram sentiment), and then was propagated to the higher levels of the tree. As discussed above, we only use non-compositional sentiment of a node to predict the sentiment of the phrases above it in the tree but not the node itself. And the system still needs to balance which source it trusts more, by optimizing the overall objective.

Although the empirical improvement may depend on the percentage of non-compositional instances in a data set or the sentiment that need to be learned holistically, we present here the first effort, according to our knowledge, on studying the concern of in-

tegrating compositional and non-compositional sentiment in the semantic composition process.

5 Conclusions and future work

This paper proposes models for integrating compositional and non-compositional sentiment in the process of sentiment composition. To achieve this, we enable each composition operation to be able to choose and merge information from these two types of sources. We propose to implement such models within neural network frameworks with structures (Socher et al., 2013), in which the merging parameters can be optimized in a principled way, to minimize a well-defined objective. We conduct experiments on the Stanford Sentiment Treebank and show that the proposed models achieve better results over the model that does not consider this property.

Although the empirical improvement may depend on the percentage of non-compositional instances in a data set or the sentiment that need to be learned holistically, we present here the first effort, according to our knowledge, on studying the basic concern of integrating compositional and non-compositional sentiment in composition. While we focus on sentiment in this paper, investigating compositional and non-compositional semantics for general semantic composition with neural networks is interesting to us as an immediate future problem, as such models provide a principled way to optimize the overall objective over the sentence structures when we consider both compositional and non-compositional semantics.

References

Yejin Choi and Claire Cardie. 2008. Learning with compositional semantics as structural inference for subsentential sentiment analysis. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '08, pages 793–801, Honolulu, Hawaii.

Dmitry Davidov, Oren Tsur, and Ari Rappoport. 2010. Enhanced sentiment learning using Twitter hashtags and smileys. In *Proceedings of the 23rd International Conference on Computational Linguistics: Posters*, COLING '10, pages 241–249, Beijing, China.

Andrea Esuli and Fabrizio Sebastiani. 2006. SENTIWORDNET: A publicly available lexical resource for opinion mining. In *In Proceedings of the 5th Conference on Language Resources and Evaluation*, LREC '06, pages 417–422.

Alec Go, Richa Bhayani, and Lei Huang. 2009. Twitter sentiment classification using distant supervision. Technical report, Stanford University.

Christoph Goller and Andreas Kchler. 1996. Learning task-dependent distributed representations by backpropagation through structure. In *In Proc. of the ICNN-96*, pages 347–352, Bochum, Germany. IEEE.

Vasileios Hatzivassiloglou and Kathleen R. McKeown. 1997. Predicting the semantic orientation of adjectives. In *Proceedings of the 8th Conference of European Chapter of the Association for Computational Linguistics*, EACL '97, pages 174–181, Madrid, Spain.

S. Hochreiter and J. Schmidhuber. 1997. Long short-term memory. *Neural Computation*, 9(8):1735–1780.

Minqing Hu and Bing Liu. 2004. Mining and summarizing customer reviews. In *Proceedings of the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '04, pages 168–177, New York, NY, USA. ACM.

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, Sapporo, Japan. Association for Computational Linguistics.

Efthymios Kouloumpis, Theresa Wilson, and Johanna Moore. 2011. Twitter sentiment analysis: The Good the Bad and the OMG! In *Proceedings of the 5th International AAAI Conference on Weblogs and Social Media*.

Bing Liu and Lei Zhang. 2012. A survey of opinion mining and sentiment analysis. In Charu C. Aggarwal and ChengXiang Zhai, editors, *Mining Text Data*, pages 415–463. Springer US.

Tom M Mitchell. 1997. Machine learning. 1997. *Burr Ridge, IL: McGraw Hill*, 45.

Saif M. Mohammad and Peter D. Turney. 2010. Emotions evoked by common words and phrases: Using Mechanical Turk to create an emotion lexicon. In *Proceedings of the NAACL-HLT Workshop on Computational Approaches to Analysis and Generation of Emotion in Text*, LA, California.

Saif Mohammad, Cody Dunne, and Bonnie Dorr. 2009. Generating high-coverage semantic orientation lexicons from overtly marked words and a thesaurus. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing: Volume 2*, EMNLP '09, pages 599–608, Singapore.

S. Mohammad, S. Kiritchenko, and X. Zhu. 2013a. NRC-Canada: Building the state-of-the-art in sentiment analysis of tweets. In *Proceedings of the International Workshop on Semantic Evaluation*, SemEval '13, Atlanta, Georgia, USA, June.

Saif M. Mohammad, Bonnie J. Dorr, Graeme Hirst, and Peter D. Turney. 2013b. Computing lexical contrast. *Computational Linguistics*, 39(3):555–590.

Saif Mohammad. 2012. #emotional tweets. In *Proceedings of the First Joint Conference on Lexical and Computational Semantics*, *SEM '12, pages 246–255, Montréal, Canada. Association for Computational Linguistics.

Karo Moilanen and Stephen Pulman. 2007. Sentiment composition. In *Proceedings of RANLP 2007*, Borovets, Bulgaria.

Charles E Osgood, George J Suci, and Percy Tannenbaum. 1957. *The measurement of meaning*. University of Illinois Press.

Bo Pang and Lillian Lee. 2005. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, ACL '05, pages 115–124.

Bo Pang and Lillian Lee. 2008. Opinion mining and sentiment analysis. *Foundations and Trends in Information Retrieval*, 2(1–2):1–135.

Richard Socher, Jeffrey Pennington, Eric Huang, Andrew Y. Ng, and Christopher D. Manning. 2011. Semi-supervised recursive autoencoders for predicting sentiment distributions. In *Conference on Empirical Methods in Natural Language Processing*.

Richard Socher, Brody Huval, Christopher D. Manning, and Andrew Y. Ng. 2012. Semantic compositionality through recursive matrix-vector spaces. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '12, Jeju, Korea. Association for Computational Linguistics.

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 *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '13, Seattle, USA. Association for Computational Linguistics.

Duyu Tang, Furu Wei, Nan Yang, Ming Zhou, Ting Liu, and Bing Qin. 2014. Learning sentiment-specific word embedding for twitter sentiment classification. In *Proceedings of ACL*, Baltimore, Maryland, USA, June.

Peter Turney and Michael L Littman. 2003. Measuring praise and criticism: Inference of semantic orientation from association. *ACM Transactions on Information Systems*, 21(4).

Theresa Wilson, Janyce Wiebe, and Paul Hoffmann. 2005. Recognizing contextual polarity in phrase-level sentiment analysis. In *Proceedings of the Conference on Human Language Technology and Empirical Methods in Natural Language Processing*, HLT '05, pages 347–354, Stroudsburg, PA, USA. Association for Computational Linguistics.

Xiaodan Zhu, Hongyu Guo, Saif Mohammad, and Svetlana Kiritchenko. 2014. An empirical study on the effect of negation words on sentiment. In *Proceedings of ACL*, Baltimore, Maryland, USA, June.