Information Processing and Management 54 (2018) 576–592

Contents lists available at ScienceDirect

Information Processing and Management

journal homepage: www.elsevier.com/locate/infoproman

DRI-RCNN: An approach to deceptive review identification using recurrent convolutional neural network

Wen Zhang[a,b,*], Yuhang Du[b], Taketoshi Yoshida[c], Qing Wang[d]

[a] School of Economics and Management, Beijing University of Technology, Beijing 100124, PR China
[b] Research Center on Big Data Sciences, Beijing University of Chemical Technology, Beijing 100029, PR China
[c] School of Knowledge Science, Japan Advanced Institute of Science and Technology, 1-1 Ashahidai, Nomi City, Ishikawa 923-1292, Japan
[d] State Key Laboratory of Computer Science, Institute of Software, Chinese Academy of Sciences, Beijing 100190, PR China

ARTICLE INFO

Keywords:
Deceptive review identification
Recurrent convolutional vector
Contextual knowledge
Word embedding
DRI-RCNN

ABSTRACT

With the widespread of deceptive opinions in the Internet, how to identify online deceptive reviews automatically has become an attractive topic in research field. Traditional methods concentrate on extracting different features from online reviews and training machine learning classifiers to produce models to decide whether an incoming review is deceptive or not. This paper proposes an approach called DRI-RCNN (Deceptive Review Identification by Recurrent Convolutional Neural Network) to identify deceptive reviews by using word contexts and deep learning. The basic idea is that since deceptive reviews and truthful reviews are written by writers without and with real experience respectively, the writers of the reviews should have different contextual knowledge on their target objectives under description. In order to differentiate the deceptive and truthful contextual knowledge embodied in the online reviews, we represent each word in a review with six components as a recurrent convolutional vector. The first and second components are two numerical word vectors derived from training deceptive and truthful reviews, respectively. The third and fourth components are left neighboring deceptive and truthful context vectors derived by training a recurrent convolutional neural network on context vectors and word vectors of left words. The fifth and six components are right neighboring deceptive and truthful context vectors of right words. Further, we employ max-pooling and ReLU (Rectified Linear Unit) filter to transfer recurrent convolutional vectors of words in a review to a review vector by extracting positive maximum feature elements in recurrent convolutional vectors of words in the review. Experiment results on the spam dataset and the deception dataset demonstrate that the proposed DRI-RCNN approach outperforms the state-of-the-art techniques in deceptive review identification.

1. Introduction

With the development of E-Commerce (Buettner, 2016), there is an increasing number of users' reviews on goods and services online (Mudambi & Schuff, 2010; Chatterjeei, 2001; Valant, 2015). For instance, it is estimated that in Amazon.com, 5% of the total

* Corresponding author at: School of Economics and Management, Beijing University of Technology, Beijing 100124, PR China.
E-mail addresses: zhangwen@mail.buct.edu.cn (W. Zhang), yuhang_du@mail.buct.edu.cn (Y. Du), yoshida@jaist.ac.jp (T. Yoshida), wq@itechs.iscas.ac.cn (Q. Wang).

https://doi.org/10.1016/j.ipm.2018.03.007
Received 19 June 2017; Received in revised form 29 January 2018; Accepted 14 March 2018
Available online 31 March 2018
0306-4573/

244 million users write reviews on their purchased products[1,2]. According to public survey[3,4], 90% of the online buyers agree that their buying decisions are influenced by online reviews and, 88% of them agree that they trust online reviews as much as personal recommendations. Unfortunately, due to lack of strict inspection on online reviews, a large number of fake, misleading even deceptive reviews are coming up with the Internet for money-seeking even vicious purpose. With the online word-of-mouth effect (Hung, 2017), positive online reviews are helpful for merchants to earn a good reputation to promote their products while negative online reviews will degrade its image among consumers.

On the one hand, some business companies are striving to create positive online reviews for themselves. Meanwhile, a small part of them are also attempting to create negative reviews for their competitors in the market to defame their brands. Normally, online reviews are created by product buyers to write their experience in using the product. However, in some unexpected cases, the online merchants also created some online reviews by themselves, even by hiring "water army" to post online reviews (Liu, 2015). On the other hand, for an online shopping customer, it is indispensable to use online reviews for references in purchase decision making because he or she cannot see the tangible products in the online virtual market. In addition, it is reported by researchers that human beings are impossible to identify deceptive reviews and truthful reviews to a satisfactory extent (Ott, Choi, Cardie, & Hancock, 2011). Under the condition that an anonymous user can post online reviews freely with little cost but with great return of potential commercial profit, spam, fake, misleading and even fraudulent online reviews are fast-growing and pervasive in the Internet. As a result, there are an increasing number of customers who are anxiously worrying about being cheated even trapped by these fake reviews in online shopping (Jindal & Liu, 2008; Lim, Osman, Salahuddin, Romle, & Abdullah, 2016). This makes deceptive review identification has become a hot topic in the research field (Cagnina & Rosso, 2017; Gokhman, Hancock, Prabhu, Ott, & Cardie, 2012; Hernández, Montes-y-Gómez M, Rosso, & Guzmán R., 2015; Mudambi & Schuff, 2010; Ott et al., 2011; Rosso & Cagnina, 2017).

Although the existing research on automatic review identification can attain accuracies up to 90%, there are still leaving space worth more work. On the one hand, most existing work focus on extracting traditional linguistic units from texts such as individual words, n-grams, POS tags and CFG (context-free grammar) rules to index the reviews to be classified. It is well recognized that there should be a latent semantic space to express words' non-linear relations (Van Rijsbergen, 1997), i.e. co-occurrence. For this purpose, Latent Semantic Indexing (LSI) (Olmos, Jorge-Botana, Luzón, Martín-Cordero, & León., 2016), Probabilistic Latent Semantic Analysis (Hofmann, 1999) and Latent Dirichlet Allocation (Ma, Zhang, Liu, Li, & Yuan, 2016) are proposed to distill those so-called topics implicitly inherent in texts by either linear algebra methods or statistical generative model. However, it is very difficult to articulate those topics clearly. For example, to the best of our knowledge, we still do not know how to predefine the feasible number of topics for a given text collection and how to explain the decomposed matrices and the extracted distributions obtained from the mentioned methods, although some heuristic methods are presented recently (Ding, Li, & Peng, 2007; Srijith, Hepple, Bontcheva, & Preotiuc-Pietro., 2017; Toral, Pecina, Wang, & van Genabith, 2015).

On the other hand, an obvious disadvantage is that the existing work overlooks the surrounding information of words in a sentence. Although it can also makes success by regarding each word as an individual item in some text classification tasks (Sebastiani, 2002) such as genre classification, news topic classification and spam Email classification, automatic deceptive review identification is more complicated than these tasks. First, it is not hard by human beings to accomplish those tasks successfully by reading the whole texts to be classified if being given enough time and labor (Ott et al., 2011). Second, individual words' occurrences are more important clues than words' surrounding information in those automatic classification tasks. Nevertheless, in automatic deceptive review identification, words' surrounding information plays a more important role than occurrences. The reason is that for this problem, words' occurrences between deceptive reviews and truthful reviews are almost the same and context information of words is the most important clue in deciding a review deceptive or truthful (Feng & Hirst, 2013; Zhang, Bu, Yoshida, & Zhang, 2016).

The primary objective of this research is to study the effectiveness of using deep learning techniques to identify deceptive reviews and truthful reviews. Prior work (Collins, 2013; Feng & Hirst, 2013; Ott et al., 2011; Zhang et al., 2016) shows that statistical machine learning is a viable approach to improving the accuracy of automatic deceptive review identification. However, their focus is on extracting linguistic units from reviews by natural language processing to provide inputs for traditional machine learning methods such as support vector machines (Feng & Hirst, 2013) and naïve Bayes (Zhang et al., 2016). This paper proceeds with automatic deceptive review identification in a different manner. That is, we deal with online reviews in stand point of deep neural network learning. First, different from existing methods, we use recurrent convolutional vector to denote each word in a feature space constructed by contextual knowledge. Second, each review is represented by aggregating the recurrent convolutional vectors of words by max-pooling and ReLU (Rectified Linear Unit) filter (Nair & Hinton, 2010) and this is completely different from traditional text representation methods such as BOW (Van Rijsbergen, 1997), LSI (Olmos et al., 2016), PLSA (Hofmann, 1999) and LDA (Ma et al., 2016). Third, unlike existing work in using sophisticated learning methods to improve deceptive review identification, the method for automatic identification adopted in the paper is relatively simple, i.e., back propagation (Bottou, 1998) with softmax function.

This paper proposes an approach called DRI-RCNN (Deceptive Review Identification by Recurrent Neural Network) to automatically identify deceptive reviews by using word contexts and deep learning. The basic idea is that since deceptive reviews and

[1] Amazing Amazon Statistics and Facts (July 2016), online: http://expandedramblings.com/index.php/ amazon-statistics/.

[2] What percentages of buyers write reviews on Amazon? Online: https://www.quora.com/What-percentage-of-buyers-write-reviews-on-Amazon.

[3] A. Gesenhues.: Survey: 90% Of Customers Say Buying Decisions Are Influenced By Online Reviews. April 9, 2013. Online: http://marketingland.com/survey-customers-more-frustrated-by-how-long-it-takes-to-resolve-a-customer-service-issue-than-the-resolution-38756.

[4] M. Anderson.: 88% Of Consumers Trust Online Reviews As Much As Personal Recommendations. July 7, 2014. Online: http://searchengineland.com/88-consumers-trust-online-reviews-much-personal-recommendations-195803.

truthful reviews are written by writers without and with real experience, there should be different contexts of words used by them. If deceptive and truthful contexts are embedded into word representation to concatenate review representation, then it should be not difficult to differentiate deceptive reviews and truthful reviews.

The major contribution of the paper can be summarized as follows. First, we propose recurrent convolutional vector to represent each word in the review by using its deceptive and truthful contexts and word embedding. Second, we present a deep neural network structure to identify deceptive reviews by combining max-pooling and ReLU filter. The algorithm to train the deep neural network is also presented to update the parameters involved in the network. Third, we conduct extensive experiments to demonstrate the performances of the DRI-RCNN approach on the spam dataset (Ott et al., 2011) and the deception dataset (Li, Ott, Cardie, & Hovy, 2014) to compare the propose approach with state-of-the-art techniques. The remaining of the paper is organized as follows. Section 2 presents the related work with DRI-RCNN. Section 3 proposes the DRI-RCNN approach. Section 4 conducts the experiments. Section 5 concludes the paper.

2. Related work

The related work of the paper includes two aspects. The one is deceptive review identification and the other one is text processing by deep neural network.

2.1. Deceptive review identification

Spam detection was historically studied in the Web page and E-mail domains where noisy messages, advertisements and vicious information should be filtered out from the Web browser or Email box (Ren & Ji, 2017). With the booming of E-commerce, deceptive review identification is present to detect deceptive reviews from truthful reviews. Mostly, the spam Web page and Emails are used to cheat its readers by fake information for network defraud. However, the deceptive reviews are usually written by “water army” (Chen, Wu, Srinivasan, & Zhang, 2013) to mislead customers to purchase unwanted products. Moreover, deceptive reviews are more elusive to discern than spam Web pages and Emails.

Jindal and Liu (2008) analyzed 5.8 million reviews and 2.14 million reviewers from Amazon.com and reported that spam reviews are widespread. They categorized spam reviews into three types: untruthful opinions (type 1), reviews on brands only (type 2) and non-reviews (type 3) including advertisements and irrelevant reviews. However, due to lack of golden set of spam reviews, they merely considered duplicate reviews as spam reviews. They showed that spam reviews of type 2 and 3 are easily found using manually labeled examples. However, the spam reviews of type 1 are the most difficult to be recognized because even human beings are unable to recognize type 1 spam reviews reliably. For this reason, they argued that other ways have to be explored in order to find training examples of type 1 spam reviews.

To build a systematic approach for corpus creation within deception field, Gokhman et al. (2012) investigated traditional non-gold standard and crowdsourced approaches. The traditional approach includes sanctioned deception method and unsanctioned deception method. The non-gold standard approach includes manual annotation of deception method and heuristically labeled method. They argued that the crowdsourcing approach is the most promising one to create deceptive online reviews. This finding also validates collection of the spam dataset by Ott et al. (2011) and the deception dataset by Li et al. (2014) from TripAdvisor[5] and Amazon Mechanical Turk.[6]

To automatically identify deceptive reviews, Ott et al. (2011) used LIWC2007 (Linguistic Inquiry and Word Count) software (Pennebaker, Chung, Ireland, Gonzales, & Booth, 2007) to produce psychological features from reviews and combined these features with n-grams to automatically categorize spam opinions using support vector machines as the machine learning classifier. Their experiments on the spam dataset reported that combination of psychological features and n-grams can produce accuracy as 90% in spam opinion categorization and is much better than human judges of accuracy as 60%. They observed that truthful opinions tend to include more sensorial and concrete language than deceptive opinions and more specific about spatial configurations. Feng, Banerjee, and Choi (2012) used not only part-of-speech (POS) features but also Context Free Grammar (CFG) features to improve the spam review detection. Their experiments on four datasets as the spam dataset, Yelp dataset, Heuristic TripAdvisor dataset and Essays dataset showed that the deep syntax feature can significantly improve the performance of spam opinion detection in comparison with the baseline method by combining words and POS (part-of-speech) (Marcus, Marcinkiewicz, & Santorini, 1993) features.

Li et al. (2014) proposed to adopt a Bayesian generative approach called SAGE (Sparse Additive Generative Model) to identify deceptive reviews by probabilistic generative model. Their basic idea is that each word in a review is generated by three random factors as the sentiment of the review, the domain of the review as well as the source of the review and the high order interaction of the three random factors. The source of the review means that the writer of the review is a real customer or a deceptive turker. Thus, the problem of deceptive review identification is transferred to maximize a posterior probability to generate a review given words, its features, sentiment, domain and source of the review. Their experiments on the deception dataset demonstrate that among the three types of features as unigram, LIWC and POS, the unigram feature can achieve the best performance as 0.817 in deceptive review identification.

Ren and Ji (2017) also make use of deep neural networks for deceptive review detection. In their method, they adopt CNN

[5] http://tripadvisor.com.
[6] http://mturk.com.

(convolutional neural network) to learn sentence representation from word vectors and then use GRNN (Gated Recurrent Neural Network) to learn document representation from sentence vectors. Actually, we can regard that the proposed approach DRI-CNN in the paper is a kind of deep neural network to identify deceptive reviews by making use of contextual information of words while Ren and Ji's method (Cagnina & Rosso, 2017) (hereafter in the paper we call it as GRNN–CNN) is to identify deceptive reviews by making use of contextual information of sentences. Their experiments on the deception dataset have shown that their GRNN–CNN model can achieve the best performance with accuracy as 0.8360 and F1 measure as 0.8340. Similar work in the aspect of automatic deceptive review identification includes Feng and Hirst (2013), Zhang et al. (2016) and Cagnina and Rosso (2015).

2.2. Text processing by deep neural network

Currently, there are usually three types of deep neural works involved in text processing as the Skip-gram model for word embedding, the recurrent neural network and the convolutional neural network for text representation.

2.2.1. Skip-gram model

Different from traditional vector space model (Salton, Wong, & Yang, 1975), where each word can be regarded as with a representation vector of a fixed length of the vocabulary of the documents in the corpus, word vectors are produced by learning from its surrounding words (sometimes we also call them contextual words) using neural network architecture and its lengths are variable according to specific application. More importantly, unlike traditional methods that more often than not suffer from the curse of dimensionality, the word vectors contain condensed continuous numbers as elements with much smaller length than the number of documents in the corpus (Collobert & Weston, 2008). By this way, the words in texts are embedded in distributed numerical vectors and similar words are represented with similar contexts mathematically.

Usually, the word vectors can be learned by using either continuous Bag-of-Words (CBOW) model or continuous Skip-gram model (Mikolov, Sutskever, Chen, Corrado, & Dean, 2013). For simplicity and computation efficiency, the Skip-gram model is state-of-the-art model adopted by many tasks in natural language processing. For brevity, the Skip-gram model can be depicted in Fig. 1. Here, we use the word $w(t)$ to predict its 5 maximum distance words ($w(t-2)$, $w(t-1)$, $w(t+1)$, $w(t+2)$) where $w(t)$ denotes the word at t position in a document. Note that $w(t-2)$,…, $w(t+2)$ are all continuous numerical vectors and the projection can be regarded as a neural network with one layer of hidden units.

The goal of the model training is to tune the word vectors that can be used to predict the surrounding words in a sentence. That is, given a sequence of training words $w_1, w_2, \ldots, w_T$, the Skip-gram needs to maximize the average log probability

$$\frac{1}{T}\sum_{t=1}^{T}\sum_{-c\le j\le c}\log p(\omega_{t+j}|\omega_t). \tag{1}$$

here, c is the maximum distance from the word ω_t. The larger is c, the more training examples are involved in the model leading to a higher accuracy of the model with more computation complexity. The basic Skip-gram formulation defines $p(\omega_{t+j}|\omega_t)$ using the softmax function in Eq. (2) where $v'(\omega_w)$ is the output word vector for the word ω_w and W is the size of the vocabulary. In traditional Skip-gram model, there are two word vectors as input word vector $v(\omega_w)$ and output word vector $v'(\omega_w)$ for the word ω_w. In practice, it is very difficult to maximize Eq. (1) if W is huge because, at each run of gradient descent, it will involve updating $O(W)$ parameters. Thus, negative sampling (Mikolov, Chen, Corrado, & Dean, 2013) is employed to solve this problem to compute $p(\omega_{t+j}|\omega_t)$ within a

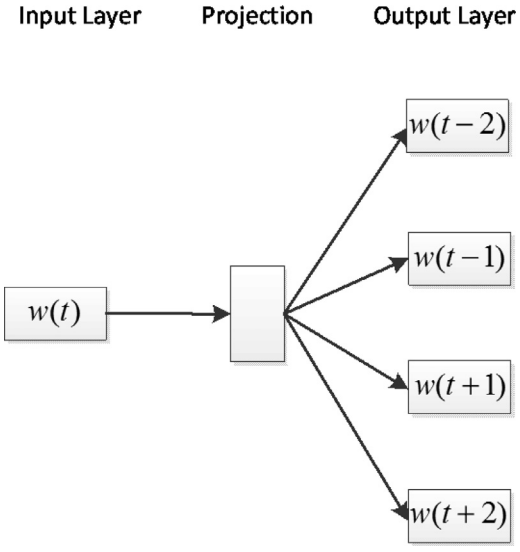

Fig. 1. The Skip-gram model.

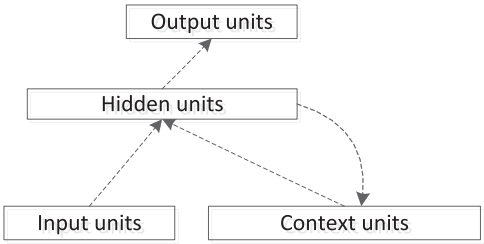

Fig. 2. The recurrent neural network.

scope of k native samples. Usually, k is in the range from 5 to 20 are useful for small training datasets, while for large datasets the k can be as small as from 2 to 5.

$$p(\omega_{t+j}|\omega_t) = \frac{\exp(v'(\omega_{t+j})^T v(\omega_t))}{\sum_{w=1}^{W} \exp(v'(\omega_w)^T v(\omega_t))}. \quad (2)$$

2.2.2. Recurrent neural network

Recurrent neural network (RNN) is proposed by Elman (Jeffery, 1990) to provide traditional neural network with a dynamic memory. Under RNN, the patterns of hidden units are feedback to context units as input to memorize the prior internal states as shown in Fig. 2. Their experiments on XOR problem, character prediction on word spelling and word prediction on sentence composition demonstrate that RNN can memorize context of each input and learn the temporal structure from input sequence which can be used to determine boundaries of linguistic units for word prediction (Jeffery, 1990). In the paper, we adopt RNN to capture of word context from deceptive and truthful reviews, respectively. The basic idea here is that words and its contexts used deceptive and truthful reviews are different from each other because of the difference between imaginative and real experiences of review writers.

Although structures of sentences can be expressed explicitly by PCFG tree (Collins, 2013) and then recursive neural network (Recursive NN) (Socher, Huang, Pennington, Ng, & Manning, 2011) can be used to capture both syntactical and semantic information of word expressions, we do not adopt this method in this paper to mix the syntactical and semantic information of words together in the same representation. We regard that context information is different from syntactical information because the inherent meaning of words are dependent not only on the syntactical structure they lies in but also other surrounding words, tense and punctuations of the sentence. For this reason, we adopt RNN to capture the context information of words and word embedding to capture the semantic information of words separately in the paper and concatenate them to produce the complete word vectors for deceptive review identification.

2.2.3. Convolutional neural network

Convolutional neural network (CNN) is firstly proposed by Waibel, Hanazawa, Hinton, Shikano, and Lang (1989) for phoneme recognition and also called time-delay neural networks (TDNNs) in the literature. The basic idea behind CNN is to produce transferred intermediate representations (i.e. convolution operation) for network training using local neighboring inputs (i.e. local region view) of previous layer. In natural language processing, CNN is usually used to capture the global meaning of a sentence by combining the meaning of its local components (Collobert et al., 2011). When we want to learn an internal representation for syntactic units such as phrases, N-grams and multi-words (Socher et al., 2011), Recursive NN is more often used than CNN. However, when we want to learn a global representation for a whole sentence, CNN is more often used than Recursive NN with its global effects across all words in the sentence (Kalchbrenner & Blunsom, 2013).

CNN is adopted to extract principal components from the complete word embedding for review representation as its success in image recognition (Ciresan, Meier, Masci, Gambardella, & Schmidhuber, 2011) and text classification (Lai, Xu, Liu, & Zhao, 2015). In this paper, we adopt CNN in two aspects. The first is to combine the context vectors and word vectors using $W^{(l,\,*)}$, $W^{(r,\,*)}$ and $W^{(sl,\,*)}$, $W^{(sr,\,*)}$, respectively, to produce a global representation as recurrent convolutional vector for word x_p. That is, the left context is learned with a local view on the left context vector and word vector in embedding of previous word and, the right context is learned with a local view on the right context vector and word vector in embedding of next word. The second is to use max-pooling to combine all word representations in a review to produce the review representation $y_i^{(3)}$. Note that max-pooling has been adopted by Lai et al. (2015) and Collobert et al. (2011) successfully in text classification. Essentially, deceptive review identification can be regarded as a special application in text classification. However, the former problem is more challenging than the later one because, most texts in text classification task are news, novels, poems that are not difficult for human beings to identify their categories if being carefully studied. Nevertheless, the deceptive reviews are written by human beings purposely to mislead its readers for business benefits. For instance, it is not hard for an expert to decide the sentiment polarity of a movie review in Lai et al. (2015) but, even for an expert, he or she cannot identify truthful reviews from deceptive reviews easily (Ott et al., 2011).

3. The proposed approach – DRI-RCNN

The proposed DRI-RCNN works as follows. First, we use Skip-gram model to embedding word into two numeric vectors: deceptive

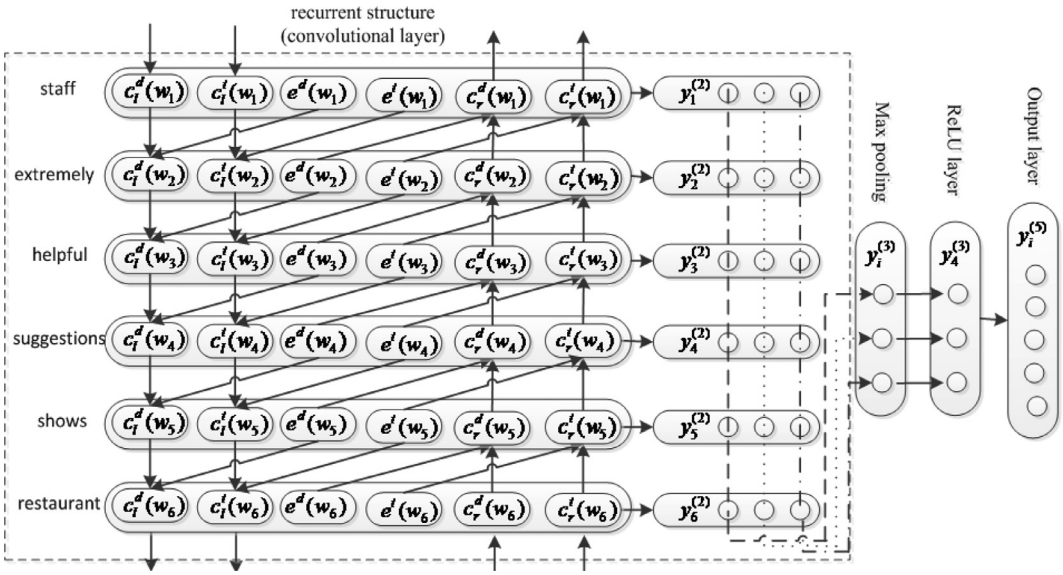

Fig. 3. The overall framework of the DRI-CNN approach.

vector and truthful vector. Second, the contextual information of words is learned by using a recurrent and convolutional network. We divide the contexts of words into left context and right context. In learning the left contextual information of a word, we use left context vector and vector in embedding of its left previous word. By analogy, in learning the right contextual information of a word, we use right context vector and vector in embedding of its right next word. Thus, a recurrent network with local view is adopted to learn the complete representation for the word. We call this complete representation of a word w_p as a recurrent convolutional vector x_p in Section 3.1. Third, max-pooling is used to extract maximum elements of features to compose words' recurrent convolutional vectors to compose review representation. Then, ReLU filter is used to remove the possible negative elements in the review representation. Finally, a fully connected network with back propagation is used to predict labels (deceptive or truthful) of the review and the softmax function is used to transfer prediction values into probabilities.

The framework of the proposed DRI-RCNN approach is depicted in Fig. 3. It can be seen that the overall structure of the DRI-RCNN approach comprises four layers as convolutional layer, max-pooling layer, ReLU layer and output layer. The convolutional layer is adopted to learn the complete vector for word representation. A recurrent neural network is adopted here to learn the left and right deceptive context vectors, the left and right truthful context vectors of words. The max-pooling layer is used to transfer recurrent convolutional vectors of all words in a review into a review vector. The ReLU layer is used to filter out those negative elements in the review vector. The reason to use ReLU layers as hidden layers is that ReLU, i.e. max $(0, x)$, is a highly useful activation function that doesn't saturate on shallow gradients as sigmoid activation functions do (Nair & Hinton, 2010). The output layer is a fully connected network used to predict the label of the review.

We can see from Fig. 3 that the DRI-RCNN approach is different from the RCNN approach in two sides. The first side is that we learn two types of contextual information of words in reviews while Lai et al. (2015) merely learn merely one type of contextual information of words in texts. The basic ideas of the DRI-RCNN approach and the RCNN approach are different in that we regard that each word has two types of contextual information: one is in the truthful reviews and the other one is in the deceptive reviews. However, Lai et al. (2015) regard that the contextual information of a word in the whole text collection is the same one. The second side is that compared with the deep neural network devised in Lai et al. (2015), we further adopt a hidden layer with a simple ReLU activation function to filter out the possible negative components in review representation. The basic idea here is quite straightforward. That is, it is very difficult to give an intuitive explanation for negative elements in representation space corresponding to text representation. We will explain the details of the DRI-RCNN approach in the following subsections.

Actually, the words in deceptive reviews and the words in truthful reviews are different from each other in its occurrence frequency as well as its neighboring words. If word embedding is adopted as concatenated vectors from truthful and deceptive reviews, we find that the words frequently occurring in deceptive reviews but infrequently occurring in truthful reviews are very near to each other in the deceptive embedding space. However, the words infrequently occurring in deceptive reviews are very far from each other in the deceptive embedding space and it is the same case for those frequent and infrequent words in the truthful views. If RCNN (Lai et al., 2015) is adopted, we cannot differentiate those frequent words and infrequent words in truthful reviews and deceptive reviews clearly because the representation in RCNN treats all words equally without considering the words' occurrences in deceptive or truthful reviews. In this point of view, we may say that the concatenated vectors in the DRI-RCNN approach encode both deceptive and truthful contextual information of words to enrich its representation space. By this kind of representation space enrichment, the accuracy of deceptive review identification is improved (see Section 4.3) though it is hard to analyze the direct causality between the space enrichment and the accuracy improvement theoretically. Nevertheless, we can regard that each review in

DRI-RCNN is represented by two separate vectors, i.e., one is by aggregating deceptive word vectors and the other one is by aggregating truthful word vectors. By pairwise learning (Fürnkranz et al., 2013) from the paired representations of reviews, the neural network can be more effective in identifying deceptive reviews.

3.1. Usage of the network

We define $c_i^{(d)}(w_p)$ in Eq. (1) as the left context vector of word w_p from deceptive reviews and its length is $|c|$. $c_i^{(d)}(w_p)$ is derived by combining the left context vector of its left word w_{p-1}, i.e. $c_l^{(d)}(w_{p-1})$ and its word vector in embedding, i.e. $e^{(d)}(w_{p-1})$. By analogy, $c_l^{(t)}(w_p)$ is defined as the left context vector of word w_p from truthful reviews in Eq. (2). $c_r^{(d)}(w_p)$ is defined as the right context vector of word w_p from deceptive reviews in Eq. (3) and $c_r^{(t)}(w_p)$ is defined as the right context vector of word w_p from truthful reviews in Eq. (4).

$$c_l^{(d)}(w_p) = f\left(W^{(l,d)}c_l^{(d)}(w_{p-1}) + W^{(sl,d)}e^{(d)}(w_{p-1}) + b_l^{(d)}\right) \tag{1}$$

$$c_r^{(t)}(w_i) = f\left(W^{(r,t)}c_r^{(t)}(w_{i-1}) + W^{(sr,t)}e^{(t)}(w_{i-1})\right) \tag{2}$$

$$c_r^{(d)}(w_p) = f\left(W^{(r,d)}c_r^{(d)}(w_{p+1}) + W^{(sr,d)}e^{(d)}(w_{p+1}) + b_r^{(d)}\right) \tag{3}$$

$$c_r^{(t)}(w_p) = f\left(W^{(r,t)}c_r^{(t)}(w_{p+1}) + W^{(sr,t)}e^{(t)}(w_{p+1}) + b_r^{(t)}\right) \tag{4}$$

Then, for each word w_p in a review D_i, it is represented by combing its left context vectors $c_i^{(d)}(w_p)$ and $c_i^{(t)}(w_p)$, word vectors in embedding $e^{(d)}(w_p)$ and $e^{(t)}(w_p)$, and right context vectors $c_r^{(d)}(w_p)$ and $c_r^{(t)}(w_p)$ as a recurrent convolutional vector x_p as in Eq. (5)_.

$$x_p = [c_l^{(d)}(w_p); c_l^{(t)}(w_p); e^{(d)}(w_p); e^{(t)}(w_p); c_r^{(t)}(w_p); c_r^{(d)}(w_p)] \tag{5}$$

Here, we define f as a continuously differentiable sigmoid function, that is, $f(x) = \frac{1}{1+e^{-x}}$ and its derivative can be easily derived as $f'(x) = (1 - f(x))f(x)$. We also apply an S-shape function as tanh non-linearity function to squashes the word vector x_p to range between $[-1,1]$ in $y_p^{(2)}$ as in Eq. (6). Note that the derivative function of tanh is $(\tanh(x))' = 1 - (\tanh(x))^2$.

$$y_p^{(2)} = \tanh(W^{(2)}x_p + b^{(2)}) \tag{6}$$

For all the normalized word vectors $y_p^{(2)}$ in a review D_i, we select the maximum elements in each row of all $y_p^{(2)}$ to obtain the review vector as in Eq. (7). Here, $|D_i|$ is the number of words in the review D_i. For instance, assuming that a review D_i contains 4 words as w_1, w_2, w_3 and w_4. Each word is represented by a vector of length 3 as $y_{i,1}^{(2)} = (0.1, 0.2, 0.3)^T$, $y_{i,2}^{(2)} = (0.2, 0.5, 0.4)^T$, $y_{i,3}^{(2)} = (0.3, 0.2, 0.1)^T$ and $y_{i,4}^{(2)} = (0.4, 0.2, 0.1)^T$.Then, the document vector for the review D_i, i.e. $y_i^{(3)}$, is produced as $y_i^{(3)} = (0.4, 0.5, 0.4)^T$.

$$y_i^{(3)} = \max_{p=1}^{|D_i|} y_p^{(2)} \tag{7}$$

After obtaining review vector $y_i^{(3)}$, we introduce a hidden layer with ReLU activation function to avoid the possible negative elements in the review vector as in Eq. (8). The transformation in Eq. (8) is quite straightforward, i.e., to make all the elements in the document vector larger than zero by using a ReLU filter.

$$y_i^{(4)} = \max(0, W^{(4)}y_i^{(3)} + b^{(4)}) \tag{8}$$

Eq. (9) is the output layer with a simple linear function to summarize all the inputs from the hidden layer to obtain the final output vector $y_i^{(5)}$ for review D_i. The size of the output vector $y_i^{(5)}$ corresponds to the number of possible categories for review D_i. In the case of deceptive review identification, the size of $y_i^{(5)}$ is 2 because review D_i is either deceptive or truthful.

$$y_i^{(5)} = W^{(5)}y_i^{(4)} + b^{(5)} \tag{9}$$

Finally, softmax function is adopted to transfer each element $y_{i,l_q}^{(5)}$ of real number in the vector $y_i^{(5)}$ into a probability vector p_{i,l_q}, where l_q is a possible label for the review D_i in Eq. (10). Note that $p_{i,Label(D_i)}$ is the probability of review D_i being labeled with its correct $Label(D_i)$. In the specific case of deceptive review identification, there are only two ($L = 2$ in Eq. (10) possible labels for each review, i.e. deceptive or truthful ($l_1 = deceptive$ and $l_2 = truthful$). Then, p_{i,l_1} is the probability that the review D_i being labeled with deceptive and p_{i,l_2} is the probability that the review D_i being labeled with truthful.

$$p_{i,l_q} = \frac{\exp\left(y_{i,l_q}^{(5)}\right)}{\sum_{q=1}^{L}\exp\left(y_{i,l_q}^{(5)}\right)} \tag{10}$$

The loss function we set for network training is shown in Eq. (11) where $Loss_i$ is defined as $-\ln(p_{i,Lable(D_i)})$. We can draw from Eq. (10) that $p_{i,Lable(D_i)}$ denotes the probability of review D_i being labeled with its correct label $Lable(D_i)$. That is to say, we use the output probability $p_{i,Lable(D_i)}$ from the network to measure its loss. Here, $p_{i,Lable(D_i)}$ means that for a given review D_i, to how much probability it will be labeled with its correct label $Lable(D_i)$ amongst all the possible labels l_q. If $p_{i,Lable(D_i)}$ is near to 1, which means the network

predict the label of D_i approximately correctly, then the loss on D_i, i.e. $Loss_i$, will be near to zero. On the opposite case, if $p_{i,Lable(D_i)}$ is near to zero, then $Loss_i$ will be a large number.

$$\begin{aligned} Loss = \frac{1}{N}\sum_{i=1}^{N} Loss_i + \frac{1}{2}\lambda\Bigg(& \sum_{i,j}(W_{ij}^{(5)})^2 + \sum_{i,j}(W_{ij}^{(4)})^2 + \sum_{i,j}(W_{ij}^{(2)})^2 \\ & + \sum_{i,j}(W_{ij}^{(r,t)})^2 + \sum_{i,j}(W_{ij}^{(sr,t)})^2 + \sum_{i,j}(W_{ij}^{(r,d)})^2 + \sum_{i,j}(W_{ij}^{(sr,d)})^2 \\ & + \sum_{i,j}(W_{ij}^{(l,t)})^2 + \sum_{i,j}(W_{ij}^{(sl,t)})^2 + \sum_{i,j}(W_{ij}^{(l,d)})^2 + \sum_{i,j}(W_{ij}^{(sl,d)})^2\Bigg) \end{aligned} \tag{11}$$

For the remaining component of the loss function *Loss* as $\frac{1}{2}\lambda\left(\sum_{i,j}(W_{ij}^{(5)})^2 + ... + \sum_{i,j}(W_{ij}^{(sl,d)})^2\right)$, we adopt the *L2*-regularization to smooth the weights of the input and hidden layers for each neuron to alleviate the possible overfitting where λ is the regulation coefficient to avoid overfitting (Ng, 2004). The ideal case is that all the elements in $W_{ij}^{(*)}$ can have a small value to make use of all inputs and hidden nodes in the network to decide the final output together and, the bad case is that only a small number of elements in $W_{ij}^{(*)}$ dominate the whole weights and all the other elements are zeros.

3.2. Gradient descent

The parameters in the network are $\theta = \{W^{(l,\ d)}, W^{(sl,\ d)}, b_l^{(d)}, W^{(l,\ t)}, W^{(sl,\ t)}, b_l^{(t)}, W^{(r,\ d)}, W^{(sr,\ d)}, b_r^{(d)}, W^{(r,\ t)}, W^{(sr,\ t)}, b_r^{(t)}, W^{(2)}, b^{(2)}, W^{(4)}, b^{(4)}, W^{(5)}, b^{(5)}\}$, where $\{W^{(l,\ d)}, W^{(2)}, W^{(r,\ d)}, W^{(r,\ t)}\} \in R^{c \times c}$, $\{W^{(sl,\ d)}, W^{(l,\ t)}, W^{(sr,\ d)}, W^{(sr,\ t)}\} \in R^{c \times e}$, $\{b_l^{(d)}, b_l^{(t)}, b_r^{(d)}, b_r^{(t)}\} \in R^c$, $W^{(2)} \in R^{H_2 \times (2e+4c)}$, $b^{(2)} \in R^{H_2}$, $W^{(4)} \in R^{H_4 \times H_2}$, $b^{(4)} \in R^{H_4}$, $W^{(5)} \in R^{O \times H_4}$, $b^{(5)} \in R^O$.

Here, e is the size of word vector in word embedding derived from the skip model and c is the predefined size of contextual vector. The target of the network training is to minimize the loss function *Loss* in order to maximize the probability of data sample D_i to be predicted with its correct label $Label(D_i)$ meanwhile reducing the possible overfitting of the trained network. We use the formula $\theta \leftarrow \theta + \alpha\frac{\partial Loss}{\partial \theta}$ to update the parameters in θ iteratively where α is the learning rate and we set it as 0.001 in the learning process to control the step size. λ is the weight used to tradeoff the loss of inaccurate prediction and the loss of possible overfitting and we set it as 0.5 in the learning process. We use the chain rule to derive all the gradients of the parameters as follows. For computation simplicity, we divide the loss function as Eq. (11) into two parts. That is $Loss = Loss^{(1)} + Loss^{(2)}$, $Loss^{(1)} = \frac{1}{N}\sum_{i=1}^{N} Loss_i$ and $Loss^{(2)} = \frac{1}{2}\lambda\left(\sum_{i,j}(W_{ij}^{(5)})^2 + ... + \sum_{i,j}(W_{ij}^{(sl,d)})^2\right)$. For the ith element in the output vector $y^{(5)}$, the partial derivative $Loss_i$ with respect to $y_{i.l_q}^{(5)}$ is described in Eq. (12).

$$\frac{\partial Loss_i}{\partial y_{i,l_q}^{(5)}} = \frac{\partial\left(\ln\left(\sum_{k=1}^{L}\exp(y_{i,l_k}^{(5)})\right) - y_{i,l_q}^{(5)}\right)}{\partial y_{i,l_q}^{(5)}} = \frac{\exp\left(y_{i,l_q}^{(5)}\right)}{\sum_{k=1}^{L}\exp\left(y_{i,l_q}^{(5)}\right)} - 1\left(y_{(l_q)}\right) = p_{i,y_i} - 1\left(y_{l_q}\right) \tag{12}$$

Here, $1\left(y_{l_q}\right)$ denotes a vector that the element in the position corresponding to the category l_q of the label of review D_i is 1 and the elements in other positions are zeros. For instance, assuming that we have two categories as [0, 1] and, for a review D_i with label 1, it is predicted by the trained network as with probability $[0.3, 0.7]^T$ for each category [0, 1] respectively. Then, the gradient of the loss with respect to the category on D_i is $[0.3, -0.3]^T$. Thus, for all the reviews D_i ($1 \le i \le N$) and their categories l_q ($1 \le q \le L$), we have $\frac{\partial Loss}{\partial y^{(5)}} \in R^{O \times N}$ and the size of output vector O is equal to the number of categories L.

Following this line of thought, we firstly do not consider the *L2*-regulation component $Loss^{(2)}$ for computation simplicity and derive the gradients of the first part $Loss^{(1)}$ of the full loss function *Loss*, i.e. $Loss^{(1)}$ with respect to $W^{(5)}$, $y^{(4)}$ and $b^{(5)}$ as shown in Eqs. (13), (14) and (15), respectively. Note that $\frac{\partial Loss}{\partial y^{(5)}}$ is equal to $\frac{\partial Loss^{(1)}}{\partial y^{(5)}}$.

$$\nabla W^{(5)} = \frac{\partial Loss^{(1)}}{\partial W^{(5)}} = \frac{\partial Loss^{(1)}}{\partial y^{(5)}}\frac{\partial y^{(5)}}{\partial W^{(5)}} = \frac{\partial Loss^{(1)}}{\partial y^{(5)}} \cdot (y^{(4)})^T \tag{13}$$

Note that in Eq. (13), if $y_{i,j}^{(4)}$ is less than zero, then its gradient should be zero because $y_i^{(4)}$ is computed as the maximum number between 0 and $W^4 y_i^{(3)} + b^{(4)}$ in Eq. (8). Otherwise, its gradient can be derived from $(W^{(5)})^T\frac{\partial Loss^{(1)}}{\partial y^{(5)}}$ directly.

$$\frac{\partial Loss^{(1)}}{\partial y^{(4)}} = (W^{(5)})^T\frac{\partial Loss^{(1)}}{\partial y^{(5)}} \text{ and } \frac{\partial Loss^{(1)}}{\partial y^{(4)}}\left[y_{i,j}^{(4)} \le 0\right] = 0 \tag{14}$$

Note that in Eq. (14), $b^{(5)} \in R^O$, and we sum over the elements in $\frac{\partial Loss^{(1)}}{\partial y^{(5)}}$ by rows as the gradient of the loss function with respect to the bias vector $b^{(5)}$.

$$\nabla b^{(5)} = \frac{\partial Loss^{(1)}}{\partial b^{(5)}} = \sum_{i=1}^{N} \left(\frac{\partial Loss^{(1)}}{\partial y^{(5)}} \right)_{(:,i)} \frac{\partial y^{(5)}}{\partial b^{(5)}} = \sum_{i=1}^{N} \left(\frac{\partial Loss^{(1)}}{\partial y^{(5)}} \right)_{(:,i)} \tag{15}$$

Similarly, we can derive the gradients of the partial loss function $Loss^{(1)}$ with respect $\nabla W^{(4)}$, $\frac{\partial Loss^{(1)}}{\partial y^{(3)}}$ and $\nabla b^{(4)}$ as Eqs. (16), (17) and (18), respectively.

$$\nabla W^{(4)} = \frac{\partial Loss^{(1)}}{\partial W^{(4)}} = \frac{\partial Loss^{(1)}}{\partial y^{(4)}} \frac{\partial y^{(4)}}{\partial W^{(4)}} = \frac{\partial Loss^{(1)}}{\partial y^{(4)}} \bullet (y^{(3)})^T \tag{16}$$

$$\frac{\partial Loss^{(1)}}{\partial y^{(3)}} = (W^{(4)})^T \frac{\partial Loss^{(1)}}{\partial y^{(4)}} \tag{17}$$

$$\nabla b^{(4)} = \frac{\partial Loss^{(1)}}{\partial b^{(4)}} = \sum_{i=1}^{N} \left(\frac{\partial Loss^{(1)}}{\partial y^{(4)}} \right)_{(:,i)} \tag{18}$$

Note that in Eq. (7), $y_i^{(3)}$ is derived by aggregation of maximum elements of rows in $y_p^{(2)}$. That is, the document vector $y_i^{(3)}$ for review D_i is derived by combing the vectors $y_p^{(2)}$ of words in the review. Thus, for each review D_i, its gradient is computed as Eq. (19). Using the same example as used to describe Eq. (7), suppose the ith column in $\frac{\partial Loss^{(1)}}{\partial y^{(3)}}$ is $\left(\frac{\partial Loss^{(1)}}{\partial y^{(3)}} \right)_{(:,i)} = (0.5, 0.4, 0.2)^T$. In the beginning, we initialize the gradients as $\frac{\partial Loss^{(1)}}{\partial y^{(2)}} = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}$, and for the first row of matrix $(y_{i,1}^{(2)}, y_{i,2}^{(2)}, y_{i,3}^{(2)}, y_{i,4}^{(2)})$, its maximum element is from $y_{i,4}^{(2)}$, thus, $\frac{\partial Loss^{(1)}}{\partial y^{(2)}}[(1, 4)_{\max} = true] = \left(\frac{\partial Loss^{(1)}}{\partial y^{(3)}} \right)_{(1,i)} = 0.5$. By analogy, $\frac{\partial Loss^{(1)}}{\partial y^{(2)}}[(2, 2)_{\max} = true] = \left(\frac{\partial Loss^{(1)}}{\partial y^{3}} \right)_{(2,i)} = 0.4$, $\frac{\partial Loss^{(1)}}{\partial y^{(2)}}[(3, 2)_{\max} = true] = \left(\frac{\partial Loss^{(1)}}{\partial y^{(3)}} \right)_{(2,i)} = 0.4$. As a result, for the ith review D_i, we re-write $\frac{\partial Loss^{(1)}}{\partial y^{(2)}} = \begin{pmatrix} 0 & 0 & 0 & 0.5 \\ 0 & 0.4 & 0 & 0 \\ 0 & 0.4 & 0 & 0 \end{pmatrix}$.

$$\frac{\partial Loss^{(1)}}{\partial y^{(2)}} = 0 \text{ and } \left(\frac{\partial Loss^{(1)}}{\partial y^{(2)}} \right)[(:,p)_{\max} = true] = \left(\frac{\partial Loss^{(1)}}{\partial y^{(3)}} \right)_{(:,i)} \tag{19}$$

Note that in Eq. (6), x_p represents the numeric vector of the pth term in review D_i. y_p is the transferred vector of x_p with elements in range $[-1, 1]$ by using tanh activation function. Thus, it is easy to derive that the pth column in final $\frac{\partial Loss^{(1)}}{\partial y^{(2)}}$ is equal to $\frac{\partial Loss^{(1)}}{\partial y_p^{(2)}}$ of review D_i. For better illustration, we aggregate all x_p in the review D_i by columns as a matrix X_i. Then, Eq. (6) can be rewritten as $y^{(2)} = \tanh(W^{(2)} X_i + b^{(2)})$. Similarly, we can derive the gradients of the partial loss function $Loss^{(1)}$ with respect to $W^{(2)}$, x_p and $b^{(2)}$ as Eqs. (20), (21) and (22), respectively. Note that the derivative function of tanh is $(\tanh(x))' = 1 - (\tanh(x))^2$. The notation I denotes an identity matrix and $diag((y^{(2)})^T y^{(2)})$ is to make a diagonal matrix using the elements in the main diagonal of matrix $(y^{(2)})^T y^{(2)}$.

$$\nabla W^{(2)} = \frac{\partial Loss^1}{\partial W^{(2)}} = \frac{\partial Loss^1}{\partial y^{(2)}} \frac{\partial y^{(2)}}{\partial W^{(2)}} = \frac{\partial Loss^1}{\partial y^{(2)}} (I - diag((y^{(2)})^T y^{(2)})) X_i^T \tag{20}$$

$$\frac{\partial Loss^1}{\partial X_i} = \frac{\partial Loss^1}{\partial y^{(2)}} \frac{\partial y^{(2)}}{\partial X_i} = (W^{(2)})^T \frac{\partial Loss^1}{\partial y^{(2)}} (I - diag((y^{(2)})^T y^{(2)})) \tag{21}$$

$$\nabla b^{(2)} = \frac{\partial Loss^1}{\partial b^{(2)}} = \sum_{p=1}^{|D_i|} \left(\frac{\partial Loss^1}{\partial y^{(2)}} (I - diag((y^{(2)})^T y^{(2)}))_{(:,p)} \right) \tag{22}$$

It can be drawn from Eq. (21) that the pth column $\frac{\partial Loss^{(1)}}{\partial X_i}$ is $\frac{\partial Loss^{(1)}}{\partial x_p}$, i.e. $\left(\frac{\partial Loss^{(1)}}{\partial X_i} \right)_{:,p} = \frac{\partial Loss^{(1)}}{\partial x_p}$. Note that the recurrent convolutional vector x_p for w_p is combined by six parts as the left deceptive context vector $c_l^{(d)}(w_p)$, the left truthful context vector $c_l^{(t)}(w_p)$, the deceptive word vector in embedding $e^{(d)}(w_p)$, the truthful word vector in embedding $e^{(t)}(w_p)$, the right deceptive context vector $c_r^{(d)}(w_p)$ and the right truthful context vector $c_r^{(t)}(w_p)$ as described in Eq. (5). Thus, we can easily derive $\frac{\partial Loss^{(1)}}{\partial c_l^{(d)}(w_p)} = \left(\frac{\partial Loss^{(1)}}{\partial x_p} \right)_{(1:c,:)}$, $\frac{\partial Loss^{(1)}}{\partial c_l^{(t)}(w_p)} = \left(\frac{\partial Loss^{(1)}}{\partial x_p} \right)_{(c:2c,:)}$, $\frac{\partial Loss^{(1)}}{\partial c_r^{(t)}(w_p)} = \left(\frac{\partial Loss^{(1)}}{\partial x_p} \right)_{(2e+2c:2e+3c,:)}$ and $\frac{\partial Loss^{(1)}}{\partial c_r^{d}(w_p)} = \left(\frac{\partial Loss^{(1)}}{\partial x_p} \right)_{(2e+3c:2e+4c,:)}$. Thus, the gradients of the partial loss function $Loss^{(1)}$ with respect to weights $W^{(l,\,d)}$ of left deceptive context vectors, the weights $W^{(sl,\,d)}$ of word vectors in embedding and the bias vector $b_l^{(d)}$ are computed as Eqs. (23), (24) and (25). Note that for the sigmoid function $f(x)$, its derivative function $f'(x)$ is $f'(x) = (1 - f(x)) f(x)$.

$$\nabla W^{(l,d)} = \frac{\partial Loss^1}{\partial W^{(l,d)}} = \frac{\partial Loss^1}{\partial c_l^{(d)}(w_p)} (I - diag(c_l^{(d)}(w_p)) c_l^{(d)}(w_p)) c_l^{(d)}(w_{p-1})^T \tag{23}$$

$$\nabla W^{(sl,d)} = \frac{\partial Loss^1}{\partial W^{(sl,d)}} = \frac{\partial Loss^1}{\partial c_l^{(d)}(w_p)}(I - diag(c_l^{(d)}(w_p))c_l^{(d)}(w_p))e^{(d)}(w_{p-1})^T \tag{24}$$

$$\nabla b_l^{(d)} = \frac{\partial Loss^1}{\partial b_l^{(d)}} = \sum_{i=1}^{|D_i|}\left(\frac{\partial Loss^1}{\partial c_l^{(d)}(w_p)}(I - diag(c_l^{(d)}(w_p))c_l^{(d)}(w_p))\right)_{(:,i)} \tag{25}$$

Similarly, the gradients of the partial loss function $Loss^{(1)}$ with respect to $\{W^{(l,\,t)}, W^{(sl,\,t)}, b_l^{(t)}\}$, $\{W^{(r,\,d)}, W^{(sr,\,d)}, b_r^{(d)}\}$ and $\{W^{(r,\,t)}, W^{(sr,\,t)}, b_r^{(t)}\}$ can be derived in the same manner as that of the Eqs. (23), (24) and (25).

Considering the second part $Loss^{(2)}$ of the loss function L in Eq. (11), we find that the $L2$-regulation as $Loss^{(2)}$ is only relevant with the weight matrices $W^{(*,\,*)}$. Thus, for each weight matrix $W_{i,j}^{(*,*)}$, its final gradient should be $\nabla W^{(*,*)} = \nabla W^{(*,\,*)} + \lambda W^{(*,*)}$, where $W^{(*,\,*)}$ represents all those weight matrices in Eq (11).

3.3. Network training

We use the "Xavier" method (Xavier & Yoshua, 2010) to initialize the weight matrix for each $W^{(*,\,*)}$. That is, a uniform distribution as $W_{i,j}^{(*,*)} \sim U\left(-\frac{\sqrt{6}}{\sqrt{n_{in}+n_{out}}}, \frac{\sqrt{6}}{\sqrt{n_{in}+n_{out}}}\right)$ is used to initialize $W^{(*,\,*)}$ to ensure $Var(W^{(*,\,*)}) = \frac{2}{n_{in}+n_{out}}$, where n_{in} is the number of "fan-in" with respect to $W^{(*,\,*)}$ and n_{out} is the number of "fan-out" with respect to $W^{(*,\,*)}$ (Xavier & Yoshua, 2010). Each element in bias vectors b^* is initialized as a constant as 0.001. $c_l^{(d)}(w_p)$ is initialized as the average vector of the embedding $e_l^{(d)}(w_{p-1})$ of all the words occurring in the left positions of w_p in the deceptive reviews. By analogy, we initialize $c_r^{(d)}(w_p)$ as the average vector of the embedding $e_l^{(d)}(w_{p+1})$ of all words occurring in the right positions of w_p in the deceptive reviews. Using the same method, we conduct the initialization of $c_l^{(t)}(w_p)$ and $c_r^{(t)}(w_p)$. With the initialization of weight matrices $W^{(*,\,*)}$, the bias vectors b^*, and the context vectors, the learning procedures of network training can be depicted in Fig. 4.

Input: $\theta^{(0)} = \{W^{(l,d)(0)}, W^{(sl,d)(0)}, b_l^{(d)(0)}, W^{(l,t)(0)}, W^{(sl,t)(0)}, b_l^{(t)(0)}, W^{(r,d)(0)}, W^{(sr,d)(0)}, b_r^{(d)(0)}, W^{(r,t)(0)}, W^{(sr,t)(0)}, b_r^{(t)(0)}, W^{(2)(0)}, b^{(2)(0)}, W^{(4)(0)}, b^{(4)(0)}, W^{(5)(0)}, b^{(5)(0)}\}$;

$e^{(d)}(w_p)$, the deceptive word embedding of word w_p;

$e^{(t)}(w_p)$, the truthful word embedding of word w_p;

α=0.001, the learning rate of in training process;

λ=0.5, the ratio to trade off network prediction accuracy and overfitting;

rs, the number of random sampling size from the whole dataset;

maxIter, the maximum number of iterations in the training process;

Output: $\theta^{(*)} = \{W^{(l,d)(*)}, W^{(sl,d)(*)}, b_l^{(d)(*)}, W^{(l,t)(*)}, W^{(sl,t)(*)}, b_l^{(t)(*)}, W^{(r,d)(*)}, W^{(sr,d)(*)}, b_r^{(d)(*)}, W^{(r,t)(*)}, W^{(sr,t)(*)}, b_r^{(t)(*)}, W^{(2)(*)}, b^{(2)(*)}, W^{(4)(*)}, b^{(4)(*)}, W^{(5)(*)}, b^{(5)(*)}\}$;

Procedure:

For i from 1 to *maxIter*

Random sampling *rs* reviews $D^{(rs)}$ from the whole dataset;

Forward procedure:

1) Using Equations 1-10 to compute p_{i,l_q}, i.e., the probability of each review D_i belonging to the category l_q;
2) Using Equation 11 to compute the loss function $Loss^{(i)}$ at current iteration;

Backward procedure:

1) Using Equations 12-26 to compute the gradients of $\nabla\theta^{(i)} = \{\nabla W^{(l,d)(i)}, \nabla W^{(sl,d)(i)}, \nabla b_l^{(d)(i)}, W^{(l,t)(i)}, W^{(sl,t)(i)}, b_l^{(t)(i)}, W^{(r,d)(i)}, W^{(sr,d)(i)}, b_r^{(d)(i)}, W^{(r,t)(i)}, W^{(sr,t)(i)}, b_r^{(t)(i)}, W^{(2)(i)}, b^{(2)(i)}, W^{(4)(i)}, b^{(4)(i)}, W^{(5)(i)}, b^{(5)(i)}\}$;
2) Update $\theta^{(i+1)} = \theta^{(i)} + \alpha\nabla\theta^{(i)}$

End for

$\theta^{(*)} = \theta^{(maxIter)}$

Fig. 4. The algorithm to train the proposed DRI-RCNN network.

In the implementation, we use the method of stochastic gradient descent (SGD) (Bottou, 1998) on the randomly sampled dataset $D^{(rs)}$ in order to reduce the computation complexity. At each iteration, we only update a small part of $W^{(*,\ *)}$ and $b^{(*)}$ rather than the full weight matrices and bias vectors. The parameter *maxIter* should be set to ensure that the loss function is convergent to a stable level (i.e., $Loss^{(i+1)} - Loss^{(i)} < \varepsilon$ and ε is small number near to zero). In our case, we set the parameter *maxIter* as 1000.

The algorithm involves two subordinate procedures as the forward procedure and the backward procedure. In the forward procedure, we compute all the output vectors using the initialized or updated parameters in θ and the loss function *Loss* of network prediction. In the backward procedure, we compute the gradients of parameters in θ using the SGD method and update the parameters in θ. We can see from Eqs. (1) to (26) that most computation complexity of the algorithm is cost on matrix multiplication and the maximum cost is up to $O(N \times O \times Length(y^{(*)}))$ where N is the number of training samples, O is the number of output categories and $Length(y^{(*)})$ is size of input layers or hidden layers. For instance, in Eq. (13), the number of matrix multiplication is $N \times O \times H_4$ to derive the weight matrix $W^{(5)}$.Thus, it is possible that we can run these matrix multiplications in parallel by using block matrix multiplication and the framework of MapReduce (Dean & Ghemawat, 2004) with cluster computing. However, this is another interesting topic in deep learning and is out of the current scope of the paper. Here, we simply adopt random sampling to reduce the computation complexity by reducing the number of training samples N in the algorithm. In the future work, we will make use of distributed algorithms under the framework of MapReduce to compute the gradients involved in the parameter set θ.

4. Experiments

4.1. The datasets

We use two datasets in the experiments to examine the performances of the proposed DRI-RCNN approach in deceptive review identification. The one is the spam dataset from Ott et al. (2011) and the other is the deception dataset from Li et al. (2014). For each review, we conduct sentence boundary determination, part-of speech analysis and stop-word elimination. We use the algorithm proposed by Nitin, Fred, and Zhang (2005) to conduct sentence boundary determination for each review. We obtain the stop-words from USPTO (United States Patent and Trademark Office) patent full-text and image database.[7] We use the same stop-word list to train the Skip-gram model. After that, we aggregate all the sentences in the deceptive reviews to train the deceptive word vector $e^{(d)}(w_p)$ in embedding and the sentences in the truthful reviews to train the truthful word vector $e^{(t)}(w_p)$ in embedding, respectively. For the spam dataset, we use the training deceptive reviews and training truthful reviews in both positive and negative polarities as the input for the Skip-gram model to obtain the word vectors $e^{(d)}(w_p)$ and $e^{(t)}(w_p)$ in embedding. For the deception dataset, we use the training deceptive reviews and training truthful review in both subjects as the input for the Skip-gram model to obtain the deceptive word vectors in embedding as $e^{(d)}(w_p)$ and $e^{(t)}(w_p)$. Tables 1 and 2 show the basic information of reviews in the spam dataset and the deception dataset, respectively. Actually, the reviews under the subject “hotel” in the deception dataset are from two sources: one is a duplicate of the spam dataset and the other is added by experts’ reviews. For simplicity, we combine the experts’ reviews and turkers’ reviews as deceptive reviews.

4.2. Experiment setup

We adopt the DeepLearning4J open source platform[8] to implement the Skip-gram model for word embedding. After that, we produce the complete vector as shown in Eq. (5) and implement the gradient descent for network training as shown in Fig. 4 by Java programming.

The hyper parameters in the proposed DRI-RCNN approach are the length of word embedding e, the length of context vector c, the size of hidden layer in Eq. (6) as H_2, the size of hidden layer in Eq. (8) as H_4. In addition, the window size of word embedding in the Skip gram model is a crucial parameter to influence the performance of the proposed DRI-RCNN approach in deceptive review identification. A small window size in the Skip-gram model may bring about a loss of long-distance patterns, whereas a large window size would cause unnecessary even random co-occurrence patterns and data sparsity of vectors in computation.

There are no commonly accepted rules to set the hyper parameters of neural works appropriately. In fact, these hyper parameters are dependent on the dataset of the task. By trial and error, we adopt the 5-fold cross-validation method to tune these parameters. That is, we split the training set into 5 folds randomly. 4 folds are used to train the model and the remaining 1 fold is used to validate the parameter setting. This process is repeated 5 times by using each of the 5 folds for the validation set. Then the parameters are decided with the best average accuracy among the 5 repetitions. From our experiments, in both datasets, we observe that the parameters H_2 and H_4 are not as sensitive to the performances as the parameters as e, c and the window size. If the hyper parameters H_2 and H_4 are set as not less than 50 and 20 respectively, then the performances of the DRI-RCNN approach will be kept stable. For this reason, we set the hyper parameter H_2 as 50 and the hyper parameter H_4 as 20 of the neural network in the following experiments.

In real practice, the number of deceptive reviews is much smaller than that of truthful reviews. For this reason, the class-specific recall, precision and F1 measure is proposed to gauge the performance of deceptive review identification as an unbalanced classification problem. However, for the spam dataset and the deception dataset we used in this paper, the experimental data is purposely

[7] USPTO stop words, online: http://ftp.uspto.gov/patft/help/stopword.htm.

[8] Deeplearing4j: Open source, Distributed Deep Learning for the JVM: https://deeplearning4j.org/.

Table 1
The basic information of reviews in the spam dataset.

Polarity		No. of hotels	No. of reviews	No. of sentences	No. of unique words
Positive	Deceptive_from_MTurk	20	400	3043	7268
	Truthful_from_Web	20	400	3480	7300
Negative	Deceptive_from_MTurk	20	400	4149	7159
	Truthful_from_Web	20	400	4483	7512

Table 2
The basic information of reviews in the deception dataset.

Subject	Category	No. of reviews	No. of Sentences	No. of unique words
Doctor	deceptive_MTurk	356	2369	5128
	truthful	200	1151	5098
Hotel	deceptive_MTurk	1080	8463	16,635
	truthful	1080	9258	17,328
Restaurant	Deceptive_MTurk	201	1827	5136
	truthful	200	1892	5126

collected and, we can see from Tables 1 and 2 that the number of deceptive reviews and the number of truthful reviews are very close to each other. Thus, we use Accuracy and F1 as the performance measures in the experiments. Moreover, it is not easy to deduce the class-specific recall, precision and F1 from the given Accuracy and F1 measure for one of interest.

4.3. Results

4.3.1. Tuning the window size

Fig. 5 shows the accuracies of DRI-RCNN in deceptive review identification by using 90% of the data as the training set with 5-fold cross-validation. We vary the setting of the window size as well as the length e of word vectors in embedding and the length c of context vector. In the spam dataset, the average length of sentence is 15.64 and in the deception dataset, the average length is 15.16. Thus, we vary the window size from 3 to 8 to investigate the influence of window size on the performances of the DRI-RCNN approach. The reason here is that we assume that the length of a long-distance pattern would not exceed half of a sentence.

We can see from Fig. 5 that in both datasets, when the window size is set as 5, the DRI-RCNN approach maximize its performance on deceptive review identification in all settings of the length of word embedding e and the length of context vector c. This outcome indicates that in the experiments, the co-occurrence patterns in the reviews are usually within 5 words. This outcome is also consistent with the suggestion from Mikolov Chen, Corrado, and Dean (2013) that the window size in the Skip-gram model should be set from 2 to 5. For this reason, we set the window size as 5 in the following experiments.

We also see from Fig. 5 that different configurations of the length of word embedding e and the length of context vector c may result in different performances. In the spam dataset, we see that when we set e as 100 and c as 80, the DRI-RCNN approach optimize its performance at the window size as 5 among all the configurations. That is to say, a word in reviews of the spam dataset should be embedded with 100 features and its context of the word should be represented with 80 features. In the deception dataset, we see that when we set e as 80 and c as 60, the DRI-RCNN approach optimize its performance among all the configurations.

We explain that when the parameter e is small, the semantic relationship of words cannot be expressed completely without enough semantic space. We see that the absolute values of features in an embedding vector have a large difference from each other. However, when the parameter e is large, the semantic relationships of words are diffused to a large space and we see that the absolute values of features in an embedding vector have a small difference from each other. In the former case, words are very dissimilar with each other in the semantic space and to be worse, some words with random co-occurrence are regarded as having strong semantic relationship. In the latter case, words are regarded as similar to each other in the semantic space and this makes no difference in selecting the features to represent the reviews in Eq. (7). Essentially, the context vector is derived from the word embedding by using a recurrent structure. Thus, the parameter c is decided by the parameter e to some extent. We speculate that this is the reason why when the parameter c is adjusted to be larger than the parameter e, the performances of the DRI-RCNN approach are deteriorated.

With the above observation, we set the window size as 5 for both datasets. For the hyper parameter e, we set it as 100 in the spam dataset and 80 in the deception dataset. For the hyper parameter c, we set it as 80 in the spam dataset and 60 in the deception dataset for further experiments.

4.3.2. Comparison with state-of-the-art techniques

Tables 3 to 6 demonstrate the performances of DRI-RCNN when varying the training data from 50% to 90% and using the remaining data on the experiment datasets. We compare the DRI-RCNN approach with state-of-the-art techniques including LIWC

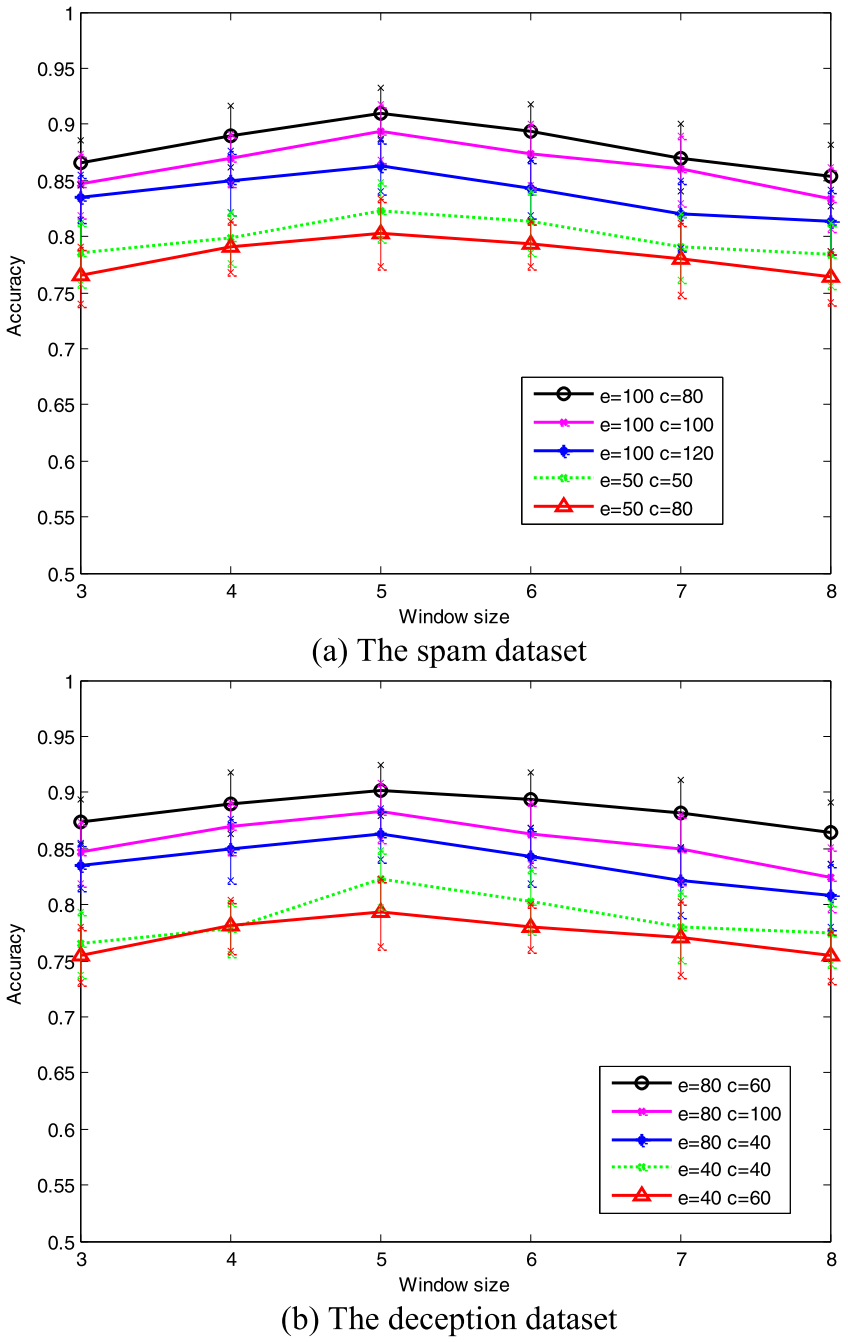

Fig. 5. The performances of the DRI-RCNN approach with different configurations of length *e* of word vector in embedding and length *c* of context vector in deceptive review identification.

features[9] and bigrams combined with SVM (Ott et al., 2011) (abbreviates as LB-SVM), LIWC features and four-grams combined with SVM (abbreviates as L4-SVM) (Cagnina & Rosso, 2015), unigram and lexicalized production rules combined with SVM model (Feng et al., 2012) (abbreviates as ULPR-SVM), profile alignment compatibility model (Feng & Hirst, 2013) (abbreviates as PACM), sparse additive generative model (abbreviates as SAGE) (Li et al., 2014), recurrent convolutional neural network (abbreviates as RCNN) (Lai et al., 2015) and gated recurrent neural network and convolutional neural network (abbreviates as GRNN–CNN) (Ren & Ji, 2017) . All parameters in these state-of-the-art techniques are set as suggested in the references with best performances. For

[9] LIWC2015, Online: http://liwc.wpengine.com/.

Table 3
The performances measured by average accuracy and its standard deviation (in the bracket) of the proposed DRI-RCNN approach in comparison with state-of-the-art techniques on the spam dataset. T.P. abbreviates for training percentage.

T.P.	0.5	0.6	0.7	0.8	0.9
DRI-RCNN	0.8293 (0.1003)	0.8415 (0.1433)	**0.8589 (0.1367)**	**0.8724 (0.1405)**	**0.8815 (0.1442)**
GRNN−CNN	0.8015 (0.1194)	0.8189 (0.0927)	0.8324 (0.1023)	0.8415 (0.1192)	0.8582 (0.1241)
RCNN	0.8003 (0.1024)	0.8039 (0.0873)	0.8124 (0.0814)	0.8321 (0.0981)	0.8351 (0.1106)
LB-SVM	0.8188 (0.1357)	0.8251 (0.1279)	0.8379 (0.1399)	0.8289 (0.1174)	0.8246 (0.1650)
ULPR-SVM	**0.8487 (0.1125)**	**0.8512 (0.1103)**	0.8558 (0.1550)	0.8609 (0.1581)	0.8571 (0.1541)
L4-SVM	0.8353 (0.0942)	0.8481 (0.1237)	0.8510 (0.1332)	0.8434 (0.1161)	0.8314 (0.1104)
SAGE	0.8037 (0.0938)	0.8101 (0.1058)	0.8232 (0.1042)	0.8310 (0.1248)	0.8317 (0.1113)
PACM	0.8289 (0.1195)	0.8471 (0.1149)	0.8523 (0.1447)	0.8591 (0.1217)	0.8600 (0.1205)

Table 4
The performances measured by F1 measure and its standard deviation (in the bracket) of the proposed DRI-RCNN approach in comparison with state-of-the-art techniques on the spam dataset. T.P. abbreviates for training percentage.

T.P.	0.5	0.6	0.7	0.8	0.9
DRI-RCNN	0.8123 (0.1153)	0.8301 (0.1116)	**0.8458 (0.1038)**	**0.8536 (0.1023)**	**0.8659 (0.1249)**
GRNN−CNN	0.8037 (0.1412)	0.8281 (0.1047)	0.8332 (0.1059)	0.8417 (0.1275)	0.8513 (0.0817)
RCNN	0.8001 (0.0834)	0.8013 (0.1036)	0.8104 (0.1011)	0.8123 (0.1063)	0.8206 (0.1124)
LB-SVM	0.8094 (0.1145)	0.8231 (0.1223)	0.8284 (0.1423)	0.8209 (0.1455)	0.8187 (0.1577)
ULPR-SVM	**0.8203 (0.1149)**	**0.8311 (0.1480)**	0.8438 (0.1270)	0.8312 (0.1493)	0.8210 (0.1212)
L4-SVM	0.8203 (0.1176)	0.8292 (0.1028)	0.8432 (0.1253)	0.8321 (0.1450)	0.8230 (0.1345)
SAGE	0.7920 (0.1280)	0.8019 (0.0974)	0.8163 (0.0969)	0.8223 (0.0875)	0.8261 (0.0829)
PACM	0.8010 (0.1175)	0.8217 (0.0998)	0.8442 (0.1226)	0.8336 (0.1108)	0.8438 (0.1137)

instance, as suggested by Ott et al. (2011), we use the linear kernel in SVM model training for deceptive review identification. We implement the SAGE model using Ark-SAGE.[10] Different from the setting in Li et al. (2014), for the spam dataset, the parameter y_{domain} does not exist because all the reviews belong to the hotel domain and the parameter y_{source} in Section 4.1 of Li et al. (2014) is rewritten as $y_{source} \in \{deceptive, truthful\}$. For the deception dataset, the parameter $y_{sentiment}$ does not exist because there no sentiment polarity for the reviews and the parameter y_{domain} is rewritten as $y_{domain} \in \{restaurant, hotel, doctor\}$ and the parameter y_{source} is rewritten as $y_{source} \in \{deceptive, truthful\}$. Note that here we only use the unigram feature in the SAGE model because it achieve the best performances in Li et al. (2014). For fair comparison, we merely use the first 4 procedures listed in Section 5.2.3 in Feng and Hirst (2013) to extract data for profile construction. For RCNN, we set the length of word embedding e and the length of context vector c as the setting for the DRI-RCNN in Section 4.3.1.

Tables 3 and 4 show the performances of the proposed DRI-RCNN approach in comparison with state-of-the-art techniques measured by average accuracy and F1 measure (Ren & Ji, 2017) on the spam dataset, respectively. We can see from Tables 3 and 4 that when training percentage is less than 0.7, ULPR-SVM performs the best among all the compared methods. When training percentage is equal to 0.7, the performance differences among DRI-RCNN, RCNN, ULPR-SVM, L4-SVM and PACM are not significant. When the percentage of the training set is not less than 0.8, the performances of the DRI-RCNN approach are better than all the state-of-the-art techniques. Moreover, when the percentage of the training set becomes 0.9, the performance superiority of the DRI-RCNN approach to other baseline methods is more significant than that at the percentage as 0.8.

We explain that when the size of the training set is small, the DRI-RCNN approach cannot have enough data for neural network model training. When more data are used to train the DRI-RCNN approach, the better performance it would produce in deceptive review identification. However, when we do not have enough data for training, the ULPR-SVM method is a good alternative for the task of deceptive review identification. We regard the reason here is that the lexicalized production rules are a good complement to characterize deceptive reviews and truthful reviews under the condition of lack of training data. Moreover, the PACM method is also acceptable when the size of training set is small despite its performances are not good as those of the ULPR-SVM method. In our experiments, we cannot observe that the accuracies of the PACM method can attain up to 0.913. We explain that the experimental data used in the paper is not the same as that used in Feng and Hirst (2013). We remove the last two procedures in data preparation as suggested by Feng and Hirst (2013) for fair comparison and this makes the data quality used for profile construction is not good as that used in their experiments. This outcome also reminds us that the PACM cannot tolerate low-quality data for training and, if the training data is not elaborately picked to make sure its high quality, then the PACM method would not take its effect to the greatest length. The better performances of DRI-RCNN than RCNN validates that the separation of deceptive context and truthful context is beneficial to deceptive review identification.

The disadvantage of the SAGE method is that it is very sensitive to occasional co-occurrences of words such as “floor, wife” and “bath, August” as a generative model. When the training set is small, it is very difficult to estimate the probabilities appropriately in

[10] Ark-SAGE, Online:https://bitbucket.org/skylander/ark-sage/.

Table 5
The performances measured by average accuracy and its standard deviation (in the bracket) of the proposed DRI-RCNN approach in comparison with state-of-the-art techniques on the deception dataset. T.P. abbreviates for training percentage.

T.P.	0.5	0.6	0.7	0.8	0.9
DRI-RCNN	0.8083 (0.0943)	0.8151 (0.1479)	**0.8400 (0.1277)**	**0.8524 (0.1190)**	**0.8601 (0.1484)**
GRNN−CNN	0.7921 (0.0965)	0.8136 (0.1084)	0.8205 (0.1135)	0.8334 (0.1006)	0.8470 (0.0969)
RCNN	0.7832 (0.0864)	0.8036 (0.0843)	0.8115 (0.1035)	0.8216 (0.1106)	0.8365 (0.1036)
LB-SVM	0.7865 (0.0781)	0.8061 (0.1197)	0.8181 (0.1056)	0.7933 (0.1264)	0.7868 (0.0883)
ULPR-SVM	0.8053 (0.1201)	**0.8239 (0.1131)**	0.8331 (0.1427)	0.8333 (0.1545)	0.8290 (0.1474)
L4-SVM	0.7953 (0.1025)	0.8091 (0.0942)	0.8151 (0.1114)	0.8167 (0.1557)	0.8000 (0.0976)
SAGE	0.7986 (0.1213)	0.8010 (0.1169)	0.8124 (0.1298)	0.8182 (0.0939)	0.8174 (0.1421)
PACM	**0.8165 (0.1053)**	0.8210 (0.1381)	0.8303 (0.1102)	0.8293 (0.1187)	0.8189 (0.1309)

Table 6
The performances measured by F1 measure and its standard deviation (in the bracket) of the proposed DRI-RCNN approach in comparison with state-of-the-art techniques on the deception dataset. T.P. abbreviates for training percentage.

T.P.	0.5	0.6	0.7	0.8	0.9
DRI-RCNN	0.7824 (0.1234)	0.8032 (0.1142)	**0.8242 (0.1100)**	**0.8356 (0.1130)**	**0.8463 (0.1600)**
GRNN−CNN	0.8010 (0.1116)	0.8047 (0.1255)	0.8163 (0.0991)	0.8286 (0.1132)	0.8343 (0.0873)
RCNN	0.7903 (0.1003)	0.8003 (0.1137)	0.8138 (0.1001)	0.8200 (0.1013)	0.8313 (0.1020)
LB + SVM	0.8186 (0.1268)	0.8201 (0.1235)	0.8213 (0.1190)	0.8283 (0.1275)	0.8202 (0.0872)
ULPR + SVM	**0.8034 (0.1327)**	0.8112 (0.1311)	0.8143 (0.1275)	0.7953 (0.1457)	0.7892 (0.1301)
L4 + SVM	0.7932 (0.1338)	0.8013 (0.1220)	0.8212 (0.1262)	0.8103 (0.1192)	0.7803 (0.1320)
SAGE	0.7821 (0.1309)	0.7901 (0.0925)	0.7911 (0.1251)	0.7938 (0.1372)	0.7942 (0.1445)
PACM	0.7937 (0.1245)	**0.8212 (0.1069)**	0.8214 (0.1550)	0.8123 (0.0985)	0.8019 (0.0956)

Eqs. (1) and (2) in Li et al. (2014) due to data sparsity and ineffectiveness of smoothing techniques. This makes SAGE performing worse than other methods in identifying deceptive and truthful reviews. This performance of the GRNN−CNN approach suggests that words' contexts are more effective than sentences' contexts in deceptive review identification. We explain this outcome that on the one hand, the sequence order of sentences is not so important as words because there is smaller semantic transition in describing temporal and spatial information in the reviews by changing orders of sentences than by changing orders of words' occurrences. On the other hand, the reviews are written by different writers with difference writing behaviors in making sentences however, if writers had similar experiences on hotels, restaurants or doctors to review, they should use the same or similar words.

Tables 5 and 6 show the performances of the proposed DRI-RCNN approach in comparison with state-of-the-art techniques measured by average accuracy and F1 measure (Ren & Ji, 2017) on the deception dataset, respectively. Compared with the performances shown in Tables 3 and 4, all the methods in deceptive review identification on the deception dataset performs worse that on the spam dataset. We explain that except the "hotel" subject, the number of reviews in the deception dataset is relatively smaller than that in the spam dataset as shown in Tables 1 and 2. Nevertheless, for the "hotel" subject in the deception dataset, the types of writers include "expert", "turker" and "truthful", which is more complicated for deceptive review identification than that in the spam dataset. Similarly, when the training percentage is not less than 0.7, DRI-CNN performs best among all the compared methods. The ULPR-SVM and PACM methods also have produced better performances than other methods when the training percentage is less than 0.7.

4.4. Implications

From the above study, we see from Fig. 5 that the performance of our proposed DRI-RCNN approach to deceptive review identification is influenced by four parameters. The first one is the parameter e as the length of word vectors produced by the Skip-gram model. In our experiment, we see that if we set e as a large number, for instance, larger than 100 for the spam dataset or larger than 80 for the deception dataset, there is a great feature redundancy in the word vectors, i.e., some features have small diminutiveness on the words. On the contrary, if we set e as a small number, for instance, smaller than 50 for the spam dataset or smaller than 40 for the deception dataset, all the words are squeezed in a small feature space and it is also hard to differentiate them using the setting features. We roughly draw that if a dataset with larger number of unique words, the parameter e should be given a large number from our analysis with the experimental datasets. However, in real practice of using the proposed DRI-RCNN approach, it is better to tune the parameter e by step-wise error and trial or exhaustive search.

The second parameter is c as the length of context vector in the recurrent convolution vector. From the above experimental results, we see from Fig. 5 that the context vectors actually take effects in deceptive review identification and the parameter c should be less than the parameter e for better performance of the proposed DRI-RCNN approach. The third parameter is the percent of data for model training, i.e. training percentage. We see from Tables 3 to 6 that the more data used for model training, the better performance is with the proposed approach. When the training data size is small, the proposed approach performs worse than other

baseline methods in deceptive review identification. Here, we come up with the implication that the contextual knowledge of writers are of great importance in improving deceptive review identification. However, it needs a large amount training data to distill the contextual knowledge from online reviews. The fourth parameter is the window size used in the Skip-gram model for word embedding. We find that the window size should be set as 5 for better performance of the DRI-RCNN approach. This is finding is also consistent with the previous study by Mikolov Chen, Corrado, and Dean (2013). However, because the reviews and the texts used in the paper and the Mikolov et al. are comprised by English sentences, we are not sure this setting is appropriate for other language type such as Chinese and Japanese.

5. Concluding remarks

This paper proposes a novel approach called DRI-RCNN for deceptive review identification using deep learning. The Skip-gram model is adopted to obtain the word embedding in the reviews using words and its neighboring words. We regard that one word has 4 types of context information: left deceptive context $c_l^{(d)}(w_p)$, left truthful context $c_l^{(t)}(w_p)$, right deceptive context $c_r^{(d)}(w_p)$ and right truthful context $c_r^{(t)}(w_p)$. The recurrent convolutional neural network is adopted to capture the context of words by concatenating the local region information of the complete representation of words. For instance, we use left deceptive context of previous word $c_l^{(d)}(w_{p-1})$ and embedding of previous word $e^{(d)}(w_{p\text{-}1})$ to produce left deceptive context $c_l^{(d)}(w_p)$ of word w_p by the recurrent convolutional neural network. After deriving the complete representation of each word x_p in each review, we use max pooling and ReLU filter to produce review vector for each review. Then, a fully connected network is employed to classify the reviews as deceptive and truthful.

The contribution of the paper can be summarized as follows. First, we present a review on related work on deceptive review identification and using deep learning for text processing. Second, we propose the DRI-RCNN approach and its mathematical formulation of each component is described in the paper. We also describe the detailed procedures of gradient descent to train the proposed neural network using real data in practice. Third, we use two datasets as the spam dataset and the deception dataset to compare the proposed DRI-RCNN approach with state-of-art techniques in deceptive review identification. Last but not less important, the experimental results on the two mentioned datasets demonstrate the superiority of the proposed DRI-RCNN approach to state-of-the-art methods.

In the future, the proposed DRI-RCNN approach will be improved in two aspects. The one is to extend it to more datasets and areas such as sentimental analysis (Cambria, Schuller, Xia, & Havasi, 2013) and recommendation system (Lathia, Hailes, Capra, & Amatriain, 2010). The other is to adopt the MapReduce framework to design a parallel distributed computation method to speed the gradient descent of the proposed DRI-RCNN approach to make it convergent more rapidly in network training.

Acknowledgment

This research is supported in part by National Natural Science Foundation of China under Grant Nos. 61379046, 71601023, 91318302 and 61432001; the Innovation Fund Project of Xi'an Science and Technology Program(Special Series for Xi'an University No. 2016CXWL21).

Supplementary materials

Supplementary material associated with this article can be found, in the online version, at doi:10.1016/j.ipm.2018.03.007.

References

Bottou, L. (1998). Online algorithms and stochastic approximations, online learning and neural networks. In Saad David (Ed.). Cambridge, UK,: Cambridge University Press.

Buettner, R. (2016). *Predicting user behavior in electronic markets based on personality-mining in large online social networks: A personality-based product recommender framework.* Electronic Markets, Springer1–19.

Cagnina, L., & Rosso, P. (2015). Classification of deceptive opinions using a low dimensionality representation. *Proceedings of the 6th workshop on computational approaches to subjectivity, sentiment & social media analysis* (pp. 58–66).

Cagnina, L., & Rosso, P. (2017). Detecting deceptive opinions: Intra and cross-domain classification using an efficient representation. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems, 25*(Supplement 2), 175–189.

Cambria, E., Schuller, B., Xia, Y., & Havasi, C. (2013). New avenues in opinion mining and sentiment analysis. *IEEE Intelligent Systems, 28*(2), 15–21.

Chatterjeei, P. (2001). Online reviews. Do consumers use them. *Proceedings of conference on association for consumer research* (pp. 129–134). 2001.

Chen, C., Wu, K., Srinivasan, V., & Zhang, X. (2013). Battling the Internet water army: Detection of hidden paid posters. *Proceedings of 2013 IEEE/ACM international conference on advances in social networks analysis and mining (ASONAM)*.

Ciresan, D. C., Meier, U., Masci, J., Gambardella, L. M., & Schmidhuber, J. (2011). Flexible, high performance convolutional neural networks for image classification. *Proceedings of the twenty-second international joint conference on artificial intelligence* (pp. 1237–1242).

Collins, M.. Available online http://www.cs.columbia.edu/~mcollins/courses/nlp2011/notes/pcfgs.pdf accessed on 26 February 2016 .

Collobert, R., & Weston, J. (2008). A unified architecture for natural language processing: Deep neural networks with multitask learning. *Proceedings of the 25th international conference on machine learning* (pp. 160–167).

Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., & Kuksa, P. (2011). Natural language processing (almost) from scratch. *The Journal of Machine Learning Research, 12*, 2493–2537.

Dean, J., & Ghemawat, S. (2004). MapReduce: Simplified data processing on large clusters. *Proceedings of the 6th conference on Symposium on Operating Systems Design & Implementation* (pp. 10–20).

Ding, C., Li, T., & Peng, W. (2007). Nonnegative matrix factorization and probabilistic latent semantic indexing: Equivalence chi-square statistic, and a hybrid method. *Proceedings of AAAI national conference on artificial intelligence (AAAI-06)* (pp. 342–347).

Feng, S., Banerjee, R., & Choi, Y. (2012). Syntactic stylometry for deception detection. *Proceedings of the 50th annual meeting of the association for computational linguistics* (pp. 171–175).
Feng, V. W., & Hirst, G. (2013). Detecting deceptive opinions with profile compatibility. *International joint conference on natural language processing* (pp. 338–346).
Fürnkranz, J., & Hüllermeier, E. (2003). Pairwise preference learning and ranking. *Proceedings of 2013 European conference on machine learning (ECML)* (pp. 145–156).
Gokhman, S., Hancock, J., Prabhu, P., Ott, M., & Cardie, C. (2012). In search of a gold standard in studies of deception. *Proceedings of the EACL 2012 workshop on computational approaches to deception detection* (pp. 23–30).
Hernández, D., Montes-y-Gómez M, M., Rosso, P., & Guzmán R., R. (2015). Detecting positive and negative deceptive opinions using PU-learning. *Information Processing & Management, 51*(4), 433–443.
Hofmann, T. (1999). Probabilistic latent semantic analysis. *In Proceedings of 22nd international ACM SIGIR conference on research and development in information retrieval* (pp. 50–57).
Hung, C. (2017). Word of mouth quality classification based on contextual sentiment lexicons. *Information Processing and Management, 53*(4), 751–763.
Jeffery, L. E. (1990). Finding structure in time. *Cognitive Science, 14*, 179–211.
Jindal, N., & Liu, B. (2008). Opinion spam and analysis. *Proceedings of the 2008 international conference on web search and data mining* (pp. 219–230).
Kalchbrenner, N., & Blunsom, P. (2013). Recurrent convolutional neural networks for discourse compositionality. *Proceedings of the workshop on continuous vector space models and their compositionality* (pp. 119–126).
Lai, S., Xu, L., Liu, K., & Zhao, J. (2015). Recurrent convolutional neural network for text classification. *Proceedings of the twenty-ninth AAAI conference on artificial intelligence* (pp. 2267–2273).
Lathia, N., Hailes, S., Capra, L., & Amatriain, X. (2010). Temporal diversity in recommender systems. *Proceedings of the 33rd international ACMSIGIR conference on research and development in information retrieval* (pp. 210–217).
Li, J., Ott, M., Cardie, C., & Hovy, E. (2014). Towards a general rule for identifying deceptive opinion spam. *Proceedings of the 52nd annual meeting of the association for computational linguistics* (pp. 1566–1576).
Lim, Y. J., Osman, A., Salahuddin, S. N., Romle, A. R., & Abdullah, S. (2016). Factors influencing online shopping behavior: The mediating role of purchase intention. *Procedia economics and finance. 35*. *Procedia economics and finance* (pp. 401–410).
Liu, B. (2015). *Sentiment analysis: Mining opinions, sentiments, and emotions.* Cambridge University Press June.
Ma, B., Zhang, N., Liu, G., Li, L., & Yuan, H. (2016). Semantic search for public opinions on urban affairs: A probabilistic topic modeling-based approach. *Information Processing & Management, 52*(3), 430–445.
Marcus, M. P., Marcinkiewicz, M. A., & Santorini, B. (1993). Building a large annotated corpus of english: The Penn Treebank. *Comput Ling, 19*(2), 313–330.
Mikolov, T., Sutskever, I., Chen, K., Corrado, G., & Dean, J. (2013a). Distributed representations of words and phrases and their compositionality. *Proceedings of the 26th international conference on neural information processing systems* (pp. 3111–3119).
Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013b). Efficient estimation of word representations in vector space. *Proceedings of international conference on learning representations* (pp. 5–16).
Mudambi, S. M., & Schuff, D. (2010). What makes a helpful online review? A study of customer review on Amazon.com. *MIS Quarterly, 34*(1), 185–200.
Nair, V., & Hinton, G. E. (2010). Rectified linear units improve restricted boltzmann machines. *Proceedings of the 27th international conference on machine learning* (pp. 807–814).
Ng, A. Y. (2004). Feature selection, *L1* vs. *L2* regularization, and rotational invariance. *Proceedings of the twenty-first international conference on machine learning* (pp. 78–85).
Nitin, I., Fred, J. D., & Zhang, T. (2005). *Text Mining: predictive methods for analyzing unstructured information.* Springer Science and Business Media, Inc15–37.
Olmos, R., Jorge-Botana, G., Luzón, J. M., Martín-Cordero, J. I., & León., J. A. (2016). Transforming LSA space dimensions into a rubric for an automatic assessment and feedback system. *Information Processing & Management, 52*(3), 359–373.
Ott, M., Choi, Y., Cardie, C., & Hancock, J. T. (2011). Finding deceptive opinion spam by any stretch of the imagination. *Proceedings of the 49th annual meeting of the association for computational linguistics* (pp. 309–319).
Pennebaker, J. W., Chung, C. K., Ireland, M., Gonzales, A., & Booth, R. J. (2007). *The development and psychometric properties of LIWC2007* LIWC.Net.
Ren, Y., & Ji, D. (2017). Neural networks for deceptive opinion spam detection: An empirical study. *Information Sciences, 385*, 213–224.
Rosso, P., & Cagnina, L. (2017). Deception detection and opinion spam. In E. Cambria, D. Das, S. Bandyopadhyay, & S. Feraco (Eds.). *A practical guide to sentiment analysis, socio-affective computing, 5* (pp. 155–171). Springer-Verlag 2017.
Salton, G., Wong, A., & Yang, C. S. (1975). A vector space model for automatic indexing. *Communications of the ACM, 18*(11), 613–620.
Sebastiani, F. (2002). Machine learning in automated text categorization. *ACM Computing Surveys, 34*(1), 1–47.
Socher, R., Huang, E. H., Pennington, J., Ng, A. Y., & Manning, C. D. (2011). Dynamic pooling and unfolding recursive autoencoders for paraphrase detection. *Proceedings of advances in neural information processing systems 24 (NIPS 2011)* (pp. 801–809).
Srijith, P. K., Hepple, M., Bontcheva, K., & Preotiuc-Pietro, D. (2017). Sub-story detection in Twitter with hierarchical Dirichlet processes. *Information Processing & Management, 53*(4), 989–1003.
Toral, A., Pecina, P., Wang, L., & van Genabith, J. (2015). Linguistically-augmented perplexity-based data selection for language models. *Journal of Computer Speech and Language, 32*(1), 11–26.
Valant, J. (2015). Online consumer reviews: The case of misleading or fake reviews. *European Parliamentary Research Service* online: http://www.eesc.europa.eu/resources/docs/online-consumer-reviews—the-case-of-misleading-or-fake-reviews.pdf.
Van Rijsbergen, C. J. (1997). A theoritical basis for the use of co-occurrence data in information retrieval. *Journal of Documentation, 33*(2), 106–119.
Waibel, A., Hanazawa, T., Hinton, G., Shikano, K., & Lang, K. J. (1989). Phoneme recognition using time-delay neural networks. *IEEE Transactions on Acoustics, Speech, and Signal Processing, 37*(3), 328–338.
Xavier, G., & Yoshua, B. (2010). Understanding the difficulty of training deep feedforward neural networks. *Proceedings of the 13th international conference on artificial intelligence and statistics (AISTATS)* (pp. 249–256).
Zhang, W., Bu, C., Yoshida, T., & Zhang, S. (2016). CoSpa: A co-training approach for spam review identification with support vector machine. *Information, 7*(1), 12.