Gated Neural Networks for Targeted Sentiment Analysis

Meishan Zhang[1,2*] and **Yue Zhang**[2*] and **Duy-Tin Vo**[2]

1. School of Computer Science and Technology, Heilongjiang University, Harbin, China
2. Singapore University of Technology and Design
{meishan_zhang, yue_zhang}@sutd.edu.sg,
duytin_vo@mymail.sutd.edu.sg

Abstract

Targeted sentiment analysis classifies the sentiment polarity towards each target entity mention in given text documents. Seminal methods extract manual discrete features from automatic syntactic parse trees in order to capture semantic information of the enclosing sentence with respect to a target entity mention. Recently, it has been shown that competitive accuracies can be achieved without using syntactic parsers, which can be highly inaccurate on noisy text such as tweets. This is achieved by applying distributed word representations and rich neural pooling functions over a simple and intuitive segmentation of tweets according to target entity mentions. In this paper, we extend this idea by proposing a sentence-level neural model to address the limitation of pooling functions, which do not explicitly model tweet-level semantics. First, a bi-directional gated neural network is used to connect the words in a tweet so that pooling functions can be applied over the hidden layer instead of words for better representing the target and its contexts. Second, a three-way gated neural network structure is used to model the interaction between the target mention and its surrounding contexts. Experiments show that our proposed model gives significantly higher accuracies compared to the current best method for targeted sentiment analysis.

Introduction

Targeted sentiment analysis investigates the classification of opinion polarities towards certain target entity mentions in given sentences (Jiang et al. 2011; Dong et al. 2014; Vo and Zhang 2015). An example is shown in Figure 1, where the inputs are a set of tweets, with the entities "miley ray cyrus", "taylor swift", "Windows 7", "nick cannon" and "britney spears" being marked as the opinion targets, respectively. The outputs consist of three way (+, -, 0) sentiment labels on each entity.

A very important issue in targeted sentiment analysis is how to model the association between targets and their contexts, which can involve complex syntactic and semantic

---

She began to love [**miley ray cyrus**]$_+$ since 2013 :)
Some chocolate a tup of ice cream and [**taylor swift**]$_+$ songs. Best remedy to help cure the pain.
Does Vmware fusion support [**Windows 7**]$_0$ yet?
[**nick cannon**]$_-$ face is annoying!!!!!!
I have no interest in seeing [**britney spears**]$_-$ do anything.

Figure 1: *Targeted sentiment analysis.*

structures such as predicate-argument links, negations, coreferences and even sarcasm. Jiang et al. (2011) define rich features over POS tags and dependency links of a given tweet as well as its context tweets, in order to capture target sentiment polarities. Dong et al. (2014) use word embeddings and neural networks instead of manual features, but also rely on syntactic dependencies, which are produced by automatic POS-tagging and parsing.

Compared with discrete manual features, embedding features are less sparse, and can be learned from large raw texts, capturing distributional syntactic and semantic information automatically. As a result, they offer a potentially more elegant and effective solution to complex semantic problems such as negation and sarcasm. Vo and Zhang (2015) exploit this advantage by defining rich automatic features via neural pooling over multifarious word embeddings. Shown in Figure 2(a), given a target, they divide the enclosing sentence into three segments, using pooling functions on each segment to obtain features for the left context, the target and the right context, respectively. The automatic features are then fed into a linear classifier for sentiment classification.

The method of Vo and Zhang (2015) avoids the reliance on automatic syntactic parsers, which can be highly inaccurate on tweets, thereby leading to improved sentiment classification results compared with Dong et al. (2014). On the other hand, it does not fully capture semantic information of the tweet and the given target entity. First, pooling functions can select the most useful features from a sequence of words, but do not capture underlying tweet-level syntactic and semantic information. Second, the interaction between the target and its contexts is modeled only implicitly over the pooling functions and the linear classifier, but not explicitly.

We address the above two limitations by using gated

neural network structures to model the syntax and semantics of the enclosing tweet, and the interaction between the surrounding contexts and the target, respectively. Recurrent neural networks (RNN) have been shown effective in modeling sentences (Mikolov et al. 2010; Graves 2012; Cho et al. 2014b; Sutskever, Vinyals, and Le 2014), capturing long-range dependencies between words. Gated recurrent neural networks (GRNN) (Cho et al. 2014a) and long-short-term-memories (LSTM) (Hochreiter and Schmidhuber 1997) facilitate the training of recurrent networks by solving the diminishing and exploding gradient problem in the deep structure (Bengio, Simard, and Frasconi 1994). For efficiency consideration, we use a simple gated recurrent neural network, which has been shown to give comparable accuracies with LSTMs for several tasks (Chung et al. 2014).

In addition, we explicitly model the interaction between the left context, the right context and the target, using a gated neural network layer. For some tweets, such as "I love [**Facebook**].", the left context (i.e. "I love") indicates the sentiment class of the target. For tweets such as "I think [**Transformers**] is boring.", the right context dominates the polarity. For tweets such as "Defeating [**Manchester United**] is challenging.", both contexts interact to express the sentiment polarity of the target. Accordingly, we design a gated neural network layer between the left and the right context (Figure 4), which explicitly interpolates the left context, the right context and a combination of both.

The effect of contexts on target sentiment also depends on the target entity itself. For example, the sentiment polarity of "[**X**] dropped" is "[**X**]$_+$" if X is "cost", but "[**X**]$_-$" if X is "revenue". Correspondingly, we add the target to the control gates in the above neural networks. Experiments show that the gated neural network structures lead to significantly improved sentiment classification accuracies compared with the method of Vo and Zhang (2015). We make our system and source code public under GPL at *https://github.com/SUTDNLP/NNTargetedSentiment*.

Related Work

Targeted sentiment analysis is related to fine-grained sentiment analysis (Wiebe, Wilson, and Cardie 2005; Jin, Ho, and Srihari 2009; Li et al. 2010; Yang and Cardie 2013; Nakov et al. 2013), which extracts opinion expressions, holders and targets jointly from given sentences. Compared with fine-grained sentiment, targeted sentiment offers less operational details, but on the other hand requires less manual annotation. There has also been work on open domain targeted sentiment (Mitchell et al. 2013; Zhang, Zhang, and Vo 2015), which identifies both the opinion targets and their sentiments. The task can be regarded as a joint problem of entity recognition and targeted sentiment classification.

Other related tasks include aspect-oriented sentiment analysis (Hu and Liu 2004; Popescu and Etzioni 2007), which extracts product features and opinions towards them from user reviews, and topic-oriented sentiment analysis (Yi et al. 2003; Wang et al. 2011), which extracts features and/or sentiments towards certain topics or subjects. These tasks are close in spirit to targeted sentiment analysis, with subtle variations on the domain and task formulation.
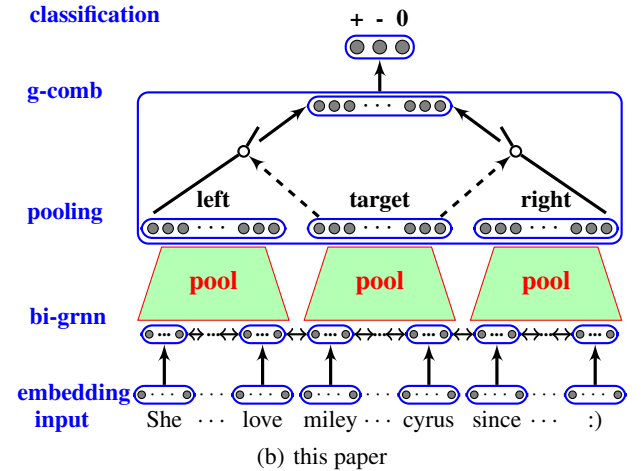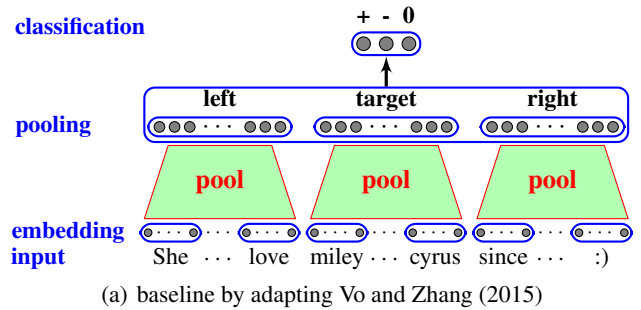

Figure 2: Baseline model and the model of this paper (She began to love miley ray cyrus since 2013 :)).

Traditional sentiment analysis systems rely on manual features (Pang, Lee, and Vaithyanathan 2002; Go, Bhayani, and Huang 2009; Mohammad, Kiritchenko, and Zhu 2013). Recently, distributed word representations (Socher et al. 2013; Tang et al. 2014; Vo and Zhang 2015) and deep neural network structures (Irsoy and Cardie 2013; Paulus, Socher, and Manning 2014; Kalchbrenner, Grefenstette, and Blunsom 2014; Zhou et al. 2014; Dong et al. 2014; dos Santos and Gatti 2014) have been used for the task, giving competitive accuracies. Our work is in line with these methods, using word embeddings and a deep neural network structure to automatically exploit the syntactic and semantic structures of tweets. However, we model targeted sentiment instead of document-level sentiment, which most prior work does. To our knowledge, we are the first to use deep neural networks to model sentences with respect to targeted sentiments.

Baseline

We take the model of Vo and Zhang (2015) as our baseline. Shown in Figure 2(a), it takes input tweets with target entities, and outputs the sentiment polarity of each target. For each given target, the model computes the sentiment class in three steps.

First, each word in a given tweet is mapped to a low-dimensional, real-valued embedding vector. Second, element-wise pooling functions are applied to extract use-

ful features from the target entity, the left context and the right context, respectively. We follow Vo and Zhang (2015), using the concatenation of *max*, *min*, *average*, *product*, and *standard deviation* pooling results for automatic features. Finally, all the features resulted from the pooling functions are concatenated and used as input to a linear classifier, which predicts the sentiment polarity of the target.

Our baseline is a much simplified version of Vo and Zhang (2015), who applied pooling functions over the full tweet in addition to the target entity, the left and the right contexts, in order to extract global features over a tweet. However, we do not include such features because our extended models explicitly capture global syntactic and semantic information over full tweets. Another source of information that Vo and Zhang (2015) exploit is sentiment lexicons. To separately study the influence of network structures, we do not include lexicons in the baseline model. Our neural models do not rely on any external resources.

Modeling Tweet-Level Syntactic and Semantic Information Using Bi-Directional GRNN

The baseline model does not explicitly capture the underlying semantic information of input tweets, such as dependency relations, co-references and negation scopes. We address this potential disadvantage by using a gated recurrent neural network layer to model tweet-level syntactic and semantic information of the input, capturing the interactions between its words.

The difference between our proposed models and the baseline model is shown in Figure 3, where x_i represents the embedding of the ith word in the input tweet, regardless whether it belongs to the target, the left context or the right context. For the baseline model (Figure 3(a)), features are extracted directly from individual word embeddings.

Recurrent tweet model (*RNN*). We make two extensions to the baseline system. First, a relatively simple extension can be made by adding a recurrent hidden layer over the input layer (Figure 3(b)). In this layer, each node h_i corresponds to the input x_i. However, rather than taking information from the word x_i alone, it is also connected with its predecessor h_{i-1}. Formally, the value of h_i is

$$h_i = \tanh\Big(W_1 x_i + U_1 h_{i-1} + b_1\Big),$$

where the matrices W_1 and U_1, and the vector b_1 are model parameters, and $\tanh$ is the activation function.

The hidden nodes h_i indirectly connect words in the tweet from left to right, thereby collecting information from not only x_i, but also from $[x_1, \cdots, x_{i-1}]$. In order to also capture information from $[x_{i+1}, \cdots, x_n]$, where n is the size of the tweet, we add a counterpart of h_i in the reverse direction,

$$h'_i = \tanh\Big(W'_1 x_i + U'_1 h'_{i+1} + b'_1\Big)$$

Here W'_1, U'_1 and b'_1 are the counterparts to W_1, U_1 and b, respectively.

We apply pooling functions over $h_i \oplus h'_i$ rather than x_i, where $\oplus$ represents vector concatenation, extracting features from the bi-directional recurrent sequence model.
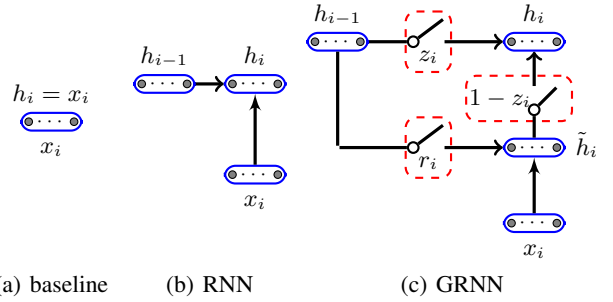

(a) baseline   (b) RNN   (c) GRNN

Figure 3: Comparisons between baseline, recurrent and gated recurrent tweet models. (The gate signals z_i and r_i are determined by x_i and h_{i-1}.)

Gates (*GRNN*). A second extension is to add gates to the recurrent hidden layers, which control information flow between nodes in the hidden and input layers. Gated neural networks have been shown to reduce informal bias of vanilla recurrent neural networks towards the ends of a sequence by better propagation of gradients.

The sub network structure over x_i is shown in Figure 3(c). Two main changes are made compared with Figure 3(b). First, a new hidden node $\tilde{h}_i$ is introduced, which represents a combination of h_{i-1} and x_i. Here $\tilde{h}_i$ is similar to h_i in Figure 3(b). Second, h_i now results from an interpolation of h_{i-1} and $\tilde{h}_i$, with the weights being controlled by two complimentary gates z_i and $\vec{1} - z_i$. The contribution to $\tilde{h}_i$ from h_{i-1} is also controlled by a gate r_i. Formally,

$$\tilde{h}_i = \tanh\Big(W_1 x_i + U_1(r_i \odot h_{i-1}) + b_1\Big),$$

where

$$r_i = \text{sigmoid}\Big(W_2 x_i + U_2 h_{i-1} + b_2\Big)$$

Each bit in r_i controls the contribution of the corresponding bit in h_{i-1} to $\tilde{h}_i$. Here $\odot$ represents the bit-wise product operation, and sigmoid is the sigmoid activation function. The interpolation of h_{i-1} to $\tilde{h}_i$ for h_i can be formulated as

$$h_i = (\vec{1} - z_i) \odot h_{i-1} + z_i \odot \tilde{h}_i,$$

where

$$z_i = \text{sigmoid}\Big(W_3 x_i + U_3 h_{i-1} + b_3\Big)$$

In the above equations, the matrices W_1, W_2, W_3, U_1, U_2 and U_3 and the vectors b_1, b_2 and b_3 are model parameters.

Similar to Figure 3(b), the gated recurrent neural network is applied in both directions, with the counterpart of h_i in the reverse direction being h'_i, and the corresponding parameters to $W_1, W_2, W_3, U_1, U_2, U_3, b_1, b_2$ and b_3 being $W'_1, W'_2, W'_3, U'_1, U'_2, U'_3, b'_1, b'_2$ and b'_3 respectively. *max*, *min*, *average*, *product* and *standard deviation* pooling are performed over the hidden node sequence $h_i \oplus h'_i$.

Modeling the Interaction between the Target and the Surrounding Context

Three-way gate (*G3*). The baseline in Figure 2(a) does not explicitly model the interaction between the left context and
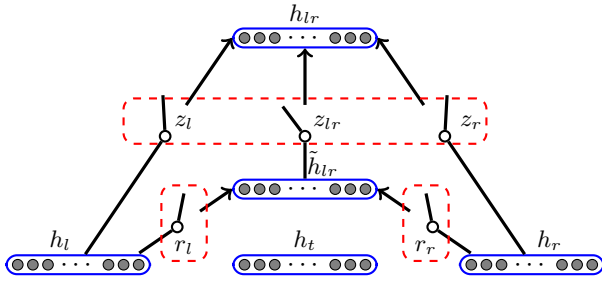
Figure 4: Gated neural network for choosing of left and right features for target. (The gate signals z_l and r_l are determined by h_l and h_t, z_r and r_r by h_r and h_t, and z_{lr} by $\tilde{h}_{lr}$ and h_t.)

right context in deciding the sentiment polarity of the target. We address this limitation by connecting the contexts using a gated neural network structure. As illustrated in Figure 4, we use h_l to represent the output of pooling functions over the left context, h_r to represent the output of pooling functions over the left context, and h_t to represent the output of pooling functions over the target entity span.

We first make a neural combination of h_l and h_r, denoting it with $\tilde{h}_{lr}$. Then a three-way bit-wise linear interpolation is performed to combine h_l, h_r and $\tilde{h}_{lr}$. The intuition of gates is similar to that of the gated recurrent hidden layer, where the gates control the contribution of each source to the targeted sentiment. This is motivated by the fact that sentiment signals can be dominated by the left context, the right context or a combination of both. Some examples have been discussed in the introduction. The target entity participates in the gate control weights, because the sentiment polarity over the target depends on the contexts and the target itself.

Similar to the gated recurrent neural network layer, in which the combination of h_{i-1} and x_i to form $\tilde{h}_i$, we use gates to control the combination of h_l and h_r in forming $\tilde{h}_{lr}$. Formally,

$$\tilde{h}_{lr} = \tanh\Big(W_1(r_l \odot h_l) + U_1(r_r \odot h_r) + b_1\Big)$$

where r_l and r_r are the bit-wise gates to control signal flow from h_l and h_r, respectively. Here

$$r_l = \text{sigmoid}\Big(W_2 h_l + U_2 h_t + b_2\Big)$$

$$r_r = \text{sigmoid}\Big(W_3 h_l + U_3 h_t + b_3\Big)$$

The linear interpolation between h_l, h_r and $\tilde{h}_{lr}$ can be formulated as

$$h_{lr} = z_l \odot h_l + z_r \odot h_r + z_{lr} \odot \tilde{h}_{lr},$$

where z_l, z_r and z_{lr} are the gate control weights, $z_l + z_r + z_{lr} = \vec{1}$, and

$$z_l \infty \exp\Big(W_4 h_l + U_4 h_t + b_4\Big)$$

$$z_r \infty \exp\Big(W_5 h_r + U_5 h_t + b_5\Big)$$

$$z_{lr} \infty \exp\Big(W_6 \tilde{h}_{er} + U_6 h_t + b_6\Big)$$

	#Targets	#+	#-	#0
training	9,489	2,416	2,384	4,689
development	1,036	255	272	509
testing	1,170	294	295	581

Table 1: Experimental corpus statistics.

A final note is that we apply dimensionality reduction to the pooling result of the left context, the target and the right context in order to derive h_l, h_t and h_r:

$$h_l = \tanh\Big(\text{pooling(left\_context)} \cdot W_7 + b_7\Big)$$

$$h_t = \tanh\Big(\text{pooling(target)} \cdot W_8 + b_8\Big)$$

$$h_r = \tanh\Big(\text{pooling(right\_context)} \cdot W_9 + b_9\Big)$$

In the above equations, the matrices $W_1, W_2, \cdots, W_9$, $U_1, U_2, \cdots, U_6$ and the vectors $b_1, b_2, \cdots, b_9$ are model parameters.

Training Method

Given a set of annotated training examples, our models are trained to minimize a cross-entropy loss objective, with a l_2 regularization term, defined by

$$L(\theta) = -\sum_i \log p_{t_i} + \frac{\lambda}{2} \parallel \theta \parallel^2$$

where θ is the set of model parameters, p_{t_i} is the probability of the ith training example as given by the model and λ is the regularization hyper-parameter.

We apply online training, where model parameters are optimized by using Adagrad (Duchi, Hazan, and Singer 2011). In order to avoid overfitting, we use the dropout technique (Hinton et al. 2012), randomly dropping some dimensions of the input word embedding with a fixed probability p_{drop}.

Experiments

Data

Our experimental data are collected from three sources, including Dong et al. (2014), which consists of 6,940 examples, the MPQA corpus[1], from which we collected 1,467 targets that have been annotated as being positive/negative, and the corpus of Mitchell et al. (2013)[2], which consists of 3,288 entities. We merge the three sources of annotations, shuffle them randomly, and divide them into training, development and testing sets. Table 1 shows the corpus statistics.

Evaluation Metrics

We use two evaluation metrics for targeted sentiment analysis, including the classification accuracy and macro F1-measure, which is computed by averaged F1-measure over the three sentiment classes (positive, negative and neutral).

---

[1] http://mpqa.cs.pitt.edu/corpora/mpqa_corpus/
[2] http://www.m-mitchell.com/code/index.html

Type	hyper-parameters
Network structure	$H_{rnn} = 150$, $H_{context} = 150$
Training	$\lambda = 10^{-8}$, $\alpha = 0.01$, $p_{drop} = 0.25$

Table 2: Hyper-parameter values in our model.

Model	Accuracy	F1(macro)
baseline	65.73	60.44
+*RNN*	68.05	65.94
+*GRNN*	71.43	69.06
+*G3*	70.84	67.33
final(+*GRNN*+*G3*)	72.49	70.57

Table 3: Development results.

Training of Word Embeddings

We follow Vo and Zhang (2015), using the concatenation of two types of word embeddings for each word. The first type of word embeddings is obtained directly from Tang et al. (2014)[3], which incorporates sentiment information. The size of this type of word embedding is 50.

The second type of word embeddings is obtained by using the *word2vec* tool[4] over large-scale sentiment-expressing tweets, which are crawled automatically using TweetAPI[5]. We follow Go, Bhayani, and Huang (2009) and use emoticons as sentiment indicators for tweets. We collect 5 million tweets, and train word embeddings with a dimension size of 100, using the skip-gram algorithm (Mikolov et al. 2013). Thus, the final combined word embeddings have a dimension size of 150.

Parameter Settings

There are several hyper-parameters in our neural models. We tune them according to the development dataset. Table 2 shows the values, where H_{rnn} denotes the dimension size of the recurrent neural layers, $H_{context}$ denotes the dimension reduction sizes for the left context representation, the right context representation and the target representation, λ denotes the regularization hyper-parameter, α denotes the initial step value of parameter updating, and p_{drop} denotes the dropout value. All the matrices in the model are initialized randomly with a uniform distribution in (-0.01, 0.01).

Development Experiments

The Effect of RNNs We investigate the effectiveness of RNNs, which are used to automatically extract syntactic and semantic information from sentences, by comparing the baseline model with two models that integrate *RNN* and *GRNN*, respectively. Table 3 shows the development results, which demonstrate an accuracy improvement of 2.32% by using simple *RNN*, and a larger improvement of 5.70% by

---

[3] http://ir.hit.edu.cn/~dytang/
[4] http://word2vec.googlecode.com/
[5] https://twitter.com/twitterapi

Model	Accuracy	F1 (macro)
baseline	65.13	58.85
+GRNN	69.65[‡]	67.11[‡]
+G3	67.79[‡]	63.31[‡]
final	**71.96**[‡]	**69.55**[‡]
Vo and Zhang (2015)	69.57	65.56

Table 4: Final results on the test dataset, where [‡] denotes a p-value below 10^{-5} by pairwise t-test, compared with the baseline system.

using *GRNN*. The results by F1-scores are similar. These results show that RNNs are effective, and the use of gates can give better performances in targeted sentiment analysis.

The Effect of G3 We study the isolated effectiveness of our second gated neural network, which is used to model the interaction between targets and their surrounding contexts. The results are obtained by applying *G3* directly over the pooling results of the baseline model h_t, h_l and h_r for the target and its contexts. As shown in Table 3, *G3* gives an absolute 5.11% accuracy improvement and a 8.62% F1-score improvement over the baseline model, demonstrating that the gated structure is highly effective.

Combination of GRNN and G3 We apply both *GRNN* and *G3* in the final model. As shown in Table 3, the final model gives an absolute accuracy improvement of 1.06% over the model with *GRNN*, and an absolute accuracy improvement of 1.64% over the model with *G3*. The improved accuracies mainly result from the combination of complementary information from *GRNN* and *G3*.

Final Results

The final results on the test dataset are given in Table 4, which shows the performances of the baseline, baselines with separate *GRNN* and *G3*, and the final model (baseline with both *GRNN* and *G3*), respectively. We can see that both *GRNN* and *G3* bring significant improvements, and the combination of the two leads to the best results, which are consistent with the development experiments.

We also compare our system with the system of Vo and Zhang (2015), which uses more contextual information and sentiment lexicons. As shown in the Table 4, Vo and Zhang (2015) gives significantly higher results compared with our baseline thanks to the additional sources of information. On the other hand, our final model gives significantly better performances compared with their model (p-value below 10^{-5} using t-test). There has been other previous work on targeted sentiment analysis (Jiang et al. 2011; Tang et al. 2014; Dong et al. 2014), which we did not include in Table 4. This is mainly because their results are reported on a different dataset, on which Vo and Zhang (2015) has given by far the best reported accuracies.

Discussion

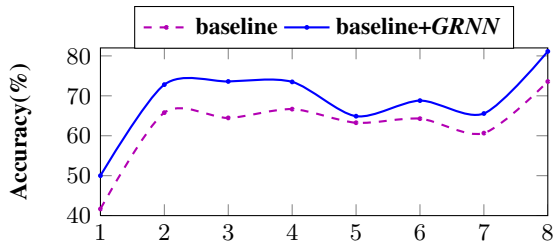We make more detailed analysis of *GRNN* and *G3* quantitatively and qualitatively by examining the final test outputs

Figure 5: Targeted sentiment accuracies against sentence length.

Figure 6: Targeted sentiment accuracies against the length difference between the left and right contexts.

baseline	baseline + *GRNN*
#nowplaying **[lady gaga]**$_+$ - let love down	#nowplaying **[lady gaga]**$_0$ - let love down
[Michael]$_0$ dislikes to work with him. I'm very glad to hear that.	**[Michael]**$_+$ dislikes to work with him. I'm very glad to hear that.
haha youre not obsessed, you just reaaaaaally love the **[lakers]**$_0$.	haha youre not obsessed, you just reaaaaaally love the **[lakers]**$_+$.

Table 5: Example outputs of baseline and GRNN.

baseline + *GRNN*	Final
Did you upgrade to **[Windows 7]**$_0$? I cannot get a fresh install installed, stupid drivers errors	Did you upgrade to **[Windows 7]**$_-$? I cannot get a fresh install installed, stupid drivers errors
Lay on my sofa and listen to (toxic) by **[Britney Spears]**$_-$. Another boring day. Nothing to do.	Lay on my sofa and listen to (toxic) by **[Britney Spears]**$_0$. Another boring day. Nothing to do.

Table 6: Example outputs of GRNN and final model.

of relevant models.

Impact of *GRNN* The impact of *GRNN* can be analyzed by contrasting the outputs of the baseline model and those of the model by adding *GRNN* only. Figure 5 shows the accuracies of the two models with respect to the sentence length. We divide the test sentences into eight bins, where the ith bin contains sentences with length between $[5*i\text{-}4, 5*i]$. *GRNN* gives higher accuracies on sentences of all length, showing that its effectiveness is not negatively affected when the input sentence grows longer.

Table 5 shows some examples that illustrate the usefulness of tweet-level information from *GRNN*. In the first example, the baseline assigns a positive sentiment polarity to "[**lady gaga**]", largely because of the positive word "love" in the right context, which dominates the pooling result. In contrast, the sentence structure indicates that the phrase after "-" is a noun phrase, and therefore the sentiment signal from "love" is rather weak. This is captured by the gated recurrent hidden layer, from which pooling is extracted. The second example contains both positive (i.e. "glad") and negative (i.e. "dislike") words in the right context, and pooling alone results in an incorrect neutral sentiment label. The third example demonstrates the effectiveness of the gated hidden layer in capturing negation, which simple pooling cannot model.

Impact of *G3* The impact of *G3* can be analyzed by contrasting the outputs of the final model with those of the model with a *GRNN* only. Figure 6 shows their accuracies with respect to absolute size difference between the left and right contexts. The ith result denotes an absolute size difference within $[2*i\text{-}1, 2*i]$. Sentences with absolute size differences above 16 are merged into the eighth result. Intuitively,
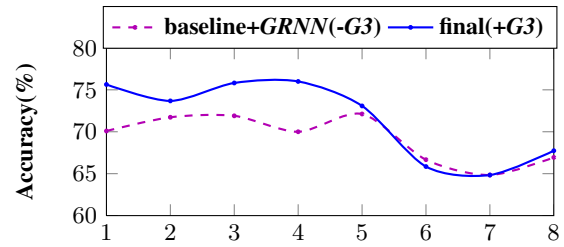
if one context is significantly longer, it is likely to dominate the sentiment polarity. Therefore the cases with an equally-large left and right contexts should be more challenging. As shown in the Figure, *G3* leads to better accuracies on the challenging sentences, which shows its effectiveness in capturing the interaction between the target and its contexts.

Some examples are shown in Table 6. In the first example, the sentiment on the targeted is decided by the right context; in the second example, the sentiment is decided by the left context. The final model gives correct predictions in both cases, thanks to *G3*.

Conclusion

We proposed two gated neural networks for targeted sentiment analysis, one being used to capture tweet-level syntactic and semantic information, and the other being used to model the interactions between the left context and the right context of a given target. Gates are used in both neural networks, so that the target influences the selection of sentiment signals over the context. Experiments demonstrated that the two gated neural networks are effective in targeted sentiment analysis, bringing significant improvements.

Acknowledgments

References

Bengio, Y.; Simard, P.; and Frasconi, P. 1994. Learning long-term dependencies with gradient descent is difficult. *Neural Networks, IEEE Transactions on* 5(2):157–166.

Cho, K.; van Merriënboer, B.; Bahdanau, D.; and Bengio, Y. 2014a. On the properties of neural machine translation: Encoder–decoder approaches. *Syntax, Semantics and Structure in Statistical Translation* 103.

Cho, K.; van Merrienboer, B.; Gulcehre, C.; Bahdanau, D.; Bougares, F.; Schwenk, H.; and Bengio, Y. 2014b. Learning phrase representations using rnn encoder–decoder for statistical machine translation. In *EMNLP*, 1724–1734.

Chung, J.; Gulcehre, C.; Cho, K.; and Bengio, Y. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*.

Dong, L.; Wei, F.; Tan, C.; Tang, D.; Zhou, M.; and Xu, K. 2014. Adaptive recursive neural network for target-dependent twitter sentiment classification. In *ACL*, 49–54.

dos Santos, C., and Gatti, M. 2014. Deep convolutional neural networks for sentiment analysis of short texts. In *Proceedings of COLING 2014*, 69–78.

Duchi, J.; Hazan, E.; and Singer, Y. 2011. Adaptive subgradient methods for online learning and stochastic optimization. *JMLR* 12:2121–2159.

Go, A.; Bhayani, R.; and Huang, L. 2009. Twitter sentiment classification using distant supervision. *CS224N Project Report, Stanford* 1:12.

Graves, A. 2012. *Supervised Sequence Labelling with Recurrent Neural Networks*, volume 385. Springer.

Hinton, G. E.; Srivastava, N.; Krizhevsky, A.; Sutskever, I.; and Salakhutdinov, R. R. 2012. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580*.

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

Hu, M., and Liu, B. 2004. Mining and summarizing customer reviews. In *SIGKDD*, 168–177.

Irsoy, O., and Cardie, C. 2013. Bidirectional recursive neural networks for token-level labeling with structure. *CoRR* abs/1312.0493.

Jiang, L.; Yu, M.; Zhou, M.; Liu, X.; and Zhao, T. 2011. Target-dependent twitter sentiment classification. In *Proceedings of the 49th ACL*, 151–160.

Jin, W.; Ho, H. H.; and Srihari, R. K. 2009. A novel lexicalized hmm-based learning framework for web opinion mining. In *Proceedings of the 26th ICML*, 465–472.

Kalchbrenner, N.; Grefenstette, E.; and Blunsom, P. 2014. A convolutional neural network for modelling sentences. In *ACL*, 655–665.

Li, F.; Han, C.; Huang, M.; Zhu, X.; Xia, Y.-J.; Zhang, S.; and Yu, H. 2010. Structure-aware review mining and summarization. In *Proceedings of the 23rd COLING*, 653–661.

Mikolov, T.; Karafiát, M.; Burget, L.; Cernocký, J.; and Khudanpur, S. 2010. Recurrent neural network based language model. In *INTERSPEECH 2010*, 1045–1048.

Mikolov, T.; Chen, K.; Corrado, G.; and Dean, J. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Mitchell, M.; Aguilar, J.; Wilson, T.; and Van Durme, B. 2013. Open domain targeted sentiment. In *EMNLP 2013*, 1643–1654.

Mohammad, S. M.; Kiritchenko, S.; and Zhu, X. 2013. Nrccanada: Building the state-of-the-art in sentiment analysis of tweets. In *Semeval 2013*, volume 2, 321–327.

Nakov, P.; Rosenthal, S.; Kozareva, Z.; Stoyanov, V.; Ritter, A.; and Wilson, T. 2013. Semeval-2013 task 2: Sentiment analysis in twitter. In *SemEval 2013*, 312–320.

Pang, B.; Lee, L.; and Vaithyanathan, S. 2002. Thumbs up?: sentiment classification using machine learning techniques. In *ACL*, 79–86.

Paulus, R.; Socher, R.; and Manning, C. D. 2014. Global belief recursive neural networks. In *NIPS*, 2888–2896.

Popescu, A.-M., and Etzioni, O. 2007. Extracting product features and opinions from reviews. In *Natural language processing and text mining*. Springer. 9–28.

Socher, R.; Perelygin, A.; Wu, J. Y.; Chuang, J.; Manning, C. D.; Ng, A. Y.; and Potts, C. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *EMNLP*, 1631–1642.

Sutskever, I.; Vinyals, O.; and Le, Q. V. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, 3104–3112.

Tang, D.; Wei, F.; Yang, N.; Zhou, M.; Liu, T.; and Qin, B. 2014. Learning sentiment-specific word embedding for twitter sentiment classification. In *ACL*, 1555–1565.

Vo, D.-T., and Zhang, Y. 2015. Target-dependent twitter sentiment classification with rich automatic features. In *Proceedings of the IJCAI*, 1347–1353.

Wang, X.; Wei, F.; Liu, X.; Zhou, M.; and Zhang, M. 2011. Topic sentiment analysis in twitter: a graph-based hashtag sentiment classification approach. In *CIKM*, 1031–1040.

Wiebe, J.; Wilson, T.; and Cardie, C. 2005. Annotating expressions of opinions and emotions in language. *Language resources and evaluation* 39(2-3):165–210.

Yang, B., and Cardie, C. 2013. Joint inference for fine-grained opinion extraction. In *ACL (1)*, 1640–1649.

Yi, J.; Nasukawa, T.; Bunescu, R.; and Niblack, W. 2003. Sentiment analyzer: Extracting sentiments about a given topic using natural language processing techniques. In *Proceedings of the ICDM*, 427–434.

Zhang, M.; Zhang, Y.; and Vo, D.-T. 2015. Neural networks for open domain targeted sentiment. In *Proceedings of the 2015 Conference on EMNLP*.

Zhou, S.; Chen, Q.; Wang, X.; and Li, X. 2014. Hybrid deep belief networks for semi-supervised sentiment classification. In *Proceedings of COLING 2014*, 1341–1349.