SCIENCE CHINA
Information Sciences

• RESEARCH PAPER •
Special Focus on Natural Language Processing and Social Computing

November 2017, Vol. 60 110101:1–110101:12
doi: 10.1007/s11432-016-9229-y

Encoding syntactic representations with a neural network for sentiment collocation extraction

Yanyan ZHAO[1], Bing QIN[2*] & Ting LIU[2]

[1]*Department of Media Technology and Art, Harbin Institute of Technology, Harbin* 150001*, China;*
[2]*Department of Computer Science and Technology, Harbin Institute of Technology, Harbin* 150001*, China*

Received January 24, 2017; accepted May 8, 2017; published online October 16, 2017

Abstract Sentiment collocation refers to the collocation of a target word and a polarity word. Sentiment collocation extraction aims to extract the targets and their modifying polarity words by analyzing the relationships between them. This can be regarded as a basic sentiment analysis task and is relevant in many practical applications. Previous studies relied mainly on the syntactic path, which is used to connect the target word and the polarity word. To deeply exploit the semantic information of the syntactic path, we propose two types of syntactic representation, namely, relation embedding and subtree embedding, to capture the latent semantic features. Relation embedding is used to represent the latent semantics between targets and their corresponding polarity words, and subtree embedding is used to explore the rich syntactic information for each word on the path. To combine the two types of syntactic representations, a neural network is constructed. We use a recursive neural network (RNN) to model the subtree embeddings, and then the subtree embedding and the word embedding are combined as the enhanced word representation for each word in the syntactic path. Finally, a convolutional neural network (CNN) is adopted to integrate the two types of syntactic representations to extract the sentiment collocations from reviews. Our experiments were conducted on six types of reviews, which included product domains (such as cameras and phones) and service domains (such as hotels and restaurants). The experimental results show that our proposed method can accurately capture the latent semantic features hidden behind the syntactic paths that neither the common feature-based methods nor the syntactic-path-based method can handle, and, further, that it significantly outperforms numerous baselines and previous methods.

Keywords sentiment collocation extraction, sentiment analysis, syntactic representation, neural network, recursive neural network (RNN), convolutional neural network (CNN)

Citation Zhao Y Y, Qin B, Liu T. Encoding syntactic representations with a neural network for sentiment collocation extraction. Sci China Inf Sci, 2017, 60(11): 110101, doi: 10.1007/s11432-016-9229-y

1 Introduction

Sentiment analysis deals with the computational treatment of opinion, sentiment, and subjectivity in text [1], and has received considerable attention in recent years [2]. Target-Polarity word (T-P) collocation extraction, which aims to extract the collocation of a target and its corresponding polarity word in a sentiment sentence, is a basic task in sentiment analysis. For example, in the sentiment sentence "这款相机拥有新颖的外形" (The camera has a novel appearance), "外形" (appearance) is the target,

* Corresponding author (email: bqin@ir.hit.edu.cn)

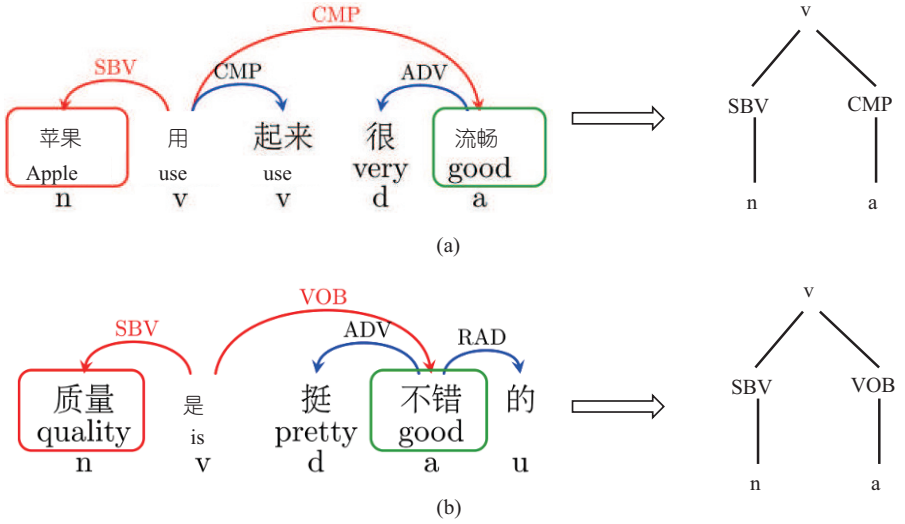

Figure 1 (Color online) Two syntactic parse trees that contain sentiment collocations. For each syntactic parse tree, the target is shown in a red box, and the polarity word is shown in a green box. The syntactic relation is depicted by the syntactic path in red between the target and the polarity word. (a) Syntactic structure 1; (b) syntactic structure 2.

and "新颖" (novel) is the polarity word that modifies "外形" (appearance). Accordingly, 〈 外形, 新颖 〉 (⟨appearance, novel⟩) is the sentiment collocation. Generally, a sentiment collocation is a basic and complete sentiment unit and thus is very useful for many sentiment analysis applications.

Features obtained from the syntactic parse trees have proved particularly useful for sentiment collocation extraction [3,4]. The syntactic path that connects the polarity word and the target word can better describe the relationship between the two elements. Thus, it is the most commonly used feature. For example, the syntactic path "Adj $\overset{\text{ATT}}{\curvearrowleft}$ Noun", where "ATT" denotes an attributive syntactic relation, can be used as important evidence in extracting the sentiment collocation ⟨propose two types of syntactic⟩ (⟨appearance, novel⟩) in the above sentiment-sentence [5–8].

Actually, the syntactic path can convey considerable information, which is always shown as the syntactic and semantic features. However, this kind of feature is always treated as a whole in the learning or matching process. Thus, the current syntactic-path-based approaches lack generalization capability and tend to result in two kinds of data sparsity problems. One is the lack of syntactic relation generalization. For example, we can use the syntactic path "n ← SBV ← v → CMP → a" to obtain the sentiment collocation 〈 苹果, 流畅 〉 (⟨Apple, good⟩), which is shown in Figure 1(a). However, syntactic features defined in this way cannot capture all useful syntactic information provided by the parse trees for sentiment collocation extraction. In another sentiment sentence, shown in Figure 1(b), the sentiment collocation 〈 质量, 不错 〉 (⟨quality, good⟩) cannot be obtained. This is because no syntactic paths can be matched, although the syntactic path between the target "质量" (quality) and the polarity word "不错" (good) in Figure 1(b) is "n ← SBV ← v → VOB → a", which is similar to the syntactic path "n ← SBV ← v → CMP → a". That is to say, the previous work treated the syntactic path along with the words on it as a whole and did not consider any sub-structured information; therefore, it failed to handle similar but non-identical syntactic structures. Further more, it is hard for the flat representation of the syntactic path to describe the explicit structured syntactic information.

The other problem in the previous studies is that they did not generalize, or only simply generalized, the word semantic representations. For instance, in one typical previous study [9] word embedding was adopted to represent each word in the syntactic path. This method can generalize the word information in a simple way. However, for each word in the syntactic path, it can also connect many other words in the sentence via the syntactic subtrees. Thus, we can explore additional useful information available from the syntactic subtrees to enhance the word representation on the path. For example, in Figure 2, for the target word "苹果 (apple)" in the sentiment sentence "美国 (US) 制造 (made) 的 (of) 苹果 (apple) 用 (use) 起来 (use) 很 (very) 流畅 (smooth)", we can use the syntactic relations (such as "ATT", "SBV",

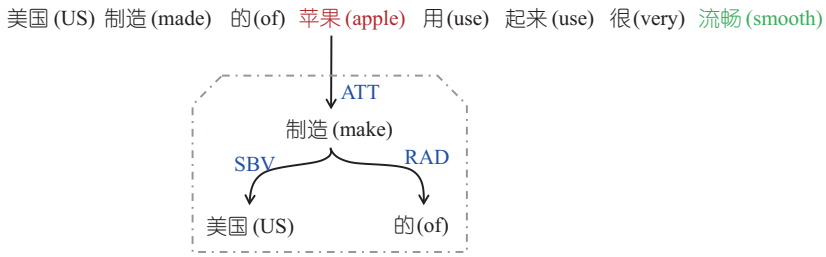

Figure 2 (Color online) Example showing how syntactic structure can help obtain compact semantics.

"RAD") and the related words (such as "制造 (made)", "美国 (US)", "的 (of)") that are connected to "苹果 (apple)" via the syntactic tree, to represent each word on the path. We call this "enhanced word representation". In this way, greater and richer information can be integrated to represent "苹果 (apple)".

As discussed above, the lack of syntactic relation generalization capability and the word semantic representation prevent previous methods from generalizing unseen data well. To avoid these two generalization problems and deeply exploit the semantic information on the syntactic paths, we propose two types of syntactic representation to capture the latent semantics, namely, relation embedding and subtree embedding, to exploit the useful semantic features of the syntactic paths.

- Relation embedding encodes the relation between any two words on the syntactic path. It is used to represent the latent semantics between the target word and the polarity word.
- Subtree embedding encodes each word on the path with the subtree information. It is used to explore the rich information behind each word (especially the target word and the polarity word) on the path.

These two types of syntactic representations can address the above two problems and better generalize the semantic and syntactic information.

To combine the two types of syntactic representation, a neural network is constructed. Recently, deep-learning-based techniques have been widely used to explore semantic representations behind complex structures. This provides us an opportunity to model the syntactic representation into a neural network framework. We use a recursive neural network (RNN) to model the subtree embedding for each word on the path, and the subtree embedding is combined with its word embedding as the enhanced word representation of each word on the syntactic path. That is to say, the representation for each word is generated from all the related relations and words in the parse tree. After that, the enhanced word representation (generated by the subtree embedding and word embedding) and the relation embedding between words can be regarded as the path representation for each sentiment collocation.

To model the syntactic relation embedding and combine the subtree embeddings, a convolutional neural network (CNN) is applied over the two syntactic representations to extract the sentiment collocations from the reviews. The reason we chose the CNN model is that CNN is suitable for capturing the effective features in a flat structure. According to recent improvements in CNN which is used in many natural language processing tasks [10, 11], it has been proven that CNN is efficient for capturing syntactics and semantics between words within a sentence. CNNs typically adopt a max-pooling layer, which uses a max operation over the syntactic representation to capture the information that is most useful.

Our experiments were conducted on five types of reviews, four in product domains (cameras, phones, notebooks and books) and two in service domains (hotels and restaurants). The experimental results for these datasets show that the two types of syntactic representations are able to effectively exploit the latent semantic features of the syntactic paths. More specifically, the syntactic relation embedding can be regarded as the substructure of the syntactic path and well generalizes the relations between the target and the polarity word on the syntactic paths. The subtree embedding can explore more information from the connected words and relations, achieving effective results for the sentiment collocation extraction task. The results also demonstrate that the neural network can better incorporate the two kinds of syntactic representation. It can significantly outperform other syntactic-path-based methods, including those with heavy hand-engineered features. Additionally, the results illustrate that our approach can better capture the latent semantic features on the syntactic paths that neither the common feature-based

methods nor the syntactic-path-based method can handle. Thus, it can further significantly outperform other state-of-the-art methods.

The contributions of the paper are as follows.

• We propose two kinds of syntactic representation, namely, subtree embedding and relation embedding, that can explore the latent general features behind the syntactic paths to describe each sentiment collocation.

• We present a neural-network-based framework to integrate the two kinds of syntactic representation to improve the sentiment collocation extraction.

This paper is organized as follows: Section 2 details the two types of syntactic representation and the neural network framework for sentiment collocation extraction; Section 3 describes the experiments on the corpora from several product and service domains; Section 4 introduces the related work on sentiment analysis; and finally, the conclusion is in Section 5.

2 Approach

2.1 Problem definition

We can formalize the sentiment collocation extraction task as a binary classification problem. In the learning framework, a training or testing instance is generated as a candidate sentiment collocation. In particular, we define all the nouns as candidate target words and all the adjectives as candidate polarity words. Therefore, a sentiment sentence that contains n candidate targets and m candidate polarity words will produce $n \times m$ candidate sentiment collocations. However, this kind of generation can produce many negative instances; thus, some strategies are needed to balance the positive and negative instances in the learning process. Inspired by the idea from Qiu et al.'s work [6], we filter out noisy instances in which neither the target words nor the polarity words appear in the standard sentiment collocations.

Therefore, for each candidate sentiment collocation in a given sentiment sentence, the task can be turned into one of predicting whether the current collocation is a correct one. In other words, we aim to predict whether the polarity word in the candidate collocation is actually modifying the target word.

2.2 Motivation

Previous work has proved that syntactic features are important and widely used in the sentiment collocation task. For instance, in Figure 1, the shortest syntactic paths between target words and polarity words describe the relationship between them and thus are useful for the sentiment collocation extraction task. However, in previous work, these syntactic features were always used in one of two ways. One way is using the syntactic path in a hard matching process, which we call the path-rule-based method. The other way is using the syntactic path as a kind of flat feature and then incorporating it into a classifier, which we call the feature-based method. It can be observed that both methods considered the syntactic path as a whole; therefore, they cannot explore the different perspectives of the syntactic paths.

In order to better explore the latent features of the syntactic path features, we propose to use the lexical and syntactic embedding to obtain the lexical and structural information, respectively, and then combine both of them to generate a more precise structure to present the semantics of a candidate sentiment collocation. We call this combined structure "path representation", as shown in Figure 3. We propose two kinds of syntactic representation to describe the path representation: relation embedding and subtree embedding. Relation embedding reflects the syntactic features between two words on the shortest path, and subtree embedding provides rich syntactic and semantic information for each word on the path. The enhanced word representation (from the subtree embedding and the word embedding) and the relation embedding between words can be regarded as the path representation for each sentiment collocation.

For a candidate sentiment collocation tp, assume the "path representation" can be divided into two parts. One is the word representation, which not only considers the information of the target and the polarity word in the collocation tp but also considers the information of the words on the syntactic path

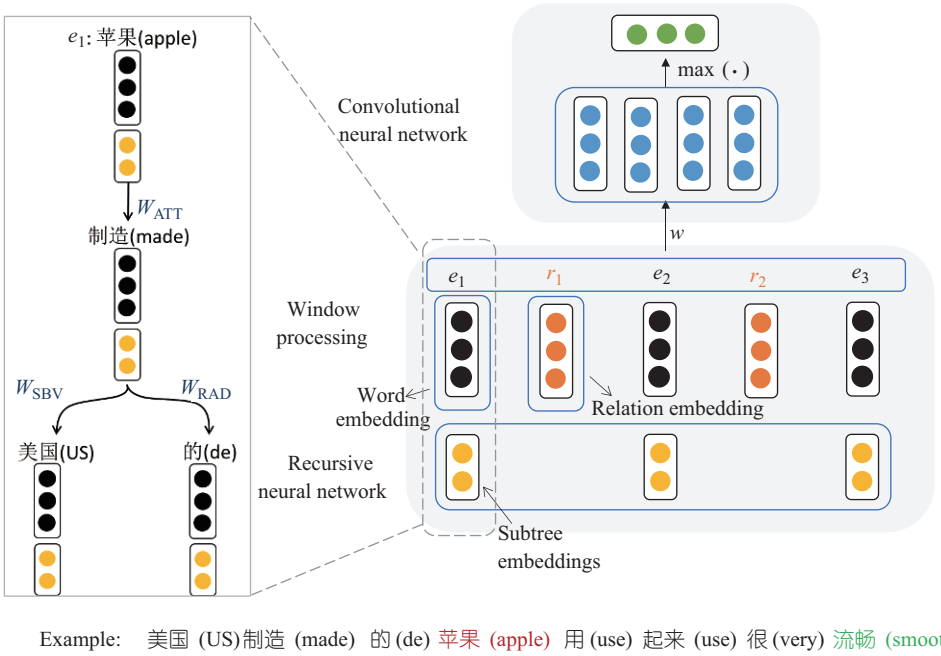

Figure 3 (Color online) A neural network based on two types of syntactic representation for sentiment collocation extraction.

connecting them. The other part is the syntactic representation, which refers to the dependencies on the shortest syntactic path. Here, the word representation is actually an enhanced word representation, generated by the word embedding and the subtree embedding. It is a recursion procedure for generating the subtree embedding. That is, the subtree embedding for each word e_i on the path is generated by the embeddings of the words that e_i is connected to and by the embeddings of the relations that e_i is related to in the parse tree. For example, we assume that for each word e in tp, the word representation is expressed as $x_e \in \mathbb{R}^d$, and for each dependency r in tp, the syntactic representation is expressed as $x_r \in \mathbb{R}^d$. Then, for a sentiment collocation tp, its path representation can be formed as the concatenation of $x_e \in \mathbb{R}^d$ and $x_r \in \mathbb{R}^d$.

Since the generation of subtree embedding for each word on the path is a recursive procedure, it is natural to choose a recursive neural network (RNN) to produce the subtree embedding. To integrate the enhanced word representation (generated by word embedding and subtree embedding) and relation embedding, we use a CNN model to capture these features. It can explore the different perspectives of the syntactic path. Recently, CNN has been widely applied in many natural language processing tasks, such as machine translation [12], information extraction [11, 13, 14], and text classification [15]. A CNN has a special sliding window that can automatically capture different perspectives of a candidate sentiment collocation, and it has a widely used max-over-time pooling operation that can retain the most important features.

Therefore, based on the semantic and syntactic information of the syntactic path connecting the polarity word and the target word in a sentiment collocation, in this paper we propose a novel and effective syntactic-representation-based neural network framework for the sentiment collocation extraction task. In this framework, in addition to considering the lexical information from the sentiment collocation itself, we also consider the words and dependency information on the path between the target and the polarity word.

2.3 Framework

Figure 3 gives a concrete example to illustrate the proposed neural network framework based on two types of syntactic representation for the sentiment collocation extraction. It can be observed that for each sentiment collocation, we use the shortest path between the target word and the polarity word to

represent it. Furthermore, the syntactic path can convey two types of important information, namely the word and the syntactic relation. Thus, we can explore much useful knowledge to represent the word and the relation on the syntactic path to improve the sentiment collocation extraction. In this framework, the word representation on the path contains two parts. One is the word embedding vector, which can be obtained from the pretrained word embedding tables. The other part is the subtree embedding, which is a kind of syntactic information and is supplementary to the word embedding. From Figure 3, we can see that the subtree embedding is constructed by a recursive procedure. For example, to capture the subtree embedding of the target word "苹果 (apple)", we need to use the information from its connected words (such as "制造 (made)") and its related relations (such as "ATT"). Recursively, the word representation for the word "制造 (made)" also contains two parts, and its subtree embedding part is also related to its connected words and relations. In addition, we use the relation embedding to represent the relationships between two words on the path.

Suppose that the candidate sentiment collocation tp has three words (e_1 and e_3, referring to the target and the polarity word, respectively, and e_2, the assisted word on the shortest syntactic path) and two dependency relations (r_1 and r_2). For example, in Figure 3, the candidate sentiment collocation is 〈 苹果, 流畅 〉 (⟨apple, smooth⟩), in which e_1 is "苹果" and e_3 is "流畅". After parsing, the word "是', which is on the shortest syntactic path, is e_2. Since the shortest syntactic path contains two relations, "SBV" and "CMP", these refer to r_1 and r_2, respectively. The idea of the method in our paper is to deeply explore the information from every e_i and r_j. We associate each word e and dependency relation r with a vector representation x_e, $x_r \in \mathbb{R}^d$. Here, the vector x_e contains two parts. We use x_w to represent the word embedding, and x_s to represent its subtree embedding. We concatenate x_w and x_s into x_w for each word as the enhanced word representation. To integrate all the semantic and syntactic information behind the syntactic path, a neural network is generated to model the syntactic features based on the two types of syntactic representation for a candidate sentiment collocation. More specifically, a recursive neural network model is used to generate the subtree embedding for each word on the syntactic path. Inspired by the work of Liu et al. [11], we set a weight w_r for each relation in the subtree and then the weights and the subtree embeddings are learned during training using the following equation:

$$\mathrm{st}_w = f\left(\sum_{q\in \mathrm{children}(w)} w_{r_{(w,q)}} \cdot x_q + b\right).$$

As mentioned above, before entering the CNN model, each token x_i is expressed into a real-valued vector by the use of representation tables as follows to obtain the different characteristics of the token.

(1) The word representation table is used to obtain the latent lexical information of the words in the collocation tp. As described in the above discussion, each enhanced word representation includes two parts, the word embedding and the subtree embedding parts. Please note that the word embedding can be initialized by some pre-trained word embeddings.

(2) The syntactic representation table is used to capture the latent syntactic properties of the dependencies on the shortest syntactic path. In practice, we initialize this table randomly.

We search the word representation table to obtain the vectors for each word token e_i, and we obtain the vectors from the syntactic representation table as the representation for the syntactic token r_i. In the method described in this paper, we combine the lexical and the syntactic representations to better explore both the lexical and the syntactic information behind the syntactic path. An easy way is to concatenate the two representations together and then regard them as the input to the CNN framework. Then, for each sentiment collocation candidate tp, we use a simple but effective way to concatenate them into a single vector $\boldsymbol{x}$, which is called the syntactic path representation, to represent tp. As a result, tp is shown as a matrix $\boldsymbol{x} = [\boldsymbol{x}_{e_1}, \boldsymbol{x}_{r_1}, \boldsymbol{x}_{e_2}, \boldsymbol{x}_{r_2}, \ldots, \boldsymbol{x}_{r_i}, \boldsymbol{x}_{e_j}]$ of size $d \times (i + j)$. Formally,

$$\boldsymbol{x} = \boldsymbol{x}_{e_1} \oplus \boldsymbol{x}_{r_1} \oplus \cdots \oplus \boldsymbol{x}_{e_j},$$

where $\oplus$ is the concatenation operator.

Table 1 Statistics for the Chinese datasets of six domains

Domain	# Reviews	# Sentences	# Collocations
Camera	138	1249	1335
Notebook	56	623	674
Phone	123	1350	1479
Book	349	1270	424
Hotel	440	1808	1120
Restaurant	386	1398	689
All	1492	7698	5721

Since we apply a sliding window in the CNN model, we set the context to a fixed window size k. Inspired by Liu et al.'s work [11], when $k = 3$, the sliding windows of a candidate tp can be shown as $[r_{\text{start}}e_1r_1], [r_1e_2r_2], \ldots, [r_{n-1}e_nr_{\text{end}}]$, where r_{start} and r_{end} denote the beginning and the end, respectively, of the shortest syntactic path linking the target word and the polarity word. Each part in the window processing is defined as $\hat{\boldsymbol{x}}$. Then, for a given syntactic-path-representation based concatenated sequence $\hat{\boldsymbol{x}}$, the convolution operation uses a filter $\boldsymbol{w}$ with a bias term b, which can be described by

$$c = f(\boldsymbol{w} \cdot \hat{\boldsymbol{x}} + b),$$

where f is a non-linear activation function, for which we can use the rectified linear unit (ReLu) or sigmoid function.

Based on the above, the syntactic path representation $\boldsymbol{x}$ is then passed through a convolution layer, a max-pooling-layer, and finally a sigmoid function, to perform the candidate sentiment collocation classification.

3 Experiments

3.1 Experimental Setup

The experiments were conducted on two types of datasets. One dataset contains three product domains, namely, camera, notebook, and phone; these data are from Task3 of the Chinese Opinion Analysis Evaluation (COAE) [16][1)]. To expand the scale of the dataset, we collected many reviews from other domains. In particular, we collected book reviews from https://www.jd.com/, and we collected many reviews from two service domains, namely, the restaurant and hotel reviews from http://www.meituan.com/.

Table 1 describes the datasets, in which 7698 sentiment sentences containing 5721 sentiment collocations were manually found and annotated from 1492 reviews.

A common evaluation method, P, R, and F-score, was used to measure the performance of each system on the sentiment collocation extraction task. Specially, a fuzzy matching evaluation was adopted for the sentiment collocation extraction. That is to say, given an extracted sentiment collocation $\langle t, p\rangle$, whose standard result is $\langle t_s, p_s\rangle$, if t is a substring of t_s, and at the same time p is a substring of p_s, the extracted $\langle t, p\rangle$ is considered to be a correct sentiment collocation.

3.2 Comparison systems

To show the impact of the two types of syntactic representation and the neural network framework, we designed several comparison systems as follows.

• **CNN_PathWord.** For each syntactic path linking the target word and the polarity word, we use the word embedding on the path to represent the whole path, and then a CNN framework is used to integrate the word embeddings to classify the candidate sentiment collocation.

1) http://www.ir-china.org.cn/coae2008.html.

• **CNN_PW_Relation.** In addition to the word embeddings on the path, we also use the relation embedding proposed in this paper to represent the syntactic path. In the same way, we use the new path representation as the input of the CNN framework.

• **CNN_PW_R_Flat.** In addition to the word embedding and the syntactic relation embedding, we also incorporate several useful features in the CNN framework.

• **CNN_PW_R_Subtree.** In addition to the word embedding and the syntactic relation embedding, we also use subtree embedding to represent each word on the path. That is to say, in this method, each word is represented by the commonly used word embedding and the subtree embedding. Then, the new path representation generated by word embedding, subtree embedding, and relation embedding is used as the input to the CNN framework.

Here, the initial values of the word embeddings of all the above systems were 50-d trained on a large corpus of Sina Weibo. The subtree embeddings and the relation embeddings were also of 50 dimensions and were initialized randomly.

Further, to illustrate that the neural-network-based method proposed in this paper is effective, we consider two syntactic-path-based methods as the baselines. One is the Path-Rule-Based method, which mainly uses nine syntactic path rules to obtain the sentiment collocations. The other one is the Feature-Based method, which applies machine learning tools and considers the syntactic path as a feature.

Path-Rule-Based method. For comparison, we used a state-of-the-art path-rule-based method to extract the sentiment collocations; this method is a semi-supervised method from Qiu et al.'s work [6]. The idea is based on two observations. The first one is that a natural syntactic relation always appears between the polarity words and the targets because of the fact that polarity words are used to modify target words. The second one is that polarity words and targets themselves also have relations in some cases [6].

More specifically, firstly an initial seed polarity word lexicon and several syntactic relations between the polarity words and the targets are used to extract the targets. Next, a new target lexicon is constructed. Then, the newly constructed target lexicon and the same syntactic relations are used to extract the polarity words, and finally, a larger-scale polarity word lexicon is obtained. This can be regarded as an iterative procedure, since this method can iteratively produce new polarity words and targets back and forth using the syntactic relations. For this baseline, nine syntactic path rules, which were proposed by Zhao et al. [8], were used. To obtain the parsing trees of the sentiment sentences, we used a Chinese natural language processing toolkit, Language Technology Platform (LTP) [17].

Feature-Based method. This method is the other baseline for comparison. In the feature-based method, all the nouns are considered as candidate targets, and all the adjectives are candidate polarity words. Therefore, $n * m$ candidate sentiment collocations are obtained from a sentiment sentence that contains n candidate targets and m candidate polarity words. Then, a binary classifier is generated on the training instances using an SVM model.

Several basic features were adopted for the target and polarity word in a sentiment collocation. Figure 4 lists the features and their detailed descriptions. These include the basic word feature (w), the POS tag feature (t), and their combination context features (01–04). Moreover, the flat syntactic features (05) are also considered. In this part, the syntactic path $\mathsf{SynF}(w_{\text{target}}, w_{\text{polarity}})$ between the target and the polarity word is regarded as the flat syntactic feature.

For learning, the binary SVMLight [18][2)] was used. Since the training and testing instances are unbalanced (there are 3–4 times as many negative instances sd positive instances), the cost factor is tuned when training with SVM.

3.3 Results of sentiment collocation extraction

Table 2 shows the experimental results of our proposed method and of the comparison systems mentioned in Subsection 3.2. To summarize, we find the following:

2) http://www.cs.cornell.edu/People/tj/svm_light/.

Basic features

01: $w_{i+k},\ -1 \leqslant k \leqslant 1$

02: $w_{i+k-1} \circ w_{i+k},\ 0 \leqslant k \leqslant 1$

03: $t_{i+k},\ -1 \leqslant k \leqslant 1$

04: $t_{i+k-1} \circ t_{i+k},\ 0 \leqslant k \leqslant 1$

Syntactic features

05: $\mathsf{SynF}(w_{\mathbf{target}}, w_{\mathbf{polarity}})$

Figure 4 Basic features for sentiment collocation extraction.

Table 2 Results of sentiment collocation extraction using different methods [a)]

Domain	Method	P (%)	R (%)	F-score (%)	Domain	Method	P (%)	R (%)	F-score (%)
	CNN_PathWord	72.1	48.0	57.8		CNN_PathWord	63.0	47.0	54.1
	CNN_PW_Relation	74.3	77.0	75.7		CNN_PW_Relation	68.2	76.0	71.9
Phone	CNN_PW_R_Flat	74.7	80.0	77.3	Camera	CNN_PW_R_Flat	69.1	77.0	73.0
	CNN_PW_R_Subtree	76.3	81.0	78.5		CNN_PW_R_Subtree	71.8	75.0	73.6
	Feature-Based	60.6	83.9	70.3		Feature-Based	52.0	75.2	61.5
	Path-Rule-Based (*)	77.3	60.9	68.1		Path-Rule-Based (*)	74.7	58.4	65.6
	CNN_PathWord	62.1	48.0	54.4		CNN_PathWord	49.3	100.0	66.0
	CNN_PW_Relation	60.0	72.0	65.4		CNN_PW_Relation	75.6	70.0	72.5
Notebook	CNN_PW_R_Flat	67.3	72.0	69.5	Book	CNN_PW_R_Flat	75.4	81.0	77.9
	CNN_PW_R_Subtree	67.8	76.0	71.6		CNN_PW_R_Subtree	75.2	79.0	76.9
	Feature-Based	59.0	81.2	68.4		Feature-Based	66.5	89.1	76.2
	Path-Rule-Based (*)	74.1	56.8	64.3					
	CNN_PathWord	78.0	70.0	73.8		CNN_PathWord	47.9	98.0	64.3
	CNN_PW_Relation	73.9	78.0	76.0		CNN_PW_Relation	78.1	71.0	74.2
Hotel	CNN_PW_R_Flat	72.1	81.0	76.2	Restaurant	CNN_PW_R_Flat	73.5	77.0	75.4
	CNN_PW_R_Subtree	77.9	82.0	79.7		CNN_PW_R_Subtree	75.9	79.0	77.3
	Feature-Based	61.5	72.1	66.4		Feature-Based	66.9	78.6	72.3
	CNN_PathWord	65.8	59.4	62.4					
	CNN_PW_Relation	71.3	75.3	73.2					
All	CNN_PW_R_Flat	71.8	78.2	74.9					
	CNN_PW_R_Subtree	74.2	78.6	**76.4**					
	Feature-Based	59.3	79.3	67.9					

a) The results indicated by * are obtained from the work of Zhao et al. [8].

First, in comparing the CNN_PathWord and CNN_PW_Relation systems, we observe that CNN_PW_Relation, which adds the syntactic relation embeddings to represent the syntactic path, performs better than CNN_PathWord, which only uses the word embeddings. This illustrates that relation embedding, which explores the syntactic features of the path, is effective.

Secondly, in comparing the CNN_PW_Relation and CNN_PW_R_Flat systems, we observe that the CNN_PW_R_Flat system, which uses the flat features shown in Figure 4, performs better than CNN_PW_Relation, which automatically selects the features according to the CNN model. This shows that the manual features can be regarded as a complement to the CNN network.

Thirdly, in comparing the CNN_PW_Relation and CNN_PW_R_Subtree systems, we observe that the CNN_PW_R_Subtree system, which adopts the subtree embedding as a supplementary part to represent each word on the path, performs better than CNN_PW_Relation, which only uses the word embedding as the word representation. This illustrates that the subtree embedding, which can convey the rich syntactic information for each word on the path, is effective. In addition, in comparing the CNN_PW_R_Flat and CNN_PW_R_Subtree systems, we observe that CNN_PW_R_Subtree also performs better than CNN_PW_R_Flat. This also demonstrate the effectiveness of subtree embedding.

Finally and most importantly, for all the domains, including both products and services, the CNN_PW_R_Subtree system performs best, achieving an F-score of 76.4%. It is significantly ($p < 0.01$) better than the CNN_PathWord system (which has an F-score of 62.4%), and at the same time significantly ($p < 0.01$) better than the system CNN_PW_Relation (with the F-score of 73.2%). This indicates that the two types of syntactic representation, namely the relation embedding and the subtree embedding, are effective and can improve the performance considerably. Moreover, it also shows that the syntactic features are useful for sentiment collocation extraction, because no syntactic features are used in the CNN_PathWord system.

In comparing the CNN_PW_R_Subtree system to the two baselines, namely, the Path-Rule-Based and Feature-Based methods, it can be seen that even though all of them use the syntactic features, the CNN_PW_R_Subtree system performs best. This indicates that using the two types of syntactic representation to represent a syntactic path is a better way to describe the relationship between the target word and the polarity word and that this method can explore the latent semantic features of the syntactic relationship. Furthermore, this demonstrates that the neural-network-based framework, which incorporates the syntactic representations, is effective for sentiment collocation extraction.

4 Related work

Sentiment analysis has been very popular in recent years [19, 20]. Sentiment collocation extraction is a basic task in sentiment analysis. To perform this task, most previous methods have used the relationships between targets and polarity words to obtain the sentiment collocation. In early studies, researchers recognized the target first and then chose its polarity word within a window of size k [21]. However, this type of method is too heuristic, and thus the performance proved weak. To solve this problem, many researchers found syntactic rules that can better describe the relationships between targets and polarity words. For instance, Bloom et al. [5] constructed a linkage specification lexicon containing 31 patterns, and Qiu et al. [6] proposed a double propagation method that uses eight heuristic syntactic patterns for the sentiment collocation extraction. Recently, Xu et al. [7] applied syntactic rules to extract the candidate collocations in a two-stage framework. Zhao et al. [8] proposed a framework adopting a CRF based sentiment sentence compression model, as a preprocessing step to improve the sentiment collocation extraction.

Based on these ideas, it can be concluded that syntactic features are vital for the sentiment collocation extraction task. The previous work focused mainly on the exact matching of syntactic features such as syntactic paths. However, this can result in the low recall values seen in the rule-based method. Moreover, this kind of feature is not general and is prone to leading to the data sparsity problems of the feature-based method. Our proposed syntactic representation can exploit the latent syntactic features from the path, and meanwhile, the CNN model can better incorporate these features.

Additionally, sentiment collocation extraction aims to explore the relationship between two words in a sentiment sentence, and thus it can be treated as a kind of relation extraction task. We find that many researchers have applied CNN-based methods for relation extraction. For instance, Vu et al. [14] presented a new context representation for CNN for relation classification and then combined it with a bi-directional recurrent neural network. Santos et al. [22] addressed the relation classification task using a convolutional neural network that performs classification by ranking (CR-CNN). Liu et al. [11] used a CNN to incorporate the augmented dependency path (ADP) structure for the relation classification task. CNN is also very popular in the sentiment classification task; Kim [23] reported on a series of experiments with CNNs trained on top of pre-trained word vectors for sentence-level sentiment classification. Ma et al. [24] proposed a dependency-based convolution approach, making use of tree-based n-grams rather than surface ones to classify the sentiment of a sentence. Inspired by their work, we have proposed the two syntactic representations that are incorporated into a neural network framework.

5 Conclusion

In this paper, we propose a method for extracting sentiment collocations by incoraporating the syntactic features into a neural network framework. Two types of syntactic representation are designed, namely,

relation embedding and subtree embedding, to capture the latent semantics behind the syntactic path. Relation embedding is for encoding the relation between any two words on the syntactic path. It can effectively represent the latent semantics between target and polarity word. Subtree embedding is for encoding each word on the path with its own subtree information. It can explore the rich syntactic and lexical information behind each word on the path. The experimental results on datasets from four product domains and two service domains show that the two kinds of syntactic representation that consider more comprehensive dependency information can explore the latent syntactic features that other state-of-the-art methods cannot capture. Moreover, the proposed neural network framework performs significantly better than the flat syntactic-path-based method, indicating that our proposed syntactic representations are effective.

In this paper, the syntactic features are mainly used and other useful features are omitted, such as the lexical features. Thus, in the future, we will incorporate other useful features into the neural network framework. Further, we observed that the syntactic path linking the target word and the polarity word is directed, and a Long-Short-Term Memory (LSTM) network can better handle this feature because of its special design. Thus, we will use LSTM to incorporate our proposed syntactic representations. In addition, in a recursive neural network it is more natural to use the constituent parsing results, so, in future, we plan to replace the dependency parsing with constituent parsing in the neural network for sentiment collocation extraction.

Acknowledgements This work was supported by National Basic Research Program of China (973 Program) (Grant No. 2014CB340506) and National Natural Science Foundation of China (Grant Nos. 61632011, 61370164). We thank the anonymous reviewers for their helpful comments.

Conflict of interest The authors declare that they have no conflict of interest.

References

1 Pang B, Lee L L. Opinion mining and sentiment analysis. Found Trends Inf Retr, 2008, 2: 1–135
2 Liu B. Sentiment analysis and opinion mining. Synth Lect Human Language Tech, 2012, 5: 1–167
3 Abbasi A, Chen H, Salem A. Sentiment analysis in multiple languages: feature selection for opinion classification in web forums. ACM Trans Inf Syst, 2008, 26: 1–34
4 Duric A, Song F. Feature selection for sentiment analysis based on content and syntax models. Decis Support Syst, 2012, 53: 704–711
5 Bloom K, Garg N, Argamon S. Extracting appraisal expressions. In: Proceedings of Human Language Technologies: the Annual Conference of the North American Chapter of the Association for Computational Linguistics, Rochester, 2007. 308–315
6 Qiu G, Liu B, Bu J J, et al. Opinion word expansion and target extraction through double propagation. Comput Linguist, 2011, 37: 9–27
7 Xu L H, Liu K, Lai S, et al. Mining opinion words and opinion targets in a two-stage framework. In: Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics, Sofia, 2013. 1764–1773
8 Zhao Y Y, Che W X, Guo H L, et al. Sentence compression for target-polarity word collocation extraction. In: Proceedings of the 25th International Conference on Computational Linguistics: Technical Papers, Dublin, 2014. 1360–1369
9 Zhao Y Y, Li S Q, Qin B, et al. Encoding dependency representation with convolutional neural network for target-polarityword collocation extraction. In: Social Meida Processing. Singapore: Springer, 2016
10 Chen Y B, Xu L, Liu K, et al. Event extraction via dynamic multi-pooling convolutional neural networks. In: Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing, Beijing, 2015. 167–176
11 Liu Y, Wei F R, Li S J, et al. A dependency-based neural network for relation classification. In: Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing, Beijing, 2015. 285–290
12 Meng F D, Lu Z D, Wang M X, et al. Encoding source language with convolutional neural network for machine translation. In: Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing, Beijing, 2015. 20–30
13 Nguyen T H, Grishman R. Event detection and domain adaptation with convolutional neural networks. In: Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing, Beijing, 2015. 365–371
14 Vu N T, Adel H, Gupta P, et al. Combining recurrent and convolutional neural networks for relation classification. In:

Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, San Diego, 2016. 534–539
15 Lee J Y, Dernoncourt F. Sequential short-text classification with recurrent and convolutional neural networks. In: Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, San Diego, 2016. 515–520
16 Zhao J, Xu H B, Huang X J, et al. Overview of chinese pinion analysis evaluation 2008. Proceedings of the 1st Chinese Opinion Analysis Evaluation (COAE), 2008
17 Che W X, Li Z H, Liu T. LTP: a chinese language technology platform. In: Proceedings of the 23rd International Conference on Computational Linguistics: Demonstrations, Beijing, 2010. 13–16
18 Joachims T. Learning to Classify Text Using Support Vector Machines – Methods, Theory, and Algorithms. Norwell: Kluwer Academic Publishers, 2002
19 Gui L, Zhou Y, Xu R F, et al. Learning representations from heterogeneous network for sentiment classification of product reviews. Knowledge-Based Syst, 2017, 124: 34–45
20 Chen T, Xu R F, He Y L, et al. Learning user and product distributed representations using a sequence model for sentiment analysis. IEEE Comput Intell Mag, 2016, 11: 34–44
21 Hu M Q, Liu B. Mining and summarizing customer reviews. In: Proceedings of the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Seattle, 2004. 168–177
22 dos Santos C, Xiang B, Zhou B. Classifying relations by ranking with convolutional neural networks. In: Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing, Beijing, 2015. 626–634
23 Kim Y. Convolutional neural networks for sentence classification. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), Doha, 2014. 1746–1751
24 Ma M B, Huang L, Xiang B, et al. Dependency-based convolutional neural networks for sentence embedding. In: Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing, Beijing, 2015. 174–179