Recurrent Attention Network on Memory for Aspect Sentiment Analysis

Peng Chen Zhongqian Sun Lidong Bing* Wei Yang
AI Lab
Tencent Inc.
{patchen, sallensun, lyndonbing, willyang}@tencent.com

Abstract

We propose a novel framework based on neural networks to identify the sentiment of opinion targets in a comment/review. Our framework adopts multiple-attention mechanism to capture sentiment features separated by a long distance, so that it is more robust against irrelevant information. The results of multiple attentions are non-linearly combined with a recurrent neural network, which strengthens the expressive power of our model for handling more complications. The weighted-memory mechanism not only helps us avoid the labor-intensive feature engineering work, but also provides a tailor-made memory for different opinion targets of a sentence. We examine the merit of our model on four datasets: two are from SemEval2014, i.e. reviews of restaurants and laptops; a twitter dataset, for testing its performance on social media data; and a Chinese news comment dataset, for testing its language sensitivity. The experimental results show that our model consistently outperforms the state-of-the-art methods on different types of data.

1 Introduction

The goal of aspect sentiment analysis is to identify the sentiment polarity (i.e., negative, neutral, or positive) of a specific opinion target expressed in a comment/review by a reviewer. For example, in "I bought a mobile phone, its camera is wonderful but the battery life is short", there are three opinion targets, "camera", "battery life", and "mobile phone". The reviewer has a positive sentiment on the "camera", a negative sentiment on the "battery life", and a mixed sentiment on the "mobile phone". Sentence-oriented sentiment analysis methods (Socher et al., 2011; Appel et al., 2016) are not capable to capture such fine-grained sentiments on opinion targets.

In order to identify the sentiment of an individual opinion target, one critical task is to model appropriate context features for the target in its original sentence. In simple cases, the sentiment of a target is identifiable with a syntactically nearby opinion word, e.g. "wonderful" for "camera". However, there are many cases in which opinion words are enclosed in more complicated contexts. E.g., "Its camera is not wonderful enough" might express a neutral sentiment on "camera", but not negative. Such complications usually hinder conventional approaches to aspect sentiment analysis.

To model the sentiment of the above phrase-like word sequence (i.e. "not wonderful enough"), LSTM-based methods are proposed, such as target dependent LSTM (TD-LSTM) (Tang et al., 2015). TD-LSTM might suffer from the problem that after it captures a sentiment feature far from the target, it needs to propagate the feature word by word to the target, in which case it's likely to lose this feature, such as the feature "cost-effective" for "the phone" in "My overall feeling is that the phone, after using it for three months and considering its price, is really cost-effective".[1] Attention mechanism, which has been successfully used in machine translation (Bahdanau et al., 2014), can enforce a model to pay more attention to the important part of a sentence. There are already some works using attention in sentiment analysis to exploit this advantage (Wang et al., 2016; Tang et al., 2016). Another observation is that some types of

*Corresponding author.

[1] Although LSTM could keep information for a long distance by preventing the vanishing gradient problem, it usually requires a large training corpus to capture the flexible usage of parenthesis.

Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, pages 452–461
Copenhagen, Denmark, September 7–11, 2017.

sentence structures are particularly challenging for target sentiment analysis. For example, in "Except Patrick, all other actors don't play well", the word "except" and the phrase "don't play well" produce a positive sentiment on "Patrick". It's hard to synthesize these features just by LSTM, since their positions are dispersed. Single attention based methods (e.g. (Wang et al., 2016)) are also not capable to overcome such difficulty, because attending multiple words with one attention may hide the characteristic of each attended word.

In this paper, we propose a novel framework to solve the above problems in target sentiment analysis. Specifically, our framework first adopts a bidirectional LSTM (BLSTM) to produce the memory (i.e. the states of time steps generated by LSTM) from the input, as bidirectional recurrent neural networks (RNNs) were found effective for a similar purpose in machine translation (Bahdanau et al., 2014). The memory slices are then weighted according to their relative positions to the target, so that different targets from the same sentence have their own tailor-made memories. After that, we pay multiple attentions on the position-weighted memory and nonlinearly combine the attention results with a recurrent network, i.e. GRUs. Finally, we apply softmax on the output of the GRU network to predict the sentiment on the target.

Our framework introduces a novel way of applying multiple-attention mechanism to synthesize important features in difficult sentence structures. It's sort of analogous to the cognition procedure of a person, who might first notice part of the important information at the beginning, then notices more as she reads through, and finally combines the information from multiple attentions to draw a conclusion. For the above sentence, our model may attend the word "except" first, and then attends the phrase "don't play well", finally combines them to generate a positive feature for "Patrick". Tang et al. (2016) also adopted the idea of multiple attentions, but they used the result of a previous attention to help the next attention attend more accurate information. Their vector fed to softmax for classification is only from the final attention, which is essentially a linear combination of input embeddings (they did not have a memory component). Thus, the above limitation of single attention based methods also holds for (Tang et al., 2016). In contrast, our model combines the results of multiple attentions with a GRU network, which has different behaviors inherited from RNNs, such as forgetting, maintaining, and non-linearly transforming, and thus allows a better prediction accuracy.

We evaluate our approach on four datasets: the first two come from SemEval 2014 (Pontiki et al., 2014), containing reviews of restaurant domain and laptop domain; the third one is a collection of tweets, collected by (Dong et al., 2014); to examine whether our framework is language-insensitive (since languages show differences in quite a few aspects in expressing sentiments), we prepared a dataset of Chinese news comments with people mentions as opinion targets. The experimental results show that our model performs well for different types of data, and consistently outperforms the state-of-the-art methods.

2 Related Work

The task of aspect sentiment classification belongs to entity-level sentiment analysis. Conventional representative methods for this task include rule-based methods (Ding et al., 2008) and statistic-based methods (Jiang et al., 2011; Zhao et al., 2010). Ganapathibhotla and Liu (2008) extracted 2-tuples of (opinion target, opinion word) from comments and then identified the sentiment of opinion targets. Deng and Wiebe (2015) adopted Probabilistic Soft Logic to handle the task. There are also statistic-based approaches which employ SVM (Jiang et al., 2011) or MaxEnt-LDA (Zhao et al., 2010). These methods need either laborious feature engineering work or massive extra-linguistic resources.

Neural Networks (NNs) have the capability of fusing original features to generate new representations through multiple hidden layers. Recursive NN (Rec-NN) can conduct semantic compositions on tree structures, which has been used for syntactic analysis (Socher et al., 2010) and sentence sentiment analysis (Socher et al., 2013). (Dong et al., 2014; Nguyen and Shirai, 2015) adopted Rec-NN for aspect sentiment classification, by converting the opinion target as the tree root and propagating the sentiment of targets depending on the context and syntactic relationships between them. However, Rec-NN needs dependency parsing which is likely ineffective on nonstandard texts such as news comments and tweets. (Chen et al., 2016) employed Convolution NNs to identify the senti-

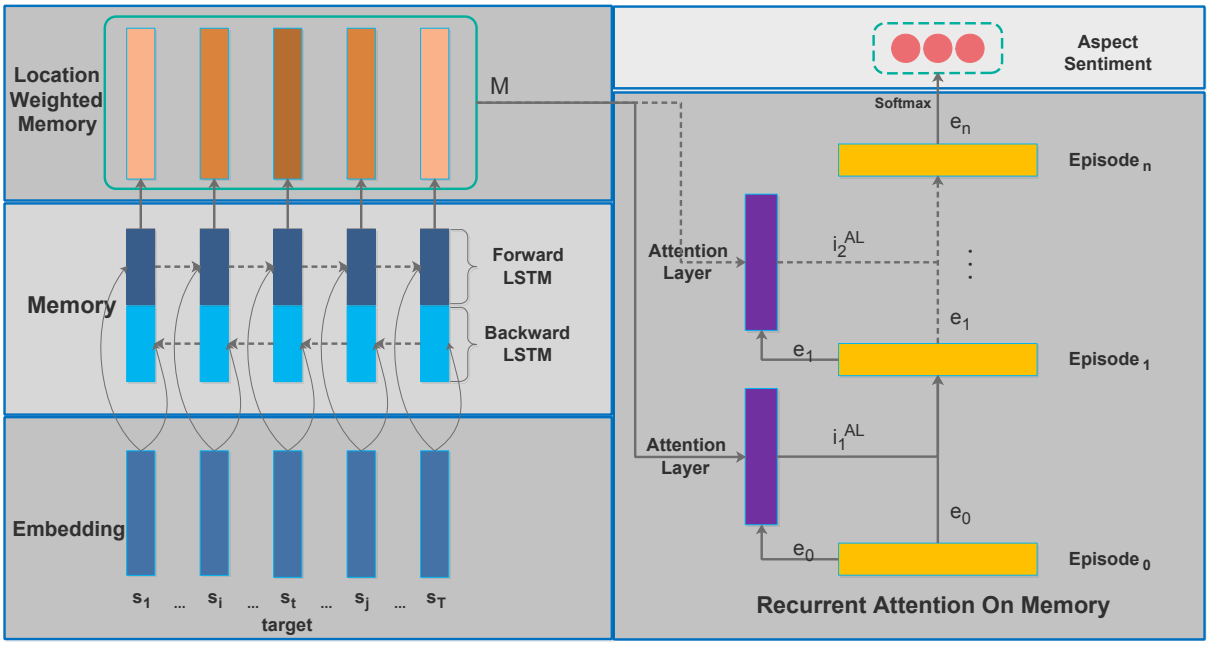


Figure 1: Model architecture. The dotted lines on the right indicate a layer may or may not be added.

ment of a clause which is then used to infer the sentiment of the target. The method has an assumption that an opinion word and its target lie in the same clause. TD-LSTM (Tang et al., 2015) utilizes LSTM to model the context information of a target by placing the target in the middle and propagating the state word by word from the beginning and the tail to the target respectively to capture the information before and after it. Nevertheless, TD-LSTM might not work well when the opinion word is far from the target, because the captured feature is likely to be lost ((Cho et al., 2014) reported similar problems of LSTM-based models in machine translation).

(Graves et al., 2014) introduced the concept of memory for NNs and proposed a differentiable process to read and write memory, which is called Neural Turing Machine (NTM). Attention mechanism, which has been used successfully in many areas (Bahdanau et al., 2014; Rush et al., 2015), can be treated as a simplified version of NTM because the size of memory is unlimited and we only need to read from it. Single attention or multiple attentions were applied in aspect sentiment classification in some previous works (Wang et al., 2016; Tang et al., 2016). One difference between our method and (Tang et al., 2016) is that we introduce a memory module between the attention module and the input module, thus our method can synthesize features of word sequences such as sentiment phrases (e.g. "not wonderful enough"). More importantly, we combine the results of attentions in a nonlinear way. (Wang et al., 2016) only uses one attention, while our model uses multiple attentions. The effectiveness of multiple attentions was also investigated in QA task (Kumar et al., 2015), which shows that multiple attentions allow a model to attend different parts of the input during each pass. (Kumar et al., 2015) assigns attention scores to memory slices independently and their attention process is more complex, while we produce a normalized attention distribution to attend information from the memory.

3 Our Model

The architecture of our model is shown in Figure 1, which consists of five modules: input module, memory module, position-weighted memory module, recurrent attention module, and output module. Suppose the input sentence is $\mathbf{s} = \{s_1, \ldots, s_{\tau-1}, s_\tau, s_{\tau+1}, \ldots, s_T\}$, the goal of our model is to predict the sentiment polarity of the target s_τ. For simplicity, we notate a target as one word here, where necessary, we will elaborate how to handle phrase-form targets, e.g. "battery life".

3.1 Input Embedding

Let $\mathbb{L} \in \mathbb{R}^{d \times |V|}$ be an embedding lookup table generated by an unsupervised method such as GloVe (Pennington et al., 2014) or CBOW

(Mikolov et al., 2013), where d is the dimension of word vectors and $|V|$ is the vocabulary size. The input module retrieves the word vectors from $\mathbb{L}$ for an input sequence and gets a list of vectors $\{v_1, \ldots, v_t, \ldots, v_T\}$ where $v_t \in \mathbb{R}^d$. $\mathbb{L}$ may or may not be tuned in the training of our framework. If it is not tuned, the model can utilize the words' similarity revealed in the original embedding space. If it is tuned, we expect the model would capture some intrinsic information that is useful for the sentiment analysis task.

3.2 BLSTM for Memory Building

MemNet (Tang et al., 2016) simply used the sequence of word vectors as memory, which cannot synthesize phrase-like features in the original sentence. It is straightforward to achieve the goal with the models of RNN family. In this paper, we use Deep Bidirectional LSTM (DBLSTM) to build the memory which records all information to be read in the subsequent modules.

At each time step t, the forward LSTM not only outputs the hidden state $\overrightarrow{h}_t^l$ at its layer l ($\overrightarrow{h}_t^0 = v_t$) but also maintains a memory $\overrightarrow{c}_t^l$ inside its hidden cell. The update process at time t is as follows:

$$i = \sigma(\overrightarrow{W}_i \overrightarrow{h}_t^{l-1} + \overrightarrow{U}_i \overrightarrow{h}_{t-1}^l) \quad (1)$$

$$f = \sigma(\overrightarrow{W}_f \overrightarrow{h}_t^{l-1} + \overrightarrow{U}_f \overrightarrow{h}_{t-1}^l) \quad (2)$$

$$o = \sigma(\overrightarrow{W}_o \overrightarrow{h}_t^{l-1} + \overrightarrow{U}_o \overrightarrow{h}_{t-1}^l) \quad (3)$$

$$g = \tanh(\overrightarrow{W}_g \overrightarrow{h}_t^{l-1} + \overrightarrow{U}_g \overrightarrow{h}_{t-1}^l) \quad (4)$$

$$\overrightarrow{c}_t^l = f \odot \overrightarrow{c}_{t-1}^l + i \odot g \quad (5)$$

$$\overrightarrow{h}_t^l = o \odot \tanh(\overrightarrow{c}_t^l) \quad (6)$$

where σ and $\tanh$ are sigmoid and hyperbolic tangent functions, $\overrightarrow{W}_i, \overrightarrow{W}_f, \overrightarrow{W}_o, \overrightarrow{W}_g \in \mathbb{R}^{\overrightarrow{d}_l \times \overrightarrow{d}_{l-1}}$, $\overrightarrow{U}_i, \overrightarrow{U}_f, \overrightarrow{U}_o, \overrightarrow{U}_g \in \mathbb{R}^{\overrightarrow{d}_l \times \overrightarrow{d}_l}$, and $\overrightarrow{d}_l$ is the number of hidden cells at the layer l of the forward LSTM. The gates $i, f, o \in \mathbb{R}^{\overrightarrow{d}_l}$ simulate binary switches that control whether to update the information from the current input, whether to forget the information in the memory cells, and whether to reveal the information in memory cells to the output, respectively. The backward LSTM does the same thing, except that its input sequence is reversed. If there are L layers stacked in the BLSTM, the final memory generated in this module is $M^* = \{m_1^*, \ldots, m_t^*, \ldots, m_T^*\}$, where $m_t^* = (\overrightarrow{h}_t^L, \overleftarrow{h}_t^L) \in \mathbb{R}^{\overrightarrow{d}_L + \overleftarrow{d}_L}$. In our framework, we use 2 layers of BLSTM to build the memory, as it generally performs well in NLP tasks (Karpathy et al., 2015).

3.3 Position-Weighted Memory

The memory generated in the above module is the same for multiple targets in one comment, which is not flexible enough for predicting respective sentiments of these targets. To ease this problem, we adopt an intuitive method to edit the memory to produce a tailor-made input memory for each target. Specifically, the closer to the target a word is, the higher its memory slide is weighted. We define the distance as the number of words between the word and the target. One might want to use the length of the path from the specific word to the target in the dependency tree as the distance, which is a worthwhile option to try in the future work, given the condition that dependency parsing on the input text is effective enough. Precisely, the weight for the word at position t is calculated as:

$$w_t = 1 - \frac{|t - \tau|}{t_{max}} \quad (7)$$

where t_{max} is truncation length of the input. We also calculate $u_t = \frac{t-\tau}{t_{max}}$ to memorize the relative offset between each word and the target. If the target is a phrase, the distance (i.e. $t - \tau$) is calculated with its left or right boundary index according to which side w_t locates. The final position-weighted memory of a target is $M = \{m_1, \ldots, m_t, \ldots, m_T\}$ where $m_t = (w_t \cdot m_t^*, u_t) \in \mathbb{R}^{\overrightarrow{d}_L + \overleftarrow{d}_L + 1}$. The weighted memory is designed to up-weight nearer sentiment words, and the recurrent attention module, discussed below, attends long-distance sentiment words. Thus, they work together to expect a better prediction accuracy.

3.4 Recurrent Attention on Memory

To accurately predict the sentiment of a target, it is essential to: (1) correctly distill the related information from its position-weighted memory; and (2) appropriately manufacture such information as the input of sentiment classification. We employ multiple attentions to fulfil the first aspect, and a recurrent network for the second which nonlinearly combines the attention results with GRUs (since GRUs have less number of parameters). For example, "except" and "don't play well" in "Except Patrick, all other actors don't play well" are attended by different attentions, and combined to produce a positive sentiment on "Patrick".

Particularly, we employ a GRU to update the episode e after each attention. Let e_{t-1} denote the episode at the previous time and i_t^{AL} is the current information attended from the memory M, and the process of updating e_t is as follows:

$$r = \sigma(W_r i_t^{AL} + U_r e_{t-1}) \quad (8)$$

$$z = \sigma(W_z i_t^{AL} + U_z e_{t-1}) \quad (9)$$

$$\tilde{e}_t = \tanh(W_x i_t^{AL} + W_g(r \odot e_{t-1})) \quad (10)$$

$$e_t = (1 - z) \odot e_{t-1} + z \odot \tilde{e}_t \quad (11)$$

where $W_r, W_z \in \mathbb{R}^{H \times (\overrightarrow{d}_L + \overleftarrow{d}_L + 1)}$, $U_r, U_z \in \mathbb{R}^{H \times H}$, $W_g \in \mathbb{R}^{H \times (\overrightarrow{d}_L + \overleftarrow{d}_L + 1)}$, $W_x \in \mathbb{R}^{H \times H}$, and H is the hidden size of GRU. As we can see from Equations (10) and (11), the state of episode e_t is the interpolation of e_{t-1} and the candidate hidden vector $\tilde{e}_t$. A vector of 0's is used as e_0.

For calculating the attended information i_t^{AL} at t, the input of an attention layer (AL for short) includes the memory slices $m_j (1 \leq j \leq T)$ and the previous episode e_{t-1}. We first calculate the attention score of each memory slice as follows:

$$g_j^t = W_t^{AL}(m_j, e_{t-1}[, v_\tau]) + b_t^{AL}, \quad (12)$$

where $[, v_\tau]$ indicates when the attention result relies on particular aspects such as those of products, we also add the target vector v_τ because different product aspects have different preference on opinion words; when the target is a person, there is no need to do so. If the target is a phrase, v_τ takes the average of word embeddings. We utilize the previous episode for the current attention, since it can guide the model to attend different useful information. (Tang et al., 2016) also adopts multiple attentions, but they don't combine the results of different attentions.

Then we calculate the normalized attention score of each memory slice as:

$$\alpha_j^t = \frac{\exp(g_j^t)}{\sum_k \exp(g_k^t)}. \quad (13)$$

Finally, the inputs to a GRU (i.e. Eqs. 8 to 11) at time t are the episode e_{t-1} at time $t-1$ and the content i_t^{AL}, which is read from the memory as:

$$i_t^{AL} = \sum_{j=1}^{T} \alpha_j^t m_j. \quad (14)$$

3.5 Output and Model Training

After N-time attentions on the memory, the final episode e_N serves as the feature and is fed into a softmax layer to predict the target sentiment.

The model is trained by minimizing the cross entropy plus an L_2 regularization term:

$$\mathcal{L} = \sum_{(x,y) \in D} \sum_{c \in C} y^c \log f^c(x; \theta) + \lambda \parallel \theta \parallel^2 \quad (15)$$

where C is the sentiment category set, D is the collection of training data, $y \in \mathbb{R}^{|C|}$ is a one-hot vector where the element for the true sentiment is 1, $f(x; \theta)$ is the predicted sentiment distribution of the model, λ is the weight of L_2 regularization term. We also adopt dropout and early stopping to ease overfitting.

4 Experiments

4.1 Experimental Setting

We conduct experiments on four datasets, as shown in Table 1. The first two are from SemEval 2014 (Pontiki et al., 2014), containing reviews of restaurant and laptop domains, which are widely used in previous works. The third one is a collection of tweets, collected by (Dong et al., 2014). The last one is prepared by us for testing the language sensitivity of our model, which contains Chinese news comments and has politicians and entertainers as opinion targets. We purposely add more negation, contrastive, and question comments to make it more challenging. Each comment is annotated by at least two annotators, and only if they agree with each other, the comment will be added into our dataset. Moreover, we replace each opinion target (i.e. word/phrase of pronoun or person name) with a placeholder, as did in (Dong et al., 2014). For the first two datasets, we removed a few examples having the "conflict label", e.g., "Certainly not the best sushi in New York, however, it is always fresh" (Pontiki et al., 2014).

We use 300-dimension word vectors pre-trained by GloVe (Pennington et al., 2014) (whose vocabulary size is 1.9M[2]) for our experiments on the English datasets, as previous works did (Tang et al., 2016). Some works employed domain-specific training corpus to learn embeddings for better performance, such as TD-LSTM (Tang et al., 2015) on the tweet dataset. In contrast, we prefer to use

[2] http://nlp.stanford.edu/projects/glove/

Dataset		Negative	Neutral	Positive
Laptop reviews	Training	858	454	980
	Testing	128	171	340
Restaurant reviews	Training	800	632	2,159
	Testing	195	196	730
Tweets	Training	1,563	3,127	1,567
	Testing	174	346	174
News comments	Training	6,001	8,403	5,633
	Testing	838	1,054	732

Table 1: Details of the experimental datasets.

the general embeddings from (Pennington et al., 2014) for all datasets, so that the experimental results can better reveal the model's capability and the figures are directly comparable across different papers. The embeddings for Chinese experiments are trained with a corpus of 1.4 billion tokens with CBOW[3].

4.2 Compared Methods

We compare our proposed framework of Recurrent Attention on Memory (**RAM**) with the following methods:

- Average Context: There are two versions of this method. The first one, named **AC-S**, averages the word vectors before the target and the word vectors after the target separately. The second one, named **AC**, averages the word vectors of the full context.
- SVM (Kiritchenko et al., 2014): The traditional state-of-the-art method using SVMs on surface features, lexicon features and parsing features, which is the best team in SemEval 2014.
- Rec-NN (Dong et al., 2014): It firstly uses rules to transform the dependency tree and put the opinion target at the root, and then performs semantic composition with Recursive NNs for sentiment prediction.
- TD-LSTM (Tang et al., 2015): It uses a forward LSTM and a backward LSTM to abstract the information before and after the target. Finally, it takes the hidden states of LSTM at last time step to represent the context for prediction. We reproduce its results on the tweet dataset with our embeddings, and also run it for the other three datasets.
- TD-LSTM-A: We developed TD-LSTM to make it have one attention on the outputs of forward and backward LSTMs, respectively.
- MemNet (Tang et al., 2016): It applies attention multiple times on the word embeddings, and the last attention's output is fed to softmax for prediction, without combining the results of different attentions. We produce its results on all four datasets with the code released by the authors.[4]

For each method, the maximum number of training iterations is 100, and the model with the minimum training error is utilized for testing. We will discuss different settings of RAM later.

4.3 Main Results

The first evaluation metric is Accuracy, which is used in (Tang et al., 2016). Because the datasets have unbalanced classes as shown in Table 1, Macro-averaged F-measure is also reported, as did in (Dong et al., 2014; Tang et al., 2015). As shown by the results in Table 2, our RAM consistently outperforms all compared methods on these four datasets. AC and AC-S perform poorly, because averaging context is equivalent to paying identical attention to each word which would hide the true sentiment word. Rec-NN is better than TD-LSTM but not as good as our method. The advantage of Rec-NN is that it utilizes the result of dependency parsing which might shorten the distance between the opinion target and the related opinion word. However, dependency parsing is not guaranteed to work well on irregular texts such as tweets, which may still result in long path between the opinion word and its target, so that the opinion features would also be lost while being propagated. TD-LSTM performs less competitive than our method on all the datasets, particularly on the tweet dataset, because in this dataset sentiment words are usually far from person names,

[3] https://github.com/svn2github/word2vec

[4] http://ir.hit.edu.cn/∼dytang

Method	Laptop		Restaurant		Tweet		Comments	
	Acc	Macro-F1	Acc	Macro-F1	Acc	Macro-F1	Acc	Macro-F1
AC	0.6729	0.6186	0.7504	0.6396	0.6228	0.5912	0.6231	0.6182
AC-S	0.6839	0.6217	0.7585	0.6379	0.6329	0.6009	0.6425	0.6376
SVM	0.7049*	NA	0.8016*	NA	0.6340♮	0.6330♮	0.6524	0.6499
Rec-NN	NA	NA	NA	NA	0.6630*	0.6590*	NA	NA
TD-LSTM	0.7183	0.6843	0.7800	0.6673	0.6662	0.6401	0.7275	0.7260
TD-LSTM-A	0.7214	0.6745	0.7889	0.6901	0.6647	0.6404	0.7206	0.7195
MemNet	0.7033	0.6409	0.7816	0.6583	0.6850	0.6691	0.6247	0.6117
RAM	**0.7449**	**0.7135**	**0.8023**	**0.7080**	**0.6936**	**0.6730**	**0.7389**	**0.7385**

Table 2: Main results. The results with '*' are retrieved from the papers of compared methods, and those with '♮' are retrieved from Rec-NN paper.

No. of AL	Laptop	Restaurant	Tweet	Comments
RAM-1AL	0.7074	0.7996	0.6864	0.7336
RAM-2AL	**0.7465**	0.7889	0.6922	0.7363
RAM-3AL	0.7449	0.8023	**0.6936**	**0.7389**
RAM-4AL	0.7293	**0.8059**	0.6879	0.7325
RAM-5AL	0.7293	0.7960	0.6864	0.7325

Table 3: The impacts of attention layers. (Word embeddings are not tuned in the training stage.)

for which case the multiple-attention mechanism is designed to work. TD-LSTM-A also performs worse than our method, because its two attentions, i.e. one for the text before the target and the other for the after, cannot tackle some cases where more than one features being attended are at the same side of the target.

Our method steadily performs better than MemNet on all four datasets, particularly on the News comment dataset, its improvement is more than 10%. MemNet adopts multiple attentions in order to improve the attention results, given the assumption that the result of an attention at a later hop should be better than that at the beginning. MemNet doesn't combine the results of multiple attentions, and the vector fed to softmax is the result of the last attention, which is essentially the linear combination of word embeddings. As we described before, attending too many words in one time may hide the characteristic of each word, moreover, the sentiment transition usually combines features in a nonlinear way. Our model overcomes this shortcoming with a GRU network to combine the results of multiple attentions. The feature-based SVM, which needs labor-intensive feature engineering works and a mass of extra linguistic resources, doesn't display its advantage, because the features for aspect sentiment analysis cannot be extracted as easily as for sentence or document level sentiment analysis.

4.4 Effects of Attention Layers

One major setting that affects the performance of our model is the number of attention layers. We evaluate our framework with 1 to 5 attention layers, and the results are given in Table 3, where NAL means using N attentions. In general, our model with 2 or 3 attention layers works better, but the advantage is not always there for different datasets. For example, for the Restaurant dataset, our model with 4 attention layers performs the best. Using 1 attention is always not as good as using more, which shows that one-time attention might not be sufficient to capture the sentiment features in complicated cases. One the other hand, the performance is not monotonically increasing with respect to the number of attentions. RAM-4AL is generally not as good as RAM-3AL, it is because as the model's complexity increases, the model becomes more difficult to train and less generalizable.

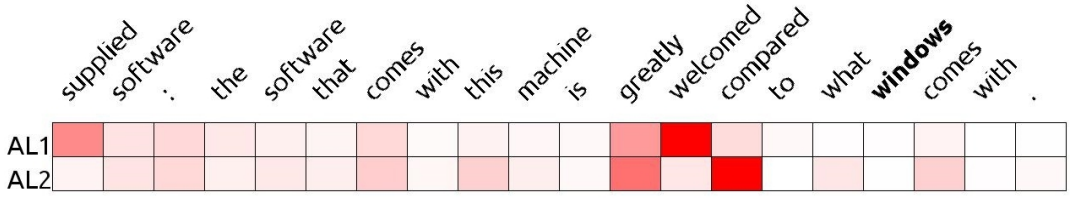
4.5 Effects of Embedding Tuning

The compared embedding tuning strategies are:

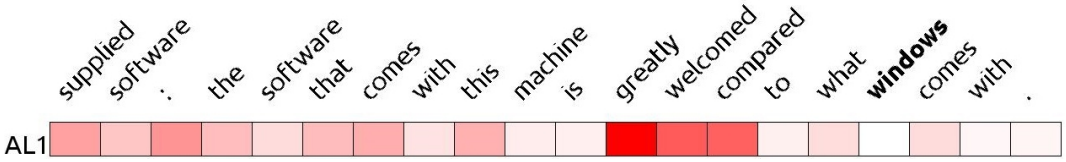
- RAM-3AL-T-R: It does not pre-train word embeddings, but initializes embeddings randomly and then tunes them in the supervised

Embedding	Laptop	Restaurant	Tweet	Comment
RAM-3AL-T-R	0.5806	0.7129	0.6272	0.6749
RAM-3AL-T	0.6854	0.7522	0.6402	0.7283
RAM-3AL-NT	**0.7449**	**0.8023**	**0.6936**	**0.7389**

Table 4: The impact of different embedding tuning strategies.



(a) Example of multiple attentions. The target is "windows".



(b) Example of single attention. The target is "windows".

Figure 2: Comparison of single attention and multiple attentions. Attention score by Eq. 13 is used as the color-coding.

training stage.

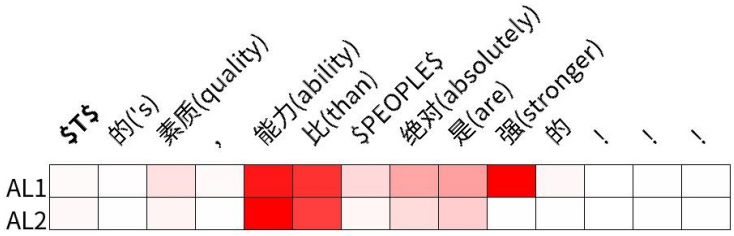
- RAM-3AL-T: Using the pre-trained embeddings initially, and they are also tuned in the training.
- RAM-3AL-NT: The pre-trained embeddings are not tuned in the training.

From Table 4, we can see that RAM-3AL-T-R performs very poorly, especially when the size of training data is smaller. The reason could be threefold: (1) The amount of labelled samples in the four experimental datasets is too small to tune reliable embeddings from scratch for the in-vocabulary words (i.e. existing in the training data); (2) A lot of out-of-vocabulary (OOV) words, i.e. absent from the training data, but exist in the testing data; (3) It increases the risk of overfitting after adding the embedding parameters to the solution space (it requires the embeddings not only to fit model parameters, but also to capture the similarity among words). During training, we indeed observed that the training error converges too fast in RAM-3AL-T-R. RAM-3AL-T can utilize the embedding similarity among words at the beginning of training, but fine tuning will destroy this similarity during training. On the other hand, the initial embeddings of OOV words in the testing data are not tuned, so that their similarity with vocabulary words are also destroyed. In addition, RAM-3AL-T also suffers from the risk of overfitting. RAM-3AL-NT performs the best on all four datasets, and we also observe that the training error converges gradually while the model parameters are being updated with the error signal from the output layer.

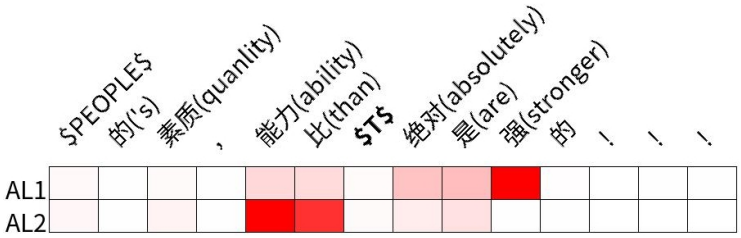
4.6 Case Study

We pick some testing examples from the datasets and visualize their attention results. To make the visualized results comprehensible, we remove the BLSTM memory module to make the attention module directly work on the word embeddings, thus we can check whether the attention results conform with our intuition. The visualization results are shown in Figures 2 and 3.

Figures 2a and 2b present the differences between using two attentions and using one attention, which show that multiple attentions are useful to attend correct features. As shown in Figure 2a, in order to identify the sentiment of "windows", the model firstly notices "welcomed" and secondly notices "compared" before the aspect target "windows". Finally it combines them with the GRU network, and generates a negative sentiment because the compared item (i.e. "windows") after a positive sentiment word (i.e. "welcomed") is less preferred. While the attention result of the model



(a) Example of a Chinese contrastive sentence, whose translation is "T's quality and ability are absolutely stronger than $PEOPLE$!!!". The target is "T".



(b) The sentence from 3a with a different target, i.e. "$PEOPLE$'s quality and ability are absolutely stronger than T!!!".

Figure 3: Example of multiple opinion targets. Attention score by Eq. 13 is used as the color-coding.

with only one attention, as shown in Figure 2a, is a sort of uniform distribution and mingles too many word vectors in a linear way, which would ruin the characteristic of each word.

Figures 3a and 3b present a case that there are more than one opinion targets in a comment, which cannot be analyzed with sentence-level sentiment analysis methods properly. Specifically, it's a comparative sentence in which the reviewer has a positive sentiment on the first commented person, but a negative sentiment on the second person, and our model predicts both of them correctly. Although all useful information (e.g. "than" and "stronger") is attended in both cases, the attention procedures of them show some interesting differences. They mainly attend important information after the target T in the first attention layer AL1. After that, Figure 3b attends more information before T in AL2. Since the same words in Figures 3a and 3b have different memory slices due to position weighting and augmented offset feature, as described in Section 3.3, our model predicts opposite sentiments on the two persons. For example in Figure 3b, the model first attends a positive word "stronger" and then attends "than" before the target, so it reverses the sentiment and finally predicts a negative sentiment.

5 Conclusions and Future Work

In this paper, we proposed a framework to identify the sentiment of opinion targets. The model first runs through the input to generate a memory, in the process of which it can synthesize the word sequence features. And then, the model pays multiple attentions on the memory to pick up important information to predict the final sentiment, by combining the features from different attentions non-linearly. We demonstrated the efficacy of our model on four datasets, and the results show that it can outperform the state-of-the-art methods.

Although multiple-attention mechanism has the potential to synthesize features in complicated sentences, enforcing the model to pay a fix number of attentions to the memory is unnatural and even sort of unreasonable for some cases. Therefore, we need a mechanism to stop the attention process automatically if no more useful information can be read from the memory. We may also try other memory weighting strategies to distinguish multiple targets in one comment more clearly.

References

Orestes Appel, Francisco Chiclana, Jenny Carter, and Hamido Fujita. 2016. A hybrid approach to the sentiment analysis problem at the sentence level. *Knowl.-Based Syst.*, 108:110–124.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *CoRR*, abs/1409.0473.

Peng Chen, Bing Xu, Muyun Yang, and Sheng Li. 2016. Clause sentiment identification based on convolutional neural network with context embedding.

In *Natural Computation, Fuzzy Systems and Knowledge Discovery (ICNC-FSKD), 2016 12th International Conference on*, pages 1532–1538.

KyungHyun Cho, Bart van Merrienboer, Dzmitry Bahdanau, and Yoshua Bengio. 2014. On the properties of neural machine translation: Encoder-decoder approaches. *CoRR*, abs/1409.1259.

Lingjia Deng and Janyce Wiebe. 2015. Joint prediction for entity/event-level sentiment analysis using probabilistic soft logic models. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*.

Xiaowen Ding, Bing Liu, and Philip S Yu. 2008. A holistic lexicon-based approach to opinion mining. In *Proceedings of the 2008 international conference on web search and data mining*, pages 231–240.

Li Dong, Furu Wei, Chuanqi Tan, Duyu Tang, Ming Zhou, and Ke Xu. 2014. Adaptive recursive neural network for target-dependent twitter sentiment classification. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*.

Murthy Ganapathibhotla and Bing Liu. 2008. Mining opinions in comparative sentences. In *Proceedings of the 22nd International Conference on Computational Linguistics-Volume 1*, pages 241–248.

Alex Graves, Greg Wayne, and Ivo Danihelka. 2014. Neural turing machines. *CoRR*, abs/1410.5401.

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*, pages 151–160.

Andrej Karpathy, Justin Johnson, and Fei-Fei Li. 2015. Visualizing and understanding recurrent networks. *CoRR*, abs/1506.02078.

Svetlana Kiritchenko, Xiaodan Zhu, Colin Cherry, and Saif Mohammad. 2014. Nrc-canada-2014: Detecting aspects and sentiment in customer reviews. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 437–442.

Ankit Kumar, Ozan Irsoy, Jonathan Su, James Bradbury, Robert English, Brian Pierce, Peter Ondruska, Ishaan Gulrajani, and Richard Socher. 2015. Ask me anything: Dynamic memory networks for natural language processing. *CoRR*, abs/1506.07285.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *CoRR*, abs/1301.3781.

Thien Hai Nguyen and Kiyoaki Shirai. 2015. Phrasernn: Phrase recursive neural network for aspect-based sentiment analysis. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543.

Maria Pontiki, Dimitris Galanis, John Pavlopoulos, Harris Papageorgiou, Ion Androutsopoulos, and Suresh Manandhar. 2014. Semeval-2014 task 4: Aspect based sentiment analysis. In *Proceedings of the 8th international workshop on semantic evaluation (SemEval 2014)*, pages 27–35.

Alexander M. Rush, Sumit Chopra, and Jason Weston. 2015. A neural attention model for abstractive sentence summarization. *CoRR*, abs/1509.00685.

Richard Socher, Christopher D Manning, and Andrew Y Ng. 2010. Learning continuous phrase representations and syntactic parsing with recursive neural networks. In *Proceedings of the NIPS-2010 Deep Learning and Unsupervised Feature Learning Workshop*, pages 1–9.

Richard Socher, Jeffrey Pennington, Eric H. Huang, Andrew Y. Ng, and Christopher D. Manning. 2011. Semi-supervised recursive autoencoders for predicting sentiment distributions. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing, EMNLP 2011*, pages 151–161.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*.

Duyu Tang, Bing Qin, Xiaocheng Feng, and Ting Liu. 2015. Target-dependent sentiment classification with long short term memory. *CoRR*, abs/1512.01100.

Duyu Tang, Bing Qin, and Ting Liu. 2016. Aspect level sentiment classification with deep memory network. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*.

Yequan Wang, Minlie Huang, xiaoyan zhu, and Li Zhao. 2016. Attention-based lstm for aspect-level sentiment classification. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*.

Wayne Xin Zhao, Jing Jiang, Hongfei Yan, and Xiaoming Li. 2010. Jointly modeling aspects and opinions with a maxent-lda hybrid. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, EMNLP '10, pages 56–65.