Neural Architecture for Negative Opinion Expressions Extraction

Hui Wen[1,2], Minglan Li[1,2], Zhili Ye[1,2]

[1] Institute of Software Chinese Academy of Sciences, Beijing 100190, China
[2] University of Chinese Academy of Sciences, Beijing 100049, China
{wenhui,minglan,zhili}2015@iscas.ac.cn

Abstract. Opinion expressions extraction is one of the main frameworks in opinion mining. Extracting negative opinions is more difficult than positive opinions because of indirect expressions. Especially, in the domain of consumer reviews, consumers are easier to be influenced by negative reviews when making decision. In this paper, we focus on the extraction of negative opinion expressions of consumer reviews. State-of-art methods heavily depend on task specific knowledge in the form of handcrafted features and data pre-processing. In this paper, we use a neural architecture by combining word embeddings, Bi-LSTM and CRF. We add a conditional random fields (CRF) layer to bidirectional long-short term memory (Bi-LSTM) recurrent neural network language model, which provides sentence level tag information and improves the result of experiment. Our model requires no feature engineering and outperforms feature dependent methods when experimenting on real-world reviews from Amazon.com.

Keywords: Neural Architecture, Opinion Extraction

1 Introduction

Opinion expressions extraction is one of the main frameworks in opinion mining. Existing works mainly focus on subjective expressions extraction and opinion target extraction [10]. Subjective expressions extraction is to distinguish the sentence or phrase is subjective or objective and extract subjective ones out. Opinion target extraction focus on finding the target terms implying sentiment in sentences, for example, "software" is the target term of the review "Updating with the latest software didn't help". However, what is important to us is the opinion implied by the target term not the term itself. Opinion expression is consisted by not only the target term but also the description of the term ("software didn't help" in the last example). Therefore, the task of extracting the whole opinion expressions or determining the opinion phrase boundary is important for further opinion mining tasks such as polarity classification and sentiment summarizations.

In the domain of consumer reviews opinion mining, precisely extracting opinion expressions is useful for further sentiment classification and summarization,which is very important for helping consumers to make desicions and helping merchant to improve the quality of their product. Also, reviews that imply negative sentiment is always much more difficult to analysis than positive ones because the obscure or indirect expression. In this work, we focus on the negative opinion expressions extraction in the domain of customer reviews from Amazon.com provided by Arjun[14].

Most previous works on opinion target extraction have used parsing, sematic, syntactic features, language patterns and task specific knowledge. The huge amount of features make the task complex and time-consuming. Conditional random fields (CRF)[8] and other improved methods based on CRF are the most popular methods for this task. However, CRF is a linear model and heavily depend on handcrafted features. Therefore, the effectiveness of previous work is limited by the selection of features and the grammatical accuracy of sentences in the dataset.

In recent years, deep learning methods have been widely studied on natrual language processing tasks. Mikolov et al.[13] presents word embeddings training with neural network which becomes effective representations of words and phrases. Recurrent neural network[12] becomes a new effective way for building language model compared with probabilistic methods. Also, long short-term memory[5] cell makes it possible for recurrent neural network to learn long sequence.

For more complex natural language processing tasks, such as named entity recognition and part-of-speach tagging, deep learning models have also outperformed feature-based methods. Neural architecture is famous for its non-linear character and can automatically learn the semantic and syntactic information of the sentences, which remedy the limitation of CRF.

To this end, we use a neural architecture by combining pre-trained word embeddings, Bi-LSTM and CRF. Pre-trained word embeddings provide word presentations learnt from huge amount of data, which also contains the context information. Bidirectional long-short term memory (Bi-LSTM) recurrent neural network language model provides non-linear character in modeling sequential data. Linear conditional random fields provides sentence level tag information.

We take the task as a sequence labeling task and use deep architecture to solve the task of opinion expressions extraction (phrase boundary detection).The main contribution of our work are summarized as follows:

– We solve the negative opinion expressions extraction task with a neural architecture with no need of feature engineering and outperform feature-based methods in real-word consumer review data.
– We prove that adding a linear CRF layer which provide sentence level tag information to neural network language model can significantly improve the result of the task of opinion expressions extraction.

The rest of this paper is organized as follows: Section 2 formulates the problem and describes the model. Section 3 discusses the experimental setup,training and evaluation of the network. Section 4 reviews the related work. Section 5 concludes the paper and presents our future work.

2 Neural Network Architecture

In this Section we formulate the problem of opinion expressions extraction, describe our neural network architecture components from bottom to up and describe the algorithm to apply the neural architecture to negative opinion expressions extraction.

2.1 Problem Statement

The opinion expressions extraction task can be formally described as follows: Given a sentence, $s = (w_1, w_2, ..., w_n)$, find sub-sequence of the sentence which contains target of opinion and opinion expression, $o = (w_p, ..., w_i, ..., w_q)$ $(p \geq 1, q \leq n)$. The aim is to correctly decide the phrase boundary of the sentence.

We take the task as a sequence labeling task. We represents the sentence into the BIO format (Beginning of opinion expressions, Inside of opinion expressions, End of opinion expressions). For a word in sentence, we label the word B if the word is the first word in opinion expressions, we label the word I if the word is inside but not in the first position of the opinion expression, and for other words which is not in the opinion expression, we label it O. For example, in the sentence below, the opinion expression is inside [[]], we label the sentence as follow:

Updating(O) with(O) the(O) latest(O) [[software(B) didn't(I) help(I)]].

The task is that given a sentence ,finding the right place for the position of BIO, which means correctly predicting every word in the sentence to be B, I or O.

2.2 Word Embeddings

Word embeddings map words to vectors using methods including probabilistic models, neural networks and so on. Word embeddings have exhibit remarkable boost in many natural language processing (NLP) tasks with the property of being able to encode the analogy between words. Since Bengio[1] et al. propose neural language and Collobert and Weston[3]train word embeddings on a large dataset and show effectiveness of word embddings. Since that, word embeddings have been used in NLP tasks in wide range.

In NLP task using neural architecture, word embedding can be used in two ways: co-trained with the task and pre-trained based on larger dataset. In our work, we use Glove pre-trained 200-dimensional word embeddings trained on 2 billions twiteers Twitter.[17] Also, in Section 3.6 we discuss the impact of word embeddings on our task using the Stanford Glove[17] two different pre-trained word embeddings trained Wikipedia and Twitter respectively with different dimensions and the 300-dimentional word embeddings proposed by Mikolov et al.[13] trained on 100 billion words from Google News.

2.3 Bi-LSTM RNN Language Model

Recurrent Neural Network (RNN) is a neural architecture feasible for dealing with sequence data, because it makes use of information and performs the same task for every element of a sequence. RNN language model has been proved to be more effective and less computational complexity compared with traditional language model by Mikolov et al [12].

However, RNN suffers vanishing/exploding gradients when learning long sequence. Long Short-term Memory (LSTM) [5] solves the problem of RNN with the addition of a memory cell and uses input gate i_t, output gate o_t, forget gate f_t to control the proportion of the input and the previous state to the memory. The LSTM outputs memory cell c_t, hidden state h_t at each time step is shown as follows:

$$\begin{aligned}
i_t &= \sigma(W^i x_t + U^i h_{t-1} + b^i) \\
f_t &= \sigma(W^f x_t + U^f h_{t-1} + b^f) \\
o_t &= \sigma(W^o x_t + U^o h_{t-1} + b^o) \\
g_t &= tanh(W^g x_t + U^g h_{t-1} + b^g) \\
c_t &= f_t \odot c_{t-1} + i_t \odot g_t \\
h_t &= o_t \odot tanh(c_t)
\end{aligned} \tag{1}$$

where σ is the element-wise sigmoid and hyperbolic tangent functions, $\odot$ is the element-wise multiplication operator.

Bi-directional LSTM,[4]composed of a forward LSTM and a backward LSTM, can capture both the past and future information respectively. The feature of Bi-LSTM makes it effective in many NLP tasks, because it takes the advantage of contextual information.

2.4 CRF

Conditional random fields(CRF)[8] is used in wide range for natural language processing sequence labeling tasks such as part-of-speech tagging, name entity recognition. CRF takes context into account, using tagging information at sentence level to model tagging decisions jointly

For a sequence labeling task, CRF gives a score for labeling result of a sentence as follows:

$$score(y|x) = \sum_{j=1}^{m} \sum_{i=1}^{n} \lambda_j f_j(y_i, y_{i-1}, x) \tag{2}$$

here $f_j$is feature fuction and $\lambda_j$is its weight. Each feature function is composed with the label of current word y_i and the label of the previous word y_{i-1} .That means, feature functions in CRF depend on current and previous label rather than arbitrary word. $\Lambda = \{\lambda_k\}$ is the weights of feature fuctions. In CRF, normalize the score into probability$p(y|x,\Lambda)$ with the normalization as follows:

$$Z(x,\Lambda) = \sum_{y}(exp(\sum_{j=1}^{m}\sum_{i=1}^{n}\lambda_j f_j(y_i, y_{i-1}, x))) \tag{3}$$

$$p(y|x,\Lambda) = \frac{exp(\sum_{j=1}^{m}\sum_{i=1}^{n}\lambda_j f_j(y_i, y_{i-1}, x))}{Z(x,\Lambda)} \tag{4}$$

During training,we minimize the negative log-probablity of the correct label.

$$\Lambda = argmin_{\Lambda}(-\sum_{j} log(p(y_i|x_i,\Lambda))) \tag{5}$$

where is (x_i, y_i) is in the training examples. Once we have trained a CRF model, given a new sentence, we can get its labeling sequence using dynamic programming, such as Viterbi Decoding.

2.5 Bi-LSTM-CRF

In our model, we combine Bi-LSTM with CRF similar with architecture in [9][11]. First, we feed word embeddings of sentence into bidirectional long-short term memory (Bi-LSTM) recurrent neural network language model. Then the output is fed into the conditional random fields (CRF) layer which take into account neighboring tags, yielding the final predictions for every word. The combination of no-linear and linear model improves the result of our experiment ,which will be discussed in Section 3.5. The architecture is shown in Fig 1.

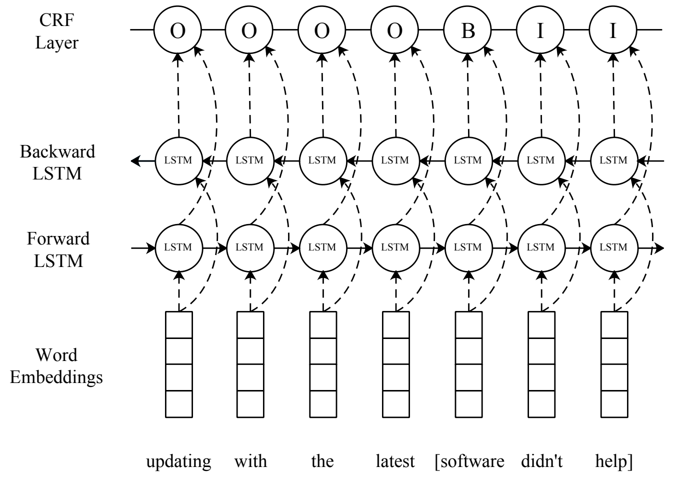

Fig. 1: The Architecture of Bi-LSTM-CRF

2.6 Apply the Neural Architecture to Negative Opinion Expressions Extraction

In this part,we will discuss the technological process to apply the neural architecture to negative opinion exprssions extraction. Given a set of real-world negative reviews from Amazon.com. The only pre-processing for the data is simply tokenize the words with blank character and padding the sentences as discussed in Section 3.3. Then we present the word with pre-trained word embeddings, and feed them into our neural architecture. Then we apply backpropagation algorithm to minimize the loss function and upgrade the parameters. The flow of our algorithm is as follows:

Table 1: The Flow for Opinion Expressions Extraction

```
Input: A set of consumer reviews $S = \{s_1, s_2, ...s_n\}$
Output: The parameter of Bi-LSTM-CRF
Parameters Initialization:Intialize the matrix and bias with a normal distribution with mean equals 0 and standard deviation equals 0.1.
Padding and initial reviews using pre-trained word embeddings.
for each review $s_i \in S$ do:
    1 Feed $x_i$, the vector prestantion of $s_i$ to Bi-LSTM-CRF and generate pridicted sequence label $y_i$.
    2 Compute the loss of $y_i$.
    3 Use backpropagation algorithm to update the parameters of the network.
end for
```

3 Experimental Setup

In this section, we describe the experimental setup of our paper, including the way we train our neural architecture and the efficiency of our model in detail.

3.1 Datasets

Most of the datasets in domain of opinion mining focus on the sentiment classification. For the task of opinion expressions extraction, the sentiment analysis dataset developed by Qiu [22] et al. and the SemEval2014 dataset [18] focus on the opinion target term extraction. The dataset provided by Wang et al. [26] expand the SemEval2014 dataset by adding manually labeled opinion terms. The dataset developed by Arjun[14] is for the task of opinion expressions boundary detection.

We use the dataset developed by Arjun [14], which provides consumer reviews from Amazon.com. The dataset contains six domains data, the number of negative and total sentences are as below: Router (1284/5063), GPS (632/2075), Keyboard (667/1446), Mouse (494/2488), MP3-Player (174/352), Earphone (359/678).

In our model, we require the recurrent network language model with long-short term memory to learn the long term context and dependencies between words in the input sentences. We need to deal with the variation of the length of sentences, we need to find a feasible max sentence length so that many of the sentences would share the same length and the length won't be too long to make the model too complex. We will describe the distribution of the sentence lengths of our dataset and the way of padding in detail in the Section 3.3.

3.2 Comparative Approaches

We compare our model with four feature based methods described in[14].

UHB: A rule based method by finding the constitution around the opinion term, such as a positive sentiment and a negator. And decide the window to find sentiment negator by experiment.

CRF: The traditional Markov linear-chain conditional random fields (CRF). The features contains two classes: Pivot Features (POS Tags, Phrase Chunk Tags, Prefixes, Suffixes, Word Sentiment Polarity), Latent Semantics.

CRF-L2R: The traditional Markov linear-chain conditional random fields (CRF) with regularization on the feature weights. During training, the log-likelihood is modified as follows:

$$\Lambda = argmin_{\Lambda}(-\sum_{j} log(p(y_i|x_i, \Lambda))) + \sum_{k} \lambda_k^2 \tag{6}$$

CRF-PSC: A constrained model which summing over only the valid phrases produced by CRF.In traditional CRF, we $p(y|x, \Lambda)$ with the normalization $Z(x, \Lambda)$, which summing over all possible sequence labeling. In CRF-PSC ,the normalization sums over only the valid sequence labeling.

3.3 Network Training

In this part, we will discuss the details about training the neural network in our work.

We implement Tensorflow library in our experiment. The details contain about parameter initialization, the sequence length for recurrent neural network language model, loss function with padding in sequence, optimization algorithm and ways to avoid overfitting.

Parameter Initialization: To verify the effectiveness of our neural network in simple pre-processing of data, the only data pre-processing is basically split the sentence into a list of words, without removing punctuations or stop words. We use Glove pre-trained 200-dimensional word embeddings trained on 2 billions twiteers.[17]. For the words which are not in the pre-trained words set, we random them uniformly distributed over the half-open interval $[-0.1, 0.1)$ so that any value within the given interval is equally likely to be drawn.

Matrix parameters and bias is used in two circumstances: (1)The softmax layer between Bi-LSTM and the output of the sequence labeling model in the contrast method of no adding a CRF layer,(2) The activation function between Bi-LSTM and the CRF layer. Matrix parameters and bias are randomly set both with a normal distribution with mean equals 0 and standard deviation equals 0.1.

Sequence length for RNN Language model: During the training of recurrent neural network, we have to feed the network the same length data in a training batch. However, the sentences in our dataset is not in the same length. Padding is the most popular ways to deal with this problem. We will pad the sentences with zero vectors to fill up the remaining part, so that in a training batch all the sentences share the same length.

The max length of padded data is one of the most important hyper-parameters in RNN language model. Too long sequence length would not help due to gradient vanish, even LSTM suffer from this problem. We analysis the distribution of the sentence lengths in our data. As we want to find the opinion expressions, we must keep them in our data. So we analysis the length of sentence after truncating at the end of opinion expressions as Figure 2. We choose the max sequence length as 50, so that many of our sentences would share the same length. We randomly keep part of sentences longer than that with no break of opinion expressions and do padding for shorter ones.

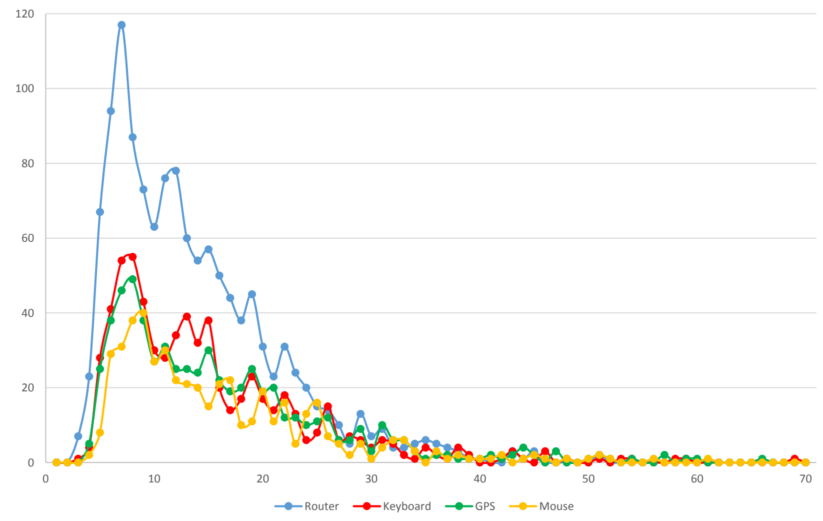

Fig. 2: The Sentence Length Distribution of Our Dataset

Loss Function with Padding in Sequence: The loss function need to be calculate accordingly after the data being padded. We mask the sequence with the actual length when training. So that during training, the padded part wouldn not been considered into loss function and not been trained.

Optimization Algorithm: We choose the mini-batch stochastic gradient descent (SGD) with momentum 0.9. Momentum[21] is a method to help avoid stuck into local minima when applying SGD. As the size of dataset is not large enough, we finally set the batch size at 1 after experiment on different batch size, which becomes standard SGD, but the time for training is still acceptable and the result is the best. We also use Adam[7] optimization algorithm when choosing the hidden size of LSTM preliminarily, because Adam has a fast convergence velocity which helps us tune our network more quickly. The hidden size for each domain is Router(64),Keyboard(30),GPS(40),Mouse(24).

We set the learning rate at 0.001 and early stop[2] based on performance on validation sets, all the results are based on 4-fold cross validation.To avoid gradient exploding, we also use gradient clipping method[16]. Also, we shuffle the data randomly before training.

Ways to avoid overfitting: In our work, we use two ways to avoid overfitting :dropout and L2-regulazition on matrix weights and bias. Dropout[23] is an effective method to deal with overfitting by randomly drop units and their connections to avoid co-adapting of units. When some neurons are randomly dropped out during training, the co-adapting between them is avoided. This makes the network have better generalization performance. We use the dropout at the input of the Bi-LSTM and put a dropout wrapper on both forward and backward cell of Bi-LSTM. We set the dropout rate at 0.6, which helps us to get the best performance on our dataset. We will further discuss the effect of dropout in Section 3.6.

L2-regularization term to the weights and bias is also one of the frequently-used method to avoid overfitting. The L2-regularization term gives penalty on the large values of parameters of neural network which increases the generality of neural network. During our training, we take our loss as a sum of negative log-likelihood of sequence labeling result(NLL) and L2-regularization term of parameters as follows:

$$Loss = NLL + \beta_1 R(weights) + \beta_2 R(bias) \tag{7}$$

In our experiment,we take $\beta_1 = \beta_2 = 0.001$, the weights and bias is discussed before in parameter initialization.

3.4 Evaluation Method

We use the same overlap matrix described in [14]. The set of sentences in test dataset is S. For each sentence $s \in S$, recall(r), precision(p) and F1 value(F_1)

as follows:

$$\begin{aligned} p &= avg_{s\in S, |s_c|\neq 0}\left(\frac{|s_c|\bigcap|s_p|}{|s_c|}\right) \\ r &= avg_{s\in S, |s_p|\neq 0}\left(\frac{|s_c|\bigcap|s_p|}{|s_p|}\right) \\ f_1 &= \frac{2pr}{p+r} \end{aligned} \tag{8}$$

where s_c, s_p denote the correct and predicted opinion expressions spans in sentence.

3.5 Model Effectiveness

In this part, we present the performance of our model. We verify the effectiveness of our model compared with feature-based methods described in Section 3.2.

Compared with feature-based methods[14]: Table 2 shows the precision, recall, f1-value on the dataset described in Section 3.1 with four domains (Router, Keyboard, GPS, Mouse). The recall and f1-value has been improved using our neural architecture. Noting we used the same architecture and no feature engineering to get the improvement on different domains of data. In the keyboard domain,however,we don't get result good enough, by ananlysis the raw data,we find there are too many numbers and html links.To show the advantage of our model, we have not done preprocessing to these circumenstances.

Table 2: Performance of four types datasets

Dataset	Method	P	R	F1	Dataset	Method	P	R	F1
Router	UHB	67.0	73.7	70.2	Keyboard	UHB	64.4	68.1	66.0
	CRF	87.9	76.9	82.0		CRF	86.8	74.8	80.3
	CRF-L2R	88.7	77.1	82.5		CRF-L2R	87.6	75.7	81.2
	CRF-PSC	92.0	80.1	85.7		CRF-PSC	90.4	78.3	83.9
	Bi-LSTM	87.0	81.3	84.0		Bi-LSTM	76.3	74.0	75.1
	Bi-LSTM+CRF	87.6	87.8	**87.7**		Bi-LSTM+CRF	80.1	82.6	81.3
GPS	UHB	67.5	67.7	67.6	Mouse	UHB	60.7	59.1	59.8
	CRF	84.1	71.9	77.5		CRF	83.8	61.6	70.9
	CRF-L2R	84.8	72.7	78.3		CRF-L2R	84.5	61.9	71.4
	CRF-PSC	86.7	73.5	79.5		CRF-PSC	86.1	63.6	73.1
	Bi-LSTM	81.2	77.6	79.4		Bi-LSTM	84.0	75.9	79.7
	Bi-LSTM+CRF	85.0	84.3	**84.7**		Bi-LSTM+CRF	80.1	81.2	**80.6**

Also, In table 2, we show the result of adding a CRF layer. Compared with feed the output of Bi-LSTM to Softmax, the CRF layer improve the result significantly. Adding a CRF layer has improved the recall a lot and thus improve the F1-value.

3.6 Impact of Dropout and Word Embeddings

In this part, we discuss about the impact of dropout and pre-trained word embeddings on our experiment.

Dropout: Deep neural networks always suffer from ovefitting because of large number of parameters, especially when dataset for training is not large enough. Dropout is a powerful method solving this problem [6]. In our experiment, the dataset is kind of small of neural network, so dropout is extremely necessary for the performance. We use dropout on the input and Bi-LSTM layer when training, the effect of Dropout is shown as Table 3, the parameter is the same as Section 3.5. As we can see from the figure, dropout is effective in improving the performance of our neural architecture model in different domains of data.

Table 3: Impact of Dropout on Four Types datasets

	Router			Keyboard			Gps			Mouse		
	P	R	F1	P	R	F1	P	R	F1	P	R	F1
Dropout	87.6	87.9	87.7	80.1	82.6	81.3	85.0	84.3	84.7	80.1	81.2	80.6
No Dropout	84.1	83.3	83.7	70.1	73.0	71.5	75.3	78.0	76.6	77.2	79.4	78.3

Word embeddings: In order to find the effect of pre-trained word embeddings on our model, we performed experiments with three different pre-trained word embeddings on the dataset in Router domain, the F1-value using different pre-trained word embeddings are shown in Table 4.

(1) Glove, trianed on Eikipedia 2014 and Gigaword 5(6 billions tokens, 400K vocabulary) with dimensions of 50,100,200,300 [17].

(2) Glove Twitter,trained on Twitter(2 billions tweets, 12M vocabulary) with dimensions of 25,50,100.200[17].

(3) Word2Vec, trained on 100 billion words from Google News with dimensions of 300 produced by Mikolov et al.[13].

As we can see from the figure, different pre-trained word embeddings effect the performance of our model. The choice of pre-trained word embeddings is important when the training dataset is not large enough to co-train the word embeddings while solving the task. Using word embeddings pre-trained on larger dataset is helpful to get better performance.

Table 4: Different Pre-trained Word Embeddings

	25D	50D	100D	200D	300D
Glove	-	80.1	83.1	81.8	83.2
Glove Twitter	83.6	86.1	84.6	87.7	-
Word2Vec	-	-	-	-	87.7

4 Related Work

Opinion expressions extraction is a major task in opinion mining. Kobayashi et al.[6] take the opinion expressions extraction as a frame consisting of: (1)Opinion Holder: the person making the evaluation (2) Target: a named entity belonging to a class of interest (e.g., iPhone) (3)Aspect: a part, member or related object, or attribute of the Subject (Target) (e.g., size, cost) (4)Evaluation: a phrase expressing an evaluation or the opinion holders mental/emotional attitude (e.g., too bulky). Opinion extraction task means filling these slots for each evaluation expressed in text.

Opinion target extraction is first studied by Hu et al.[6], they classify the opinion target into two kinds: explicit and implicit, but only deal with the explicit target with rule-based method. Popescu et al. [19] asume the class of the product is already known and use the mutual information between words and the product class to find the target words. Toh[27] et al. use CRF as a sequence labeler with features such as POS tags and WordNet taxonomies. In SemEval 2014[18] task 4, a dataset of two domains(Laptop and Restaurant) is provided for target extraction and polarity classification, many methods based on task-specific knowledge, sematic and syntactic structure have been proposed. Also, there are many methods based on topic model[15]. Neural architecture in recent years has shown improvement in this task, Soujanya[20] et al. use a deep CNN combining with rules and get the state-of-art result on the dataset of SemEval 2014.

Recent years, the co-extraction of opinion target and opinion evaluation terms has been a promising task. Qiu et al.[22] use rules and relations between them. Wang[26] et al. use recursive neural network combining conditional random fields and proposed an extended dataset for this task based on the dataset of SemEval 2014.

The task in our work is find the phrase boundaries of opinion expressions, which is similar to opinion target extraction but different from it from getting the full opinion expressions of aspect and evaluation. Arjun[14] is the first to focus on this task, and proposes the dataset for the task of opinion expressions extraction.

In recent years, neural architecture has shown significant improvement on natural language processing tasks. Mikolov et al.[13] presents the distributed representations of words and phrases with neural network, which is known as word embeddings. Pennington et al.[17] make the training process of word embedding possible on large dataset. And the recurrent neural network[12] has shown effectiveness in modeling natural language with long shor-term memory[5] cell to solve the problem of learning long sequence.

On the sequence information labeling and analysis tasks, methods with improvement based on Hidden Markov Model (HMM) and LDA haven been proved effective[24][25]. Lample[9] established two neural networks, one is a bidirectional LSTM with a sequential conditional random layer above it, the other is a new model that a new model that constructs and labels chunks of input sentences using an algorithm inspired by transition-based parsing with states represented by stack LSTMs. The model presents state-of-art name entity recognition task in different languages. Ma[11] et al. propose a neural architecture with the combination of CNN character level word embeddings , LSTM and CRF on the end-to-end sequence labeling tasks. They experiment on the two major sequence labeling tasks: part-of-speech tagging and name entity recognition, on both of tasks the model proves effectiveness over traditional feature-based methods.

Different from the target terms extraction discussed above, we focus on the task of phrase boundaries detection. And different from the model[11][9] discussed above, we don't use the CNN character level word embeddings or stack LSTM. Our model outperforms the previous feature-based methods such as feature-based CRF or rule-based methods.

5 Conclusion and future work

In this paper, we use a neural architecture by combining word embeddings, bidirectional long-short term memory (Bi-LSTM) recurrent neural network language model and conditional random fields (CRF) to solve the task of opinion expressions extraction. Our model has no need to make handcrafted features and outperforms the feature-based methods on real-world negative consumer reviews. Also, our work shows that adding a CRF layer to Bi-LSTM can significantly improve the result as import the sentence level tagging information. We also study about the methods of prevent overfitting when the dataset is not large enough and the influence of the size and corpus of the pre-trained word embeddings.

There are many directions for our future work: First, we can bring in finer granularity information to the model such as character level information of the words in sentence. Another direction is to combine our model with further task of opinion summarization and make a joint model for opinion extraction and summarization. Also, as our model has no need of task specific knowledge or handcrafted features, we can use it in other domain, such as find viewpoint in news, standpoint in academic papers and so on.

References

1. Bengio, Y., Ducharme, R., Vincent, P., Jauvin, C.: A neural probabilistic language model. Journal of machine learning research 3(Feb), 1137–1155 (2003)
2. Caruana, R., Lawrence, S., Giles, L.: Overfitting in neural nets: Backpropagation, conjugate gradient, and early stopping. In: NIPS (2000)
3. Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., Kuksa, P.: Natural language processing (almost) from scratch. Journal of Machine Learning Research 12(Aug), 2493–2537 (2011)
4. Graves, A., Schmidhuber, J.: Framewise phoneme classification with bidirectional lstm and other neural network architectures. Neural Networks 18(5), 602–610 (2005)
5. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural computation 9(8), 1735–1780 (1997)

6. Hu, M., Liu, B.: Mining and summarizing customer reviews. In: Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining. ACM (2004)
7. Kingma, D., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
8. Lafferty, J., McCallum, A., Pereira, F., et al.: Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In: Proceedings of the eighteenth international conference on machine learning, ICML. vol. 1 (2001)
9. Lample, G., Ballesteros, M., Subramanian, S., Kawakami, K., Dyer, C.: Neural architectures for named entity recognition. arXiv preprint arXiv:1603.01360 (2016)
10. Liu, B., Zhang, L.: A survey of opinion mining and sentiment analysis. In: Mining text data. Springer (2012)
11. Ma, X., Hovy, E.: End-to-end sequence labeling via bi-directional lstm-cnns-crf. arXiv preprint arXiv:1603.01354 (2016)
12. Mikolov, T., Karafiát, M., Burget, L., Cernockỳ, J., Khudanpur, S.: Recurrent neural network based language model. In: Interspeech. vol. 2, p. 3 (2010)
13. Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: Advances in neural information processing systems (2013)
14. Mukherjee, A.: Extracting aspect specific sentiment expres-sions implying negative opinions. In: In proceedings of the 17th International Conference on Intelligent Text Processing and Computational Linguistics (2016)
15. Mukherjee, A., Liu, B.: Aspect extraction through semi-supervised modeling. In: Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1. Association for Computational Linguistics (2012)
16. Pascanu, R., Mikolov, T., Bengio, Y.: On the difficulty of training recurrent neural networks. ICML (3) 28, 1310–1318 (2013)
17. Pennington, J., Socher, R., Manning, C.D.: Glove: Global vectors for word representation. In: EMNLP. vol. 14 (2014)
18. Pontiki, M., Galanis, D., Pavlopoulos, J., Papageorgiou, H., Androutsopoulos, I., Manandhar, S.: Semeval-2014 task 4: Aspect based sentiment analysis. Proceedings of SemEval (2014)
19. Popescu, A.M., Etzioni, O.: Extracting product features and opinions from reviews. In: Natural language processing and text mining. Springer (2007)
20. Poria, S., Cambria, E., Gelbukh, A.: Aspect extraction for opinion mining with a deep convolutional neural network. Knowledge-Based Systems 108, 42–49 (2016)
21. Qian, N.: On the momentum term in gradient descent learning algorithms. Neural networks 12(1), 145–151 (1999)
22. Qiu, G., Liu, B., Bu, J., Chen, C.: Opinion word expansion and target extraction through double propagation. Computational linguistics 37(1), 9–27 (2011)
23. Srivastava, N., Hinton, G.E., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: Dropout: a simple way to prevent neural networks from overfitting. Journal of Machine Learning Research 15(1), 1929–1958 (2014)
24. Su, B., Ding, X.: Linear sequence discriminant analysis: a model-based dimensionality reduction method for vector sequences. In: ICCV (2013)
25. Su, B., Ding, X., Liu, C., Wu, Y.: Heteroscedastic max-min distance analysis. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2015)
26. Wang, W., Pan, S.J., Dahlmeier, D., Xiao, X.: Recursive neural conditional random fields for aspect-based sentiment analysis. arXiv preprint arXiv:1603.06679 (2016)

27. Zhiqiang, T., Wenting, W.: Dlirec: Aspect term extraction and term polarity classification system (2014)