Chapter 8
Deep Learning in Sentiment Analysis

Duyu Tang and Meishan Zhang

Abstract Sentiment analysis (also known as opinion mining) is an active research area in natural language processing. The task aims at identifying, extracting, and organizing sentiments from user-generated texts in social networks, blogs, or product reviews. Over the past two decades, many studies in the literature exploit machine learning approaches to solve sentiment analysis tasks from different perspectives. Since the performance of a machine learner heavily depends on the choices of data representation, many studies devote to building powerful feature extractor with domain expertise and careful engineering. Recently, deep learning approaches emerge as powerful computational models that discover intricate semantic representations of texts automatically from data without feature engineering. These approaches have improved the state of the art in many sentiment analysis tasks, including sentiment classification, opinion extraction, fine-grained sentiment analysis, etc. In this paper, we give an overview of the successful deep learning approaches sentiment analysis tasks at different levels.

8.1 Introduction

Sentiment analysis (also known as opinion mining) is a field that automatically analyzes people's opinions, sentiments, emotions from user-generated texts (Pang et al. 2008; Liu 2012). Sentiment analysis is a very active research area in natural language processing (Manning et al. 1999; Jurafsky 2000), and is also widely studied in data mining, web mining,and social media analytics as sentiments are key influencers of

D. Tang
Microsoft Research Asia, Beijing, China
e-mail: dutang@microsoft.com

M. Zhang (✉)
Heilongjiang University, Harbin, Heilongjiang, China
e-mail: mszhang@hlju.edu.cn

L. Deng and Y. Liu (eds.), *Deep Learning in Natural Language Processing*, https://doi.org/10.1007/978-981-10-5209-5_8

Table 8.1 An example that illustrates the definition of sentiment

Target	Sentiment	Holder	Time
iPhone	Positive	Alice	June 4, 2015
Touch screen	Positive	Alice	June 4, 2015
Price	Negative	Alice	June 4, 2015

human behaviors. With the rapid growth of social media such as Twitter,[1] Facebook[2], and review sites such as IMDB,[3] Amazon,[4] Yelp,[5] sentiment analysis draws growing attention from both the research and industry communities (Table 8.1).

According to the definition from (Liu 2012), sentiment (or an opinion) is represented as a quintuple e, a, s, h, t, where e is the name of an entity, a is the aspect of e, s is the sentiment on aspect a of entity e, h is the opinion holder, and t is the time when the opinion is expressed by h. In this definition, a sentiment s can be a positive, negative, or neutral sentiment, or a numeric rating score expressing the strength/intensity of the sentiment (e.g., 1–5 stars) in review sites like Yelp and Amazon. The entity can be a product, service, topic organization, or event (Hu and Liu 2004; Deng and Wiebe 2015).

Let us use an example to explain the definition of "sentiment". Supposing a user named Alice posted a review "I bought an iPhone a few days ago. It is such a nice phone. The touch screen is really cool. However, the price is a little high." at June 4, 2015. Three sentiment quintuples are involved in this example, as shown in Table 8.1.

Based on the definition of "sentiment", sentiment analysis aims at discovering all the sentiment quintuples in a document. Sentiment analysis tasks are derived from the five components of the sentiment quintuple. For example, document/sentence-level sentiment classification (Pang et al. 2002; Turney 2002) targets at the third component (sentiment such as positive, negative, and neutral) while ignoring the other aspects. Fine-grained opinion extraction focuses on the first four components of the quintuple. Target-dependent sentiment classification focuses on the second and the third aspects.

Over the past two decades, machine learning-driven methods have dominated most sentiment analysis tasks. Since feature representation greatly affects the performance of a machine learner (LeCun et al. 2015; Goodfellow et al. 2016), a lot of studies in the literature focus on effective features in hand with domain expertise and careful engineering. But this can be avoided by representation learning algorithms, which automatically discover discriminative and explanatory text representations from data. Deep learning is a kind of representation learning approach, which learns multiple levels of representation with nonlinear neural networks, each of which transforms

[1] https://twitter.com/.

[2] https://www.facebook.com.

[3] http://www.imdb.com/.

[4] https://www.amazon.com/.

[5] https://www.yelp.com/.

the representation at one level into a representation at a higher and more abstract level. The learned representations can be naturally used as features and applied to detection or classification tasks. In this chapter, we introduce successful deep learning algorithms for sentiment analysis. The notation of "deep learning" in this chapter stands for the use of neural network approaches to learning continuous and real-valued text representation/feature automatically from data.

We organize this chapter as follows. Since word is the basic computational unit of natural language, we first describe the methods to learn continuous word representation, also called word embedding. These word embeddings can be used as inputs to subsequent sentiment analysis tasks. We describe semantic compositional methods that compute representations of longer expressions (e.g., sentence or document) for sentence/document-level sentiment classification task (Socher et al. 2013; Li et al. 2015; Kalchbrenner et al. 2014), followed by neural sequential models for fine-grained opinion extraction. We finally conclude this paper and provide some future directions.

8.2 Sentiment-Specific Word Embedding

Word representation aims at representing aspects of word meaning. For example, the representation of "cellphone" may capture the facts that cellphones are electronic products, that they include battery and screen, that they can be used to chat with others, and so on. A straightforward way is to encode a word as a one-hot vector. It has the same length as the size of the vocabulary, and only one dimension is 1, with all others being 0. However, the one-hot word representation only encodes the indices of words in a vocabulary, while failing to capture rich relational structure of the lexicon.

One common approach to discover the similarities between words is to learn word clusters (Brown et al. 1992; Baker and McCallum 1998). Each word is associated with a discrete class, and words in the same class are similar in some respect. This leads to a one-hot representation over a smaller vocabulary size. Instead of characterizing the similarity with a discrete variable based on clustering results which correspond to a soft or hard partition of the set of words, many researchers target at learning a continuous and real-valued vector for each word, also known as word embedding. Existing embedding learning algorithms are typically based on the distributional hypothesis (Harris 1954), which states that words in similar contexts have similar meanings. Towards this goal, many matrix factorization methods can be viewed as modeling word representations. For example, Latent Semantic Indexing (LSI) (Deerwester et al. 1990) can be regarded as learning a linear embedding with a reconstruction objective, which uses a matrix of "term–document" co-occurrence statistics, e.g., each row stands for a word or term and each column corresponds to an individual document in the corpus. Hyperspace Analogue to Language (Lund and Burgess 1996) utilizes a matrix of term–term co-occurrence statistics, where both rows and columns correspond to words and the entries stand for the number of times a given word occurs in the context of another word. Hellinger PCA (Lebret et al. 2013) is also investigated to learn word embeddings over "term–term" co-occurrence statistics. Since standard matrix factorization methods do not incorporate task-specific information, it is not clear whether they are useful enough for a target goal. Supervised Semantic Indexing (Bai et al. 2010) tackles this problem and takes the supervised information of a specific task (e.g. information retrieval) into consideration. They learn the embedding model from click-through data with a margin ranking loss. DSSM (Huang et al. 2013; Shen et al. 2014) also could be considered as learning task-specific text embeddings with weak supervision in information retrieval.

A pioneering work that explores neural network approaches is given by (Bengio et al. 2003), which introduces a neural probabilistic language model that learns simultaneously a continuous representation for words and a probability function for word sequences based on these word representations. Given a word and its preceding context words, the algorithm first maps all these words to continuous vectors with a shared lookup table. Afterward, word vectors are fed to a feed-forward neural network with softmax as output layer to predict the conditional probability of next word. The parameters of neural network and lookup table are jointly estimated with backpropagation. Following Bengio et al. (2003)'s work, several approaches are proposed to speed-up training processing or capturing richer semantic information. Bengio et al. (2003) introduce a neural architecture by concatenating the vectors of context words and current word, and use importance sampling to effectively optimize the model with observed "positive sample" and sampled "negative samples". Morin and Bengio (2005) develop hierarchical softmax to decompose the conditional probability with a hierarchical binary tree. Mnih and Hinton (2007) introduce a log-bilinear language model. Collobert and Weston (2008) train word embeddings with a ranking-type hinge loss function by replacing the middle word within a window with a randomly selected one. Mikolov et al. (2013a, b) introduce continuous bag-of-words (CBOW) and continuous skip-gram, and release the popular word2vec[6] toolkit. The CBOW model predicts the current word based on the embeddings of its context words, and the skip-gram model predicts surrounding words given the embedding of current word. Mnih and Kavukcuoglu (2013) accelerate the word embedding learning procedure with Noise Contrastive Estimation (Gutmann and Hyvärinen 2012). There are also many algorithms developed for capturing richer semantic information, including global document information (Huang et al. 2012), word morphemes (Qiu et al. 2014), dependency-based contexts (Levy and Goldberg 2014), word–word co-occurrence (Levy and Goldberg 2014), sense of ambiguous words (Li and Jurafsky 2015), semantic lexical information in WordNet (Faruqui et al. 2014), hierarchical relations between words (Yogatama et al. 2015).

The aforementioned neural network algorithms typically only use the contexts of words to learn word embeddings. As a result, the words with similar contexts but opposite sentiment polarity like "good" and "bad" are mapped into close vectors in the embedding space. This is meaningful for some tasks such as POS tagging as the two words have similar usages and grammatical roles, but this is problematic

[6] https://code.google.com/p/word2vec/.

for sentiment analysis as “good” and “bad” have the opposite sentiment polarity. In order to learn word embeddings tailored for sentiment analysis tasks, some studies encode sentiment of texts in continuous word representation. Maas et al. (2011) introduce a probabilistic topic model by inferring the polarity of a sentence based on the embedding of each word it contains. Labutov and Lipson (2013) re-embed an existing word embedding with logistic regression by regarding sentiment supervision of sentences as a regularization item. Tang et al. (2014) extend the C&W model and develop three neural networks to learn sentiment-specific word embedding from tweets. Tang et al. (2014) use the tweets that contain positive and negative emoticons as training data. The positive and negative emoticon signals are regarded as weak sentiment supervision.

We describe two sentiment-specific approaches that incorporate sentiment of sentences to learn word embeddings. The model of Tang et al. (2016c) extends the context-based model of Collobert and Weston (2008), and the model of Tang et al. (2016a) extends the context based model of Mikolov et al. (2013b). We describe the relationships between these models.

The basic idea of the context-based model (Collobert and Weston 2008) is to assign a real word-context pair (w_i, h_i) a higher score than an artificial noise (w^n, h_i) by a margin. The model is learned to minimize the following hinge loss function, where T is the training corpora:

$$loss = \sum_{(w_i,h_i)\in T} max(0, 1 - f_\theta(w_i, h_i) + f_\theta(w^n, h_i)). \tag{8.1}$$

The scoring function $f_\theta(w, h)$ is achieved with a feed forward neural network. Its input is the concatenation of the current word w_i and context words h_i, and the output is a linear layer with only one node which stands for the compatibility between w and h. During training, an artificial noise w^n is randomly selected from the vocabulary.

The basic idea of sentiment-specific approach of Tang et al. (2014) is that if the gold sentiment polarity of a word sequence is positive, the predicted positive score should be higher than the negative score. Similarly, if the gold sentiment polarity of a word sequence is negative, its positive score should be smaller than the negative score. For example, if a word sequence is associated with two scores $[f_{pos}^{rank}, f_{neg}^{rank}]$, then the values of [0.7, 0.1] can be interpreted as a positive case because the positive score 0.7 is greater than the negative score 0.1. By that analogy, the result with [−0.2, 0.6] indicates a negative polarity. The neural network-based ranking model is given in Fig. 8.1b, which shares some similarities with (Collobert and Weston 2008). As is shown, the ranking model is a feed-forward neural network consisting of four layers ($lookup \rightarrow linear \rightarrow hTanh \rightarrow linear$). Let us denote the output vector of ranking model as f^{rank}, where $C = 2$ for binary positive and negative classification. The margin ranking loss function for model training is described as below.

$$loss = \sum_{t}^{T} max(0, 1 - \delta_s(t) f_0^{rank}(t) + \delta_s(t) f_1^{rank}(t)) \tag{8.2}$$

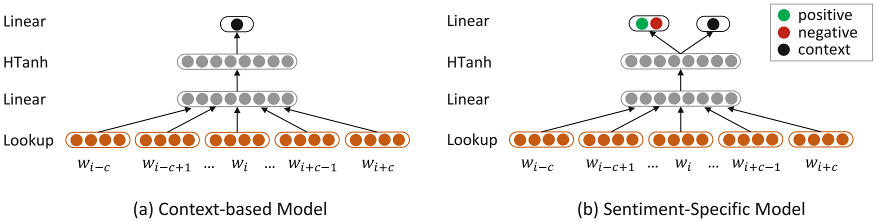


Fig. 8.1 An extension on ranking-based model for learning sentiment-specific word embeddings

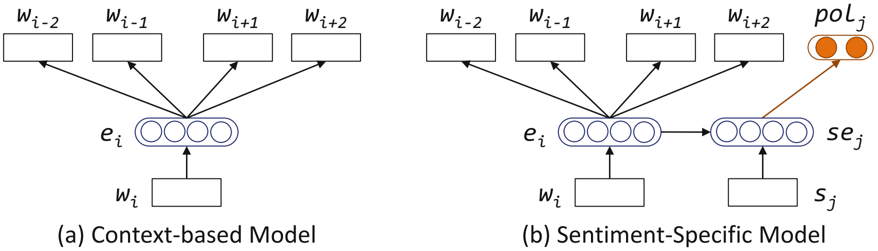


Fig. 8.2 An extension on skip-gram for learning sentiment-specific word embeddings

where T is the training corpus, f_0^{rank} is the predicted positive score, f_1^{rank} is the predicted negative score, $\delta_s(t)$ is an indicator function which reflects the gold sentiment polarity (positive or negative) of a sentence.

$$\delta_s(t) = \begin{cases} 1 & \text{if } f^g(t) = [1, 0] \\ -1 & \text{if } f^g(t) = [0, 1] \end{cases} \tag{8.3}$$

Holding a similar idea, an extension of skip-gram (Mikolov et al. 2013b) is developed to learn sentiment-specific word embeddings. Given a word w_i, skip-gram maps it into its continuous representation e_i, and utilizes e_i to predict the context words of w_i, namely w_{i-2}, w_{i-1}, w_{i+1}, w_{i+2}, et al. The objective of skip-gram is to maximize the average log probability:

$$f_{SG} = \frac{1}{T} \sum_{i=1}^{T} \sum_{-c \le j \le c, j \ne 0} log\ p(w_{i+j}|e_i), \tag{8.4}$$

where T is the occurrence of each phrase in the corpus, c is the window size, e_i is the embedding of the current phrase w_i, w_{i+j} is the context words of w_i, $p(w_{i+j}|e_i)$ is calculated with hierarchical softmax. The basic *softmax* unit is calculated as $softmax_i = exp(z_i)/\sum_k exp(z_k)$.

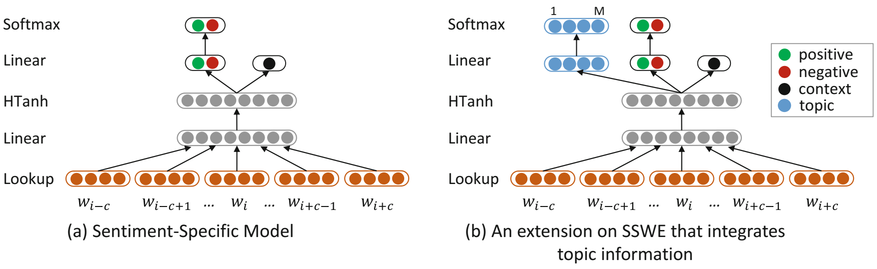


Fig. 8.3 Different ways to learn sentiment-specific word embeddings (**a**), and to incorporate topic information of texts (**b**)

The sentiment-specific model is given in Fig. 8.2b. Given a triple $\langle w_i, s_j, pol_j\rangle$ as input, where w_i is a phrase contained in the sentence s_j whose gold sentiment polarity is pol_j, the training objective is to not only utilize the embedding of w_i to predict its context words, but also to use the sentence representation se_j to predict the gold sentiment polarity of s_j, namely pol_j. The sentence vector is calculated by averaging the embeddings of words contained in a sentence. The objective is to maximize the weighted average loss function as given below.

$$f = \alpha \cdot \frac{1}{T}\sum_{i=1}^{T}\sum_{-c\leq j\leq c, j\neq 0} log\ p(w_{i+j}|e_i) + (1-\alpha)\cdot\frac{1}{S}\sum_{j=1}^{S} log\ p(pol_j|se_j), \tag{8.5}$$

where S is the occurrence of each sentence in the corpus, α weights the context, and the sentiment parts, $\sum_k pol_{jk} = 1$. For binary classification between positive and negative, the distribution of [0, 1] is for positive and [0, 1] is for negative.

There are different ways to guide the embedding learning process with sentiment information of texts. For example, the model of Tang et al. (2014) extends the ranking model of Collobert and Weston (2008) and use the hidden vector of text span to predict the sentiment label. Ren et al. (2016b) extend SSWE and further predicts the topic distribution of text based on input n-grams. These two approaches are given in Fig. 8.3.

8.3 Sentence-Level Sentiment Classification

Sentence-level sentiment analysis focuses on classifying the sentiment polarities of a given sentence. Typically, for one sentence $w_1 w_2 \ldots w_n$, we divide its polarities into two (±) or three (±/0) categories, where + denotes positive, - denotes negative, and 0 denotes neutral. The task is a representative sentence classification problem.

Under the neural network setting, sentence-level sentiment analysis can be modeled as a two-phase framework, one being a sentence representation module by using

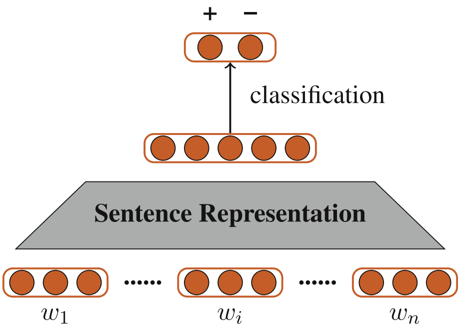


Fig. 8.4 Framework of sentiment classification

sophisticated neural structures, and the other being a simple classification module which can be resolved by a softmax operation. Figure 8.4 shows the overall framework.

Basically, with word embeddings for each sentential word, one can use pooling strategies to obtain a simple representation for a sentence, A pooling operation is able to summary salient features from a sequential input with variable length. Formally, we can use the equation $\mathbf{h} = \sum_{i=1}^{n} a_i \mathbf{x}_i$ to define popular pooling functions. For example, the widely adopted average (avg), max, and min pooling operations can be formalized as follows:

$$a_i^{avg} = \frac{1}{n}, \quad a_{ij}^{min} = \begin{cases} 1, & \text{if } i = \text{argmin}_k \mathbf{x}_{kj} \\ \\ 0, & \text{otherwise,} \end{cases}, \quad a_{ij}^{max} = \begin{cases} 1, & \text{if } i = \text{argmax}_k \mathbf{x}_{kj} \\ \\ 0, & \text{otherwise.} \end{cases} \tag{8.6}$$

Tang et al. (2014) exploit the three pooling methods to verify their proposed sentiment-encoded word embeddings, The method is just one simple example to represent sentences. In fact, recent advances on sentence representation for sentence classification are far beyond it. A number of sophisticated neural network structures have been proposed in the literature. As a whole, we summarize the related work by four categories: (1) convolutional neural networks, (2) recurrent neural networks, (3) recursive neural networks, (4) enhanced sentence representation by auxiliary resources. We introduce these works in the following subsections, respectively.

8.3.1 Convolutional Neural Networks

In the pooling neural network, we are only able to use word-level features. When the order of words changes in a sentence, the sentence representation result remains unchanged. In traditional statistical models, n-gram word features are adopted in order to alleviate the issue, showing improved performances. For neural network models, a convolution layer can be exploited to achieve a similar effect.

Formally, a convolution layer performs nonlinear transformations by traversing a sequential input with a fixed-size local filter. Give an input sequence $\mathbf{x}_1\mathbf{x}_2\ldots\mathbf{x}_n$, assuming that the size of local filter is K, then we can obtain a sequential output of $\mathbf{h}_1\mathbf{h}_2\ldots\mathbf{h}_{n-K+1}$:

$$\mathbf{h}_i = f\left(\sum_{k=1}^{K} W_k\mathbf{x}_{i+K-k}\right),$$

where f is an activation function such as $\tanh(\cdot)$ and $\mathrm{sigmoid}(\cdot)$. When $K = 3$ and $\mathbf{x}_i$ is the input word embedding, the resulting $\mathbf{h}_i$ is a nonlinear combination of $\mathbf{x}_i$, $\mathbf{x}_{i+1}$, and $\mathbf{x}_{i+2}$, similar to the mixed unigram, bigram, and trigram features, which concatenate the surface forms of the corresponding words in a hard way.

Typically, convolutional neural network (CNN) is a certain network that integrates a convolution layer and a pooling layer together, as shown in Fig. 8.5, which has been widely studied for sentence-level sentiment classification. An initial attempt by directly applying of a standard CNN is introduced by Collobert et al. (2011). The study obtains the final sentence representation by using a convolutional layer over a sequence of input word embeddings, and using a further max pooling over the resulting hidden vectors.

Kalchbrenner et al. (2014) extend the basic CNN model for better sentence representation by two aspects. On the one hand, they use dynamic k-max pooling, where top-k values are reserved during pooling instead of only one value for each dimension in the simple max pooling. The value k is defined according to sentence length dynamically. On the other hand, they enlarge the layer number of CNN, using multilayer CNN structures, motivated by the intuition that deeper neural networks can encode more sophisticated features. Figure 8.6 shows the framework of multilayer CNNs.

Several CNN variations have been studied to better represent sentences. One most representative work is the nonlinear, nonconsecutive convolution operator proposed by Lei et al. (2015), as shown in Fig. 8.7. The operator aims to extract all n-word combinations through tensor algebra, no matter whether the words are consecutive. The process is conducted recursively, first one word, then two-word and further

Fig. 8.5 Framework of CNN

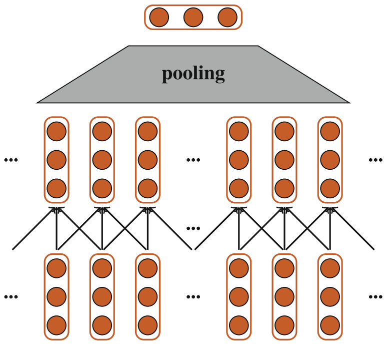


Fig. 8.6 Multilayer CNNs

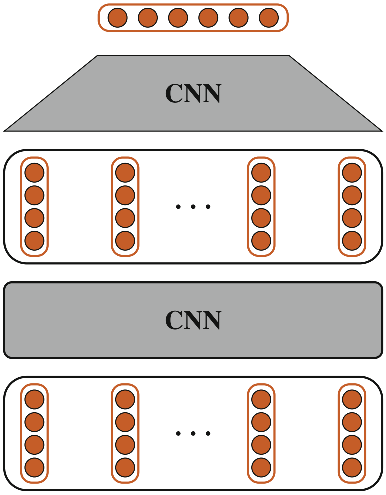
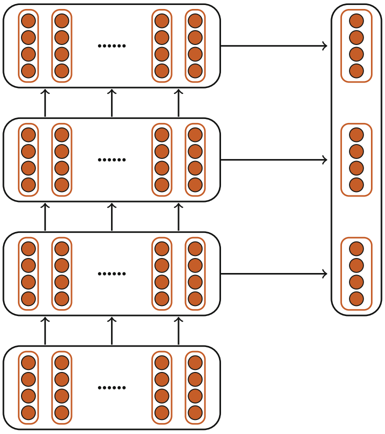


Fig. 8.7 Nonlinear, nonconsecutive convolution



three-word combinations, respectively. They extract all unigram, bigram, and trigram features by the following formulas:

$$\begin{aligned}
\mathbf{f}_i^1 &= P\mathbf{x}_i \\
\mathbf{f}_i^2 &= s_{i-1}^1 \odot Q\mathbf{x}_i \text{ where } \mathbf{s}_i^1 = \lambda \mathbf{s}_{i-1}^1 + \mathbf{f}_i^1 \\
\mathbf{f}_i^3 &= s_{i-1}^2 \odot R\mathbf{x}_i \text{ where } \mathbf{s}_i^2 = \lambda \mathbf{s}_{i-1}^2 + \mathbf{f}_i^2,
\end{aligned}$$

where P, Q, and R are model parameters, λ is a hyper-parameter, and $\odot$ denote element-wise product. Finally, they make compositions of the three kinds of features, forming the representation of a sentence.

A number of studies have focused their attention on the exploration of heterogeneous input word embeddings. For example, Kim (2014) studies three different

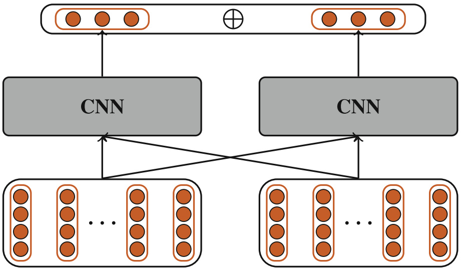


Fig. 8.8 Multichannel CNNs

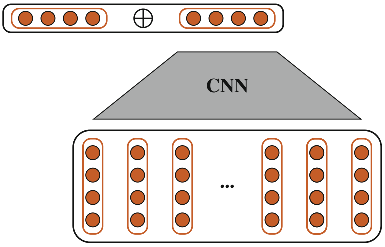


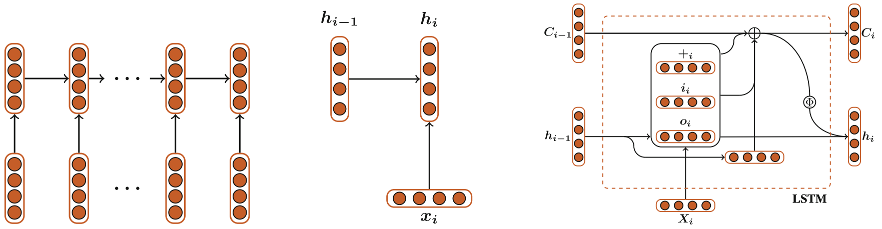
Fig. 8.9 Enhanced word representations with character features

methods of using word embedding. The author concerns two different embeddings, a randomly initialized embedding and a pretrained embedding, considering the effect of dynamic fine-tuning over these embeddings. Finally, it combines the two kinds of embeddings and proposes the multichannel CNNs based on heterogeneous word embeddings, as shown in Fig. 8.8. The work is extended by Yin and Schütze (2015), who use several different word embeddings by multichannel multilayer CNNs. And, in addition, they exploit extensive pretraining techniques for the model weight initialization. However, a simpler version of it is presented by Zhang et al. (2016d), which meanwhile shows better performances.

Another extension of word embeddings is to enhance word representation by character-level features. The neural network to build word representations based on input character sequences is in spirit similar to that of sentence representations from input word sequences. Thus, we can also apply a standard CNN structure over the character embedding sequences to derive word representations. dos Santos and Gatti (2014) study the effect of such an extension. The resulting character-level word representations are concatenated with the original word embeddings, shown in Fig. 8.9, thus can enhance the final word representations for sentence encoding.

8.3.2 Recurrent Neural Networks

The CNN structure uses a fixed-size of word window to capture the local composition features around a given position, achieving promising results. However, it ignores the long-distance dependency features that reflect syntactic and semantic information, which are particularly important in understanding natural language sentences. These



(a) Sentence representation (b) One cell of simple RNN (c) One cell of LSTM-RNN

Fig. 8.10 Sentence representation by using RNN

dependency-based features are addressed by recurrent neural network (RNN) under the neural setting, achieving great success. Formally, a standard RNN computes the output hidden vectors sequentially by $\mathbf{h}_i = f(W\mathbf{x}_i + U\mathbf{h}_{i-1} + \mathbf{b})$, where $\mathbf{x}_i$ denotes the input vector. According to the equation, we can see that the current output $\mathbf{h}_i$ relies not only on the current input $\mathbf{x}_i$, but also on the previous hidden output $\mathbf{h}_{i-1}$. In this manner, the current hidden output can have connections with previous input and output vectors without bound.

Wang et al. (2015) propose the first work of using long short-term memory (LSTM) neural networks for tweet sentiment analysis. Figure 8.10 shows the sentence representation method by using RNN, as well as the internal structures of standard and LSTM-RNN. First they apply a standard RNN over an input embedding sequence $\mathbf{x}_1\mathbf{x}_2\ldots\mathbf{x}_n$, and exploit the last hidden output $\mathbf{h}_n$ as the final representation of one sentence. Then the authors suggest a substitution by using LSTM-RNN structure, since standard RNNs may suffer the gradient explosion and diminish problems, while LSTM is much better by using three gates and a memory cell to connect input and output vectors. Formally, LSTM can be computed by

$$\begin{aligned}
\mathbf{i}_i &= \sigma(W_1\mathbf{x}_i + U_1\mathbf{h}_{i-1} + \mathbf{b}_1) \\
\mathbf{f}_i &= \sigma(W_2\mathbf{x}_i + U_2\mathbf{h}_{i-1} + \mathbf{b}_2) \\
\tilde{\mathbf{c}}_i &= \tanh(W_3\mathbf{x}_i + U_3\mathbf{h}_{i-1} + \mathbf{b}_3) \\
\mathbf{c}_i &= \mathbf{f}_i \odot \mathbf{c}_{i-1} + \mathbf{i}_i \odot \tilde{\mathbf{c}}_i \\
\mathbf{o}_i &= \sigma(W_4\mathbf{x}_i + U_4\mathbf{h}_{i-1} + \mathbf{b}_4) \\
\mathbf{h}_i &= \mathbf{o}_i \odot \tanh(\mathbf{c}_i),
\end{aligned}$$

where W, U, $\mathbf{b}$ are model parameters and σ denotes the sigmoid function.

Further, Teng et al. (2016) extend their work by two points. Figure 8.11 shows their framework. First, they exploit bidirectional LSTM instead, rather than a single left-to-right LSTM. The bidirectional can represent a sentence more comprehensively, where the hidden output of each point can have connections with both previous and future words. Second, they model sentence-level sentiment classification as a structural learning problem, predicting polarities for all sentiment words in a sentence and accumulating together as the evidence to determine the sentential polarity. By

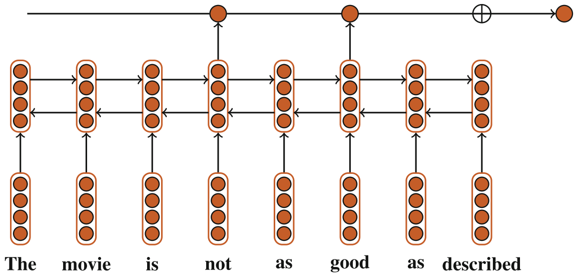


Fig. 8.11 The framework of Teng et al. (2016)

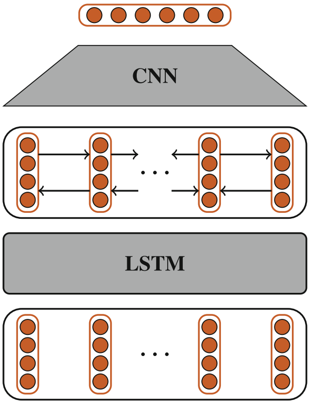


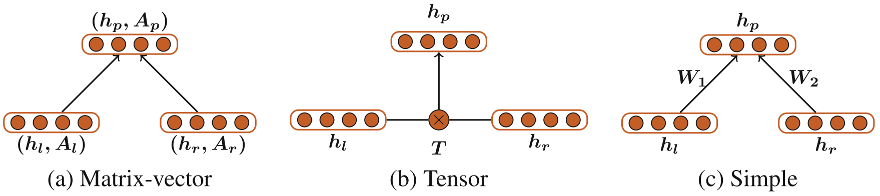
Fig. 8.12 A combination of RNN and CNN

the second extension, their model can effectively integrate the sentiment lexicons, which has been widely used in traditional statistical models.

CNN and RNN model natural language sentences in totally different ways. For example, CNN can better capture local window-based compositions, while RNN is efficient in learning implicit long-distance dependencies. Thus, one natural idea is to combine them together, taking advantages of both neural structures. Zhang et al. (2016c) propose a dependency-sensitive CNN model, which combines a LSTM and a CNN, making a CNN network structure being able to capture long- distance word dependencies as well. Concretely, first they construct a left-to-right LSTM on the input word embeddings, and then a CNN is built on the hidden outputs of the LSTM. Thus the final model can make full use of both local window-based features and global dependency-sensitive features. Figure 8.12 shows the framework of their combination model.

8.3.3 Recursive Neural Networks

Recursive neural network is recently proposed to model tree structural inputs, which are produced by explicit syntactic parsers. Socher et al. (2012) present a recursive matrix-vector neural network to compose two leaf nodes, resulting in



(a) Matrix-vector (b) Tensor (c) Simple

Fig. 8.13 Recursive neural network

the representation of the parent node. By this way, the sentence representation is constructed recursively from bottom to up. They first preprocess the input constituent trees, converting them into a binarized tree, where each parent node has two leaf nodes. Then they apply a recursive neural network over the binary tree by using matrix-vector operations. Formally, they represent each node by a hidden vector $\mathbf{h}$ and a matrix A. As shown in Fig. 8.13a, given the representations of the two child nodes, $(\mathbf{h}_l, A_l)$ and $(\mathbf{h}_r, A_r)$, respectively, the representation of the parent node is computed as follows: (1) $\mathbf{h}_p = f(A_r\mathbf{h}_l, A_l\mathbf{h}_r)$ and (2) $A_p = g(A_l, A_r)$, where $f(\cdot)$ and $g(\cdot)$ are transformation functions with model parameters.

Further, Socher et al. (2013) adopt low-rank tensor operations to substitute the matrix-vector recursion, by using $\mathbf{h}_p = f(\mathbf{h}_l T \mathbf{h}_r)$ to compute the representation of parent nodes, as shown in Fig. 8.13b, where T denotes a tensor. The model achieves better performances due to the tensor composition, which is intuitively simple than matrix-vector operation and has much less number of model parameters. In addition, they define the sentiment polarities over the non-root nodes of syntactic trees, thus can better capture the transition of sentiments from phrases to sentences.

The line of work is extended with three different directions. First, several work tries to find stronger composition operations for tree composition. For example, a number of works simply use $\mathbf{h}_p = f(W_1\mathbf{h}_l, W_2\mathbf{h}_r)$ to compose the leaf nodes, as shown in Fig. 8.13c. The method is much simpler, but suffers from the problem of gradient explosion or diminish, making the parameter learning extremely difficult. Motivated by the work of LSTM-RNN, several studies propose the LSTM adaption for recursive neural network. The representative work includes (Tai et al. 2015) and (Zhu et al. 2015), both of which show the effectiveness of LSTM over tree structures.

Second, sentence representation-based recursive neural network can be strengthened by using multichannel compositions. Dong et al. (2014b) study the effectiveness of such an enhancement. They apply C homogeneous compositions, arriving at C output hidden vectors, which are further used to represent the parent node by using an attention integration. Figure 8.14 shows the framework of their neural network. They apply the method on simple recursive neural networks, achieving consistent better performances on several benchmark datasets.

The third direction is to investigate recursive neural network by using deeper neural network structures, similar to the work of multilayer CNN. Briefly speaking, as the first layer, recursive neural network is applied over the input word embeddings. When all output hidden vectors are ready, the same recursive neural network can be

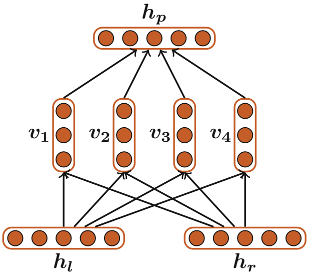


Fig. 8.14 Recursive neural network with multi-compositions

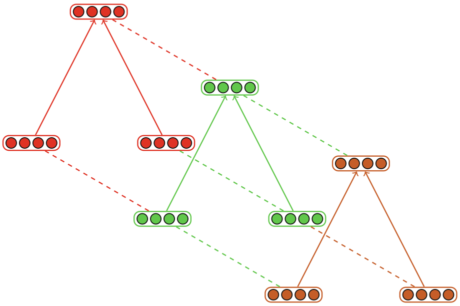


Fig. 8.15 Multilayer recursive neural network

applied by once again. The method is empirically studied by Irsoy and Cardie (2014a). Figure 8.15 shows their framework by using a three-layer recursive neural network. The experimental results demonstrate that deeper recursive neural network can bring better performances than a single-layer recursive neural network.

The above studies all construct recursive neural network over well-formed binary syntactic trees, which is seldom satisfied. Thus, they require certain preprocessing to convert original syntactic structures into binarized ones, which may be problematic without expert supervision. Recently, several studies propose to model trees with unbounded leaf nodes directly. For example, Mou et al. (2015) and Ma et al. (2015) both present a pooling operation based on the child nodes to compose variable length of inputs. Teng and Zhang (2016) perform the pooling process considering the left and right children. In addition, they suggest bidirectional LSTM recursive neural network, considering the top-to-down recursive operation, which is similar with the bidirectional LSTM-RNN.

It is worth to notice that, several works consider sentence representation by using recursive neural network without syntactic tree structures. These work suggest pseudo tree structures based on raw sentence inputs. For example, Zhao et al. (2015) construct a pseudo- directed acyclic graph in order to apply recursive neural network, as shown in Fig. 8.16. In addition, Chen et al. (2015) use a simpler method as shown in Fig. 8.17 to build a tree structure for a sentence automatically. Both the works achieve competitive performances for sentence-level sentiment analysis.

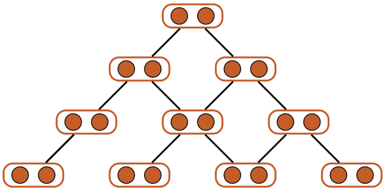


Fig. 8.16 Pseudo-directed acyclic graph of Zhao et al. (2015)

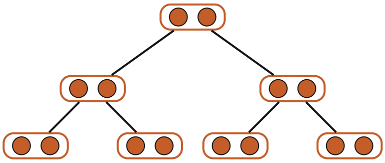


Fig. 8.17 Pseudo binary tree structure of Chen et al. (2015)

8.3.4 Integration of External Resources

The above subsections concern various neural structures for sentence representation, with the information from the source input sentences only, including words, parsing trees. Recently, another line of important work is to enhance sentence representation by integration with external resources. The major resources can be divided into three categories, including the large-scale raw corpus to pretrain supervised model parameters, external human-annotated or automatically extracted sentiment lexicons, and the background knowledge under a certain setting, for example, Twitter sentiment classification.

The exploration of large-scale corpus to enhance sentence representation has been investigated by a number of studies. Among these studies, the sequence autoencoder model proposed by Hill et al. (2016) are most representative. Figure 8.18 shows an example for the model, which first represents sentences by LSTM-RNN encoder, and then tries to generate the original sentential word step by step, thus model parameters are learned by this supervision, which are further used as external information for sentence representation. In particular, Gan et al. (2016) suggest a CNN encoder instead, aiming to solve the low-efficiency problem in LSTM-RNN.

External sentiment lexicons have been largely investigated in the statistical models, while there remains relatively little work under the neural setting, although there has been much work on automatically constructing sentiment lexicons. There are two exceptions. Teng et al. (2016) incorporate context-sensitive lexicon features in a LSTM-RNN neural network, treating sentence-level sentiment scores as a weighted sum of prior sentiment scores of negation words and sentiment words. Qian et al. (2017) go further, investigating the sentiment shifting effect of sentiment, negation, and intensity word, proposing a linguistically regularized LSTM model for sentence-level sentiment analysis.

There are several studies to investigate other information for sentence-level sentiment analysis under certain settings. In the Twitter sentiment classification, we can use several contextual information, including the tweet author's history tweets,

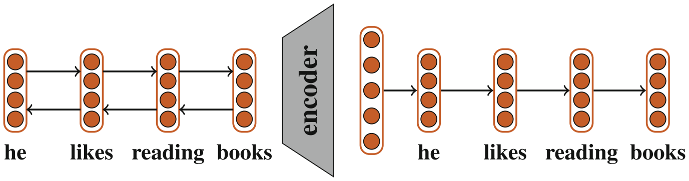


Fig. 8.18 Autoencoder by LSTM-RNN

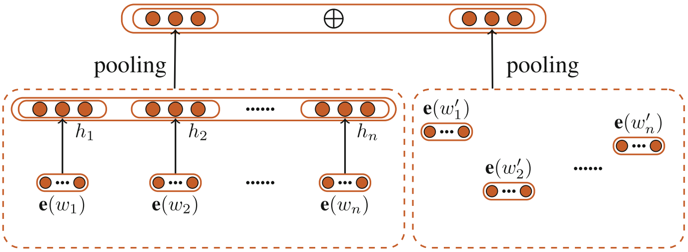


Fig. 8.19 Sentiment classification with contextual features

the conversational tweets surrounding the tweet, and the topic-related tweets. These information can be all severed as background information, which is intuitively helpful to decide the sentiment of a tweet. Ren et al. (2016a) exploit these related information in a neural network model by an additional contextual part, as shown in Fig. 8.19, to enhance sentiment analysis in Twitter. For the source input sentences, they apply a CNN to represent it, while for the contextual part, they apply a simple pooling neural network over a set of salient contextual words. Recently, Mishra et al. (2017) suggest an integration of cognitive features from gaze data to enhance sentence-level sentiment analysis, which is achieved by using an additional CNN structure to model the gaze features.

8.4 Document-Level Sentiment Classification

Document-level sentiment classification aims at identifying the sentiment label of a document (Pang et al. 2002; Turney 2002). The sentiment labels could be two categories such as *thumbs up* and *thumbs down* (Pang et al. 2002) or multiple categories such as the 1–5 stars on review sites (Pang and Lee 2005).[7]

In the literature, existing sentiment classification approaches could be grouped into two directions: lexicon- based approach and corpus-based approach. Lexicon-based approaches (Turney 2002; Taboada et al. 2011) mostly use a dictionary of

[7] In practice, it is time consuming to obtain the document- level sentiment labels via human annotation. Researchers typically leverage the review documents from IMDB, Amazon, and Yelp, and regard the associated rating stars as the sentiment labels.

sentiment words with their associated sentiment polarity, and incorporate negation and intensification to compute the sentiment polarity for each document. A representative lexicon-based method is given by (Turney 2002), which consists of three steps. Phrases are first extracted, if their POS tags conform to the predefined patterns. Afterward, the sentiment polarity of each extracted phrase is estimated through pointwise mutual information (PMI), which measures the degree of statistical dependence between two terms. In Turney's work, the PMI score is calculated by feeding queries to a search engine and collecting the number of hits. Finally, he averages the polarity of all phrases in a review as its sentiment polarity. Ding et al. (2008) apply negation words like "not", "never", "cannot", and contrary words like "but" to enhance the performance of lexicon-based method. Taboada et al. (2011) integrate intensifications and negation words with the sentiment lexicons annotated with their polarities and sentiment strengths.

Corpus-based methods treat sentiment classification as a special case of text categorization problem (Pang et al. 2002). They mostly build a sentiment classifier from documents with annotated sentiment polarity. The sentiment supervision can be manually annotated, or automatically collected by sentiment signals like emoticons in tweets or human ratings in reviews. Pang et al. (2002) pioneer to treat the sentiment classification of reviews as a special case of text categorization problem and first investigate machine learning methods. They employ Naive Bayes, Maximum Entropy, and Support Vector Machines (SVM) with a diverse set of features. In their experiments, the best performance is achieved by SVM with bag-of-words features. Following Pang et al.'s work, many studies focus on designing or learning effective features to obtain a better classification performance. On movie and product reviews, Wang and Manning (2012) present NBSVM, which trade-off between Naive Bayes and NB-feature enhanced SVM. Paltoglou and Thelwall (2010) learn feature weights by investigating variants weighting functions from Information Retrieval, such as tf.idf and its BM25 variants. Nakagawa et al. (2010) utilize dependency trees, polarity-shifting rules and conditional random fields with hidden variables to compute the document feature.

The intuition of developing neural network approach is that feature engineering is typically labor intensive. Neural network approaches instead have the ability to discover explanatory factors from the data and make the learning algorithms less dependent on extensive feature engineering. Bespalov et al. (2011) represent each word as a vector (embedding), and then get the vectors for phrases with temporal convolutional network. The document embedding is calculated by averaging the phrase vectors. Le and Mikolov (2014) extend the standard skip-gram and CBOW models Mikolov et al. (2013b) to learn the embeddings for sentences and documents. They represent each document by a dense vector which is trained to predict words in the document. Specifically, the PV-DM model extends the skip-gram model by averaging/concatenating the document vector with context vectors to predict the middle word. The models of Denil et al. (2014); Tang et al. (2015a); Bhatia et al. (2015); Yang et al. (2016); Zhang et al. (2016c) have the same intuition. They model the embedding of sentences from the words, and then use sentence vectors to compose the document vector. Specifically, Denil et al. (2014) use the same convolutional

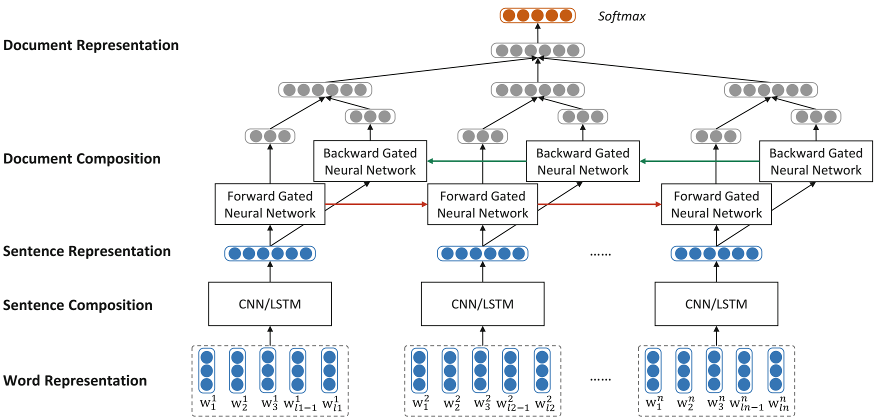

Softmax
Document Representation
Document Composition
Backward Gated Neural Network
Forward Gated Neural Network
Sentence Representation
Sentence Composition
CNN/LSTM
Word Representation
......
w_1^1 w_2^1 w_3^1 w_{l1-1}^1 w_{l1}^1
w_1^2 w_2^2 w_3^2 w_{l2-1}^2 w_{l2}^2
w_1^n w_2^n w_3^n w_{ln-1}^n w_{ln}^n

Fig. 8.20 A neural network architecture for document-level sentiment classification (Tang et al. 2015a).

neural network as the sentence modeling component and the document modeling component. Tang et al. (2015a) use convolutional neural network to calculate the sentence vector, and then use bidirectional gated recurrent neural network to calculate the document embedding. The model is given in Fig. 8.20. Bhatia et al. (2015) calculate document vector based on the structure obtained from the RST parse. Zhang et al. (2016c) calculate sentence vector with recurrent neural network, and then use convolutional network to calculate the document vector. Yang et al. (2016) use two attention layers to get the sentence vectors, and the document vector, respectively. In order to calculate the weights of different words from a sentence and the weights of different sentences of a document, they use two "context" vectors, which are jointly learned in the training process. Joulin et al. (2016) introduces a simple and efficient approach, which averages the word representations into a text representation, and then feeds the results to a linear classifier. Johnson and Zhang (2014, 2015, 2016) develop convolutional neural networks that take one-hot word vector as input and represent a document with the meanings of different regions. The aforementioned studies regard word as the basic computational unit, and compose the document vector based on word representation. Zhang et al. (2015b) and Conneau et al. (2016) use characters as the basic computational units, and explore convolutional architectures to calculate the document vector. The vocabulary for characters is dramatically smaller than the standard vocabulary of words. In Zhang et al. (2015b), the alphabet consists of 70 characters, including 26 English letters, 10 digits, 33 other characters, and the new line character. The model of Zhang et al. (2015b) has 6 convolution layers, and the model of Conneau et al. (2016) consists of 29 layers.

There also exist studies that explore side information such as individual preferences of users or overall qualities of products to improve document-level sentiment classification. For example, Tang et al. (2015b) incorporate user-sentiment consistency and user-text consistency to an existing convolutional neural network. In the

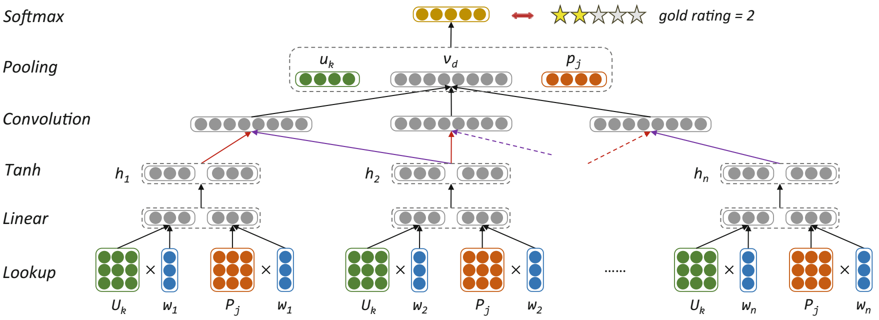


Fig. 8.21 The neural network approach that incorporates user and product information for document- level sentiment classification (Tang et al. 2015b).

user-text consistency, each user is represented as a matrix to modify the meaning of a word. In the user-sentiment consistency, each user is encoded as a vector, which is directly concatenated with the document vector and regarded as a part of the features for sentiment classification. The model is given in Fig. 8.21. Chen et al. (2016) make an extension and develop attention models to take into account the importance of words.

8.5 Fine-Grained Sentiment Analysis

In this section, we introduce the recent advances in fine-grained sentiment analysis using deep learning. Different from sentence/document-level sentiment classification, fine-grained sentiment analysis involves a number of tasks, most of which have their own characteristics. Thus, these tasks are modeled differently, carefully considering their special application settings. Here, we introduce five different topics of fine-grained sentiment analysis, including opinion mining, targeted sentiment analysis, aspect-level sentiment analysis, stance detection, and sarcasm detection.

8.5.1 Opinion Mining

Opinion mining has been a hot topic in the NLP community, which aims to extract structured opinions from user- generated reviews. Figure 8.22 shows several examples of opinion mining. Typically, the task involves two subtasks. First opinion entities such as holders, targets, and expressions are identified, and second we build relations over these entities, for example, the IS-ABOUT relation which specifies the target of a certain opinion expression, and the IS-FROM relation which links an opinion

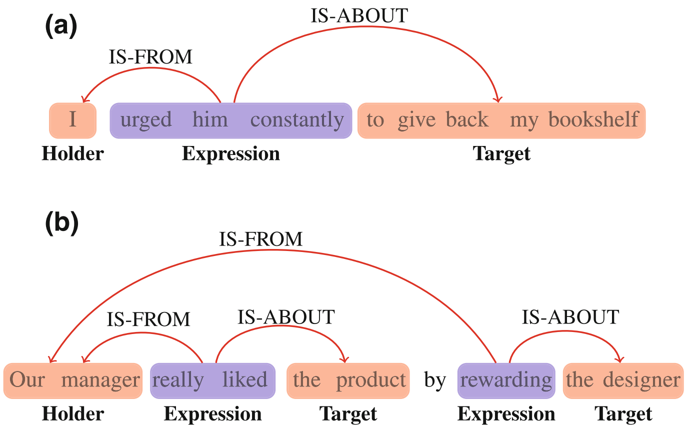


Fig. 8.22 Examples of opinion mining

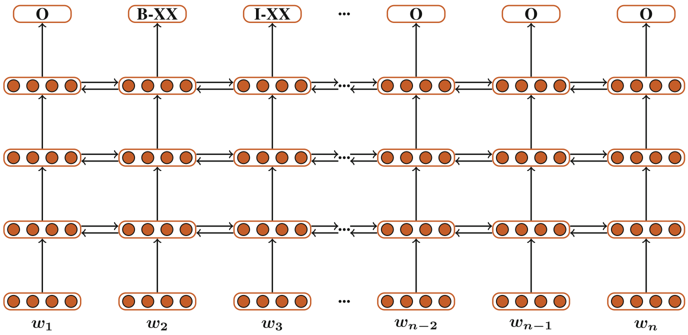


Fig. 8.23 A three-layer Bi-LSTM model for opinion entity detection

expression with its holder. In addition, the classification of sentiment polarities is an important task as well.

Opinion mining is a typical structural learning problem, which has been studied extensively by using traditional statistical models with human-designed discrete features. While recently, motivated by the great success of deep learning models on other NLP tasks, especially on sentiment analysis, neural network-based models have received grown attentions on the task as well. In the below, we describe several representative studies of this task by using neural networks.

The early work of neural network models focuses on the detection of opinion entities, treating the task as a sequence labeling problem to recognize boundaries of opinion entities. Irsoy and Cardie (2014b) investigate the RNN structure for the task. They apply the Elman-type RNNs, studying the effectiveness bidirectional RNN, and observing the influence of the RNN depth, as shown in Fig. 8.23. Their results show that bidirectional RNN can obtain better performances, and a three-layer bidirectional RNN can achieve the best performance.

A similar work is proposed by Liu et al. (2015). They make a comprehensive investigation of RNN variations, including Elman-type RNN, Jordan-type RNN, and LSTM. They study the bidirectionality as well. In addition, they compare three kinds of input word embeddings. They compare these neural network models with discrete models, and make a combination of the two different types of features. Their experiments show that the LSTM neural network combining with discrete features can achieve the best performance.

The above two studies do not involve the identification of the relation between opinion entities. Most recently, Katiyar and Cardie (2016) propose the first neural network that exploits LSTM to jointly perform entity recognition and opinion relation classification. They treat the two subtasks by a multitask learning paradigm, introducing sentence-level training considering both entity boundaries and their relations, based on a shared multilayer bidirectional LSTM. In particular, they define two sequences to denote the distance to their left and right entities of certain relations, respectively. Experimental results on benchmark MPQA datasets show that their neural model achieve the top-performing results.

8.5.2 *Targeted Sentiment Analysis*

Targeted sentiment analysis studies the sentiment polarity toward a certain entity in one sentence. Figure 8.24 shows several examples for the task, where $\{+, -, 0\}$ denote the positive, negative, and neutral sentiment, respectively.

The first neural network model for targeted-dependent sentiment analysis is proposed by Dong et al. (2014a). The model is adapted from their previous work of Dong et al. (2014b), which we have introduced in the sentence-level sentiment analysis. Similarly, they build recursive neural networks from a binarized dependency tree structure, by using multi- compositions from the child nodes. However, this work is different in that they convert the dependency tree according to the input target, making the headword of the target as the root in the resulting tree, not the original head word of the input sentence. Figure 8.25 shows the composition methods and the resulting dependency tree structure, where "phone" is the target.

The above work highly relies on the input dependency parsing trees, which are produced by automatic syntactic parsers. The trees can have errors, thus suffering from the error propagation problem. To avoid the problem, recent studies suggest

I like [**this washing machine**]$_+$! Really convenient and easy to use !

Disgust food of [**the school canteen**]$_-$! I admire myself for eating in the canteen for four years !

Love [**La La Land**]$_+$ most ! Much better than Beauty and the Beast .

I have no interest in playing [**basketball**]$_-$ and also never watch any live of it .

I do not know [**Ryan Gosling**]$_{\mathbf{0}}$, so I cannot answer any questions in your survey .

Fig. 8.24 Targeted sentiment analysis

Fig. 8.25 The framework of Dong et al. (2014a)

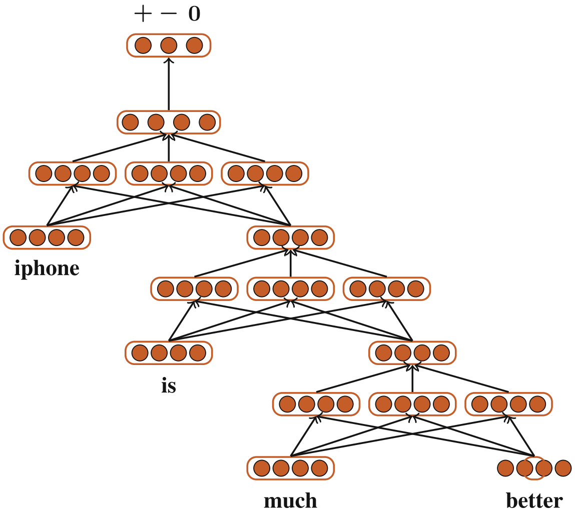
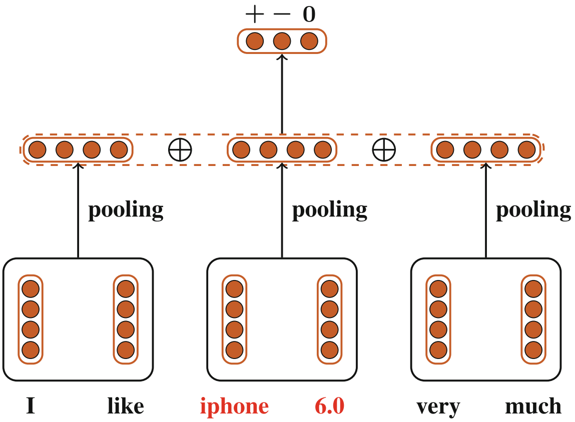


Fig. 8.26 The framework of Vo and Zhang (2015)



conducting targeted sentiment analysis with only raw sentence inputs. Vo and Zhang (2015) exploit various pooling strategies to extract a number of neural features for the task. They first divide the input sentence into three segments by a given target, and then apply different pooling functions over the three segments together with the whole sentence, as shown in Fig. 8.26. The resulting neural features are concatenated for further sentiment polarity prediction.

Recently, several works investigate the effectiveness of RNN for the task, which has brought promising performances in other sentiment analysis tasks. Zhang et al. (2016b) propose to use gated RNN to enhance the representation of sentential words. By using RNN, the resulting representations can capture context-sensitive information, as shown in Fig. 8.27. Further, Tang et al. (2016a) exploit LSTM-RNN as one basic neural layer to encode the input sequential words. Figure 8.28 shows the framework of their work. Both the works have achieved state-of-the-art performances in targeted sentiment analysis.

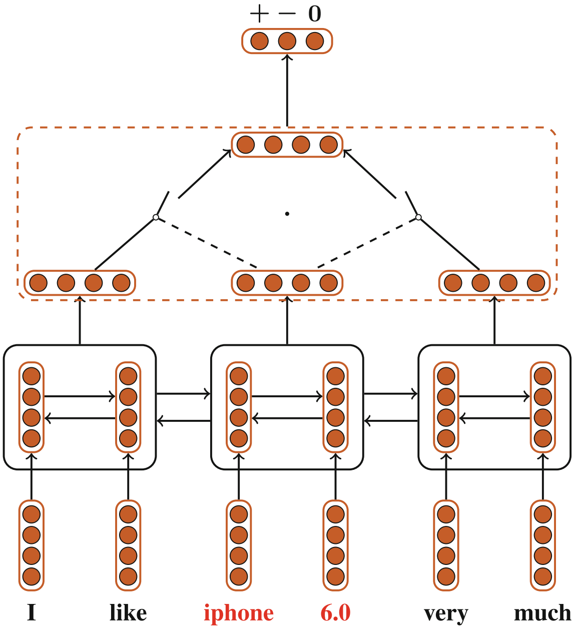


Fig. 8.27 The framework of Zhang et al. (2016b)

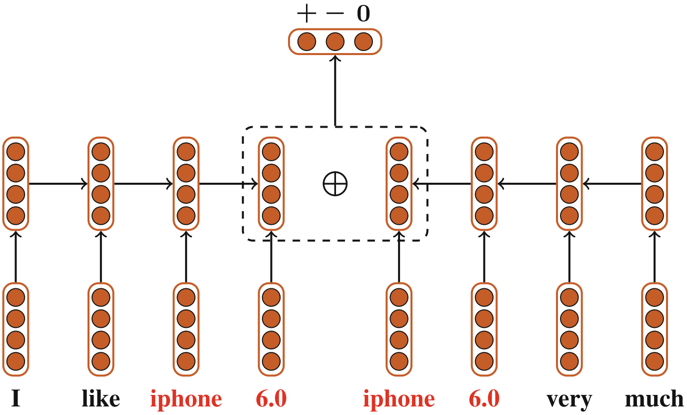
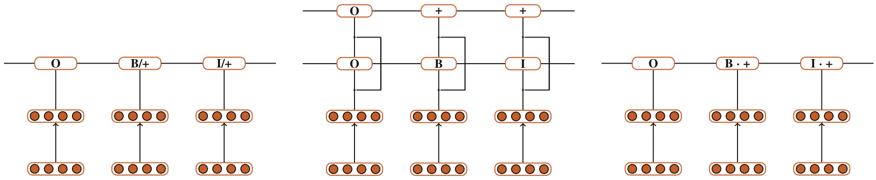


Fig. 8.28 The framework of Tang et al. (2016a)



(a) pipeline (b) joint (c) collapsed

Fig. 8.29 Open domain-targeted sentiment analysis

Besides the use of RNN, Zhang et al. (2016b) present a gated neural network to compose the features of the left, right contexts by target supervision, as shown in Fig. 8.27. The main motivation behind is that the context-neural features should not be equally treated by simply pooling. The task should carefully consider the target as well in order to choose effective features. Liu and Zhang (2017) improve the gated mechanism further, by applying an attention strategy. With the attention, their model achieves the top performances on two benchmark datasets.

Previous work demonstrated that boundaries of the input target is important for the inferring of its sentiment polarities. They assume that well-posed targets are already given, which is not always a real scenario. For example, if we want to determine the sentiment polarities of open targets, it is required to recognize the these targets in advance. Zhang et al. (2015a) study the open domain-targeted sentiment analysis by using neural networks. They investigate the problem under various settings, including pipeline, joint, and collapsed frameworks. Figure 8.29 shows the three frameworks. In addition, they combine the neural and traditional discrete features in a single model, finding that better performances can be obtained consistently under the three settings.

8.5.3 Aspect-Level Sentiment Analysis

Aspect-level sentiment analysis aims to classify the sentiment polarities in a sentence for an aspect. An aspect is one attribute of a target, over which human can express their opinions. Figure 8.30 shows several examples of the task. Usually, the task is aimed to analyze user comments for a certain product, e.g., a hotel, an electronics, or a movie. Products may have a number of aspects. For example, the aspects of a hotel include environment, price, and service, and users usually post a review to express their opinions over certain aspects. Different from targeted sentiment analysis, aspects can be enumerated when the product is given, and the aspect may not be expressed regularly in one review in some cases.

Initially, the task is modeled as a sentence classification problem, thus we can exploit the same method as the sentence-level sentiment classification, expect that the categories are different. Typically, assuming that a product has N aspects which

Sentence	Aspect	Polarity
The screen of the laptop is nice. I like it very much .	screen	positive
It is a choice as a whole, although the owner is not as friendly.	service	negative
The phone is not bad, especially for its strong battery.	battery	positive
I like the movie very much, in particular the story touches me greatly.	screenwriter	positive
I need to change my laptop now, since the key U does work.	keyboard	negative

Fig. 8.30 Aspect-level sentiment analysis

are predefined by expert, the aspect-level sentiment classification is actually a $3N$-classification problem, since each aspect can have three sentiment polarities: positive, negative, and neutral. Lakkaraju et al. (2014) propose a recursive neural network model-based matrix-vector composition for the task, which is similar to Socher et al. (2012) that performs sentence-level sentiment classification.

In later work, the task has been simplified by assuming that aspect has been given in an input sentence, thus it is equivalent to the aforementioned targeted sentiment analysis. Nguyen and Shirai (2015) propose a phrase-based recursive neural network model to the aspect-level sentiment analysis, where the input phrase structure trees are converted from dependency structures along with the input aspects. Tang et al. (2016b) apply a deep memory neural network under the same setting, without using syntactic trees. Their model achieves state-of-the-art performances, and meanwhile is highly efficient in speed in comparison with the neural models that exploit LSTM structures. Figure 8.31 shows their three-layer deep memory neural network. The final features for classification are extracted by attentions with aspect supervision.

In real scenarios, one aspect of a certain product can have several different expressions. Taking the laptop as an example, we can express the aspect screen by display, resolution, and look, which are closely related to screen. If we can group similar aspect phrases into one aspect, the results of aspect-level sentiment analysis are more helpful for further application. Xiong et al. (2016) propose the first neural network model for aspect phrase grouping. They learn representations of aspect phrase by simple multilayer feed-forward neural networks, extracting neural features with attention composition. The model parameters are trained by distant supervision with automatic training examples. Figure 8.32 shown their framework. He et al. (2017) exploit an unsupervised auto-encoder framework for aspect extraction, which can learn the scale of aspect words automatically by attention mechanism.

8.5.4 *Stance Detection*

The goal of stance detection is to recognize the attitude of one sentence toward a certain topic. Generally, the topic is specified for the task as one input, and the other input is the sentence that needs to be classified. Input sentences may not have explicit relations with the given topic. which makes the task rather different with

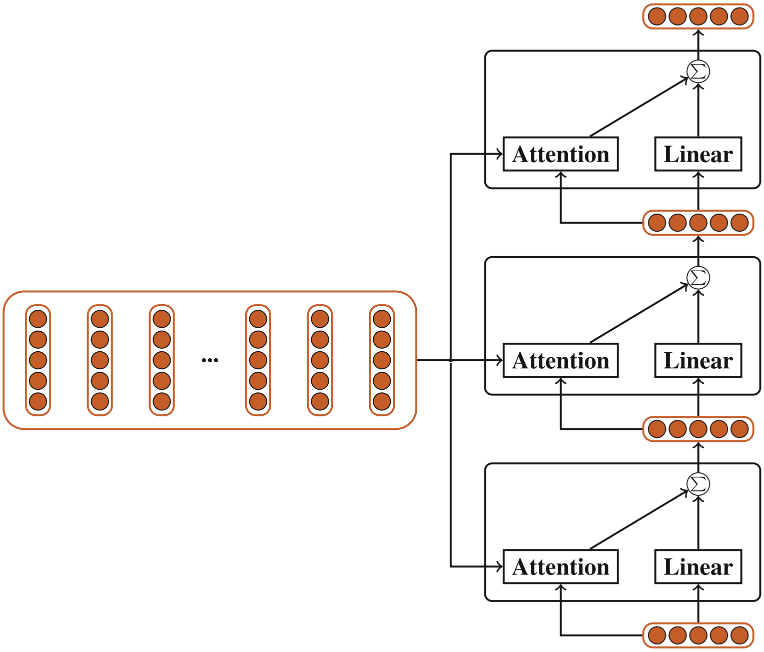


Fig. 8.31 The framework of Tang et al. (2016a)

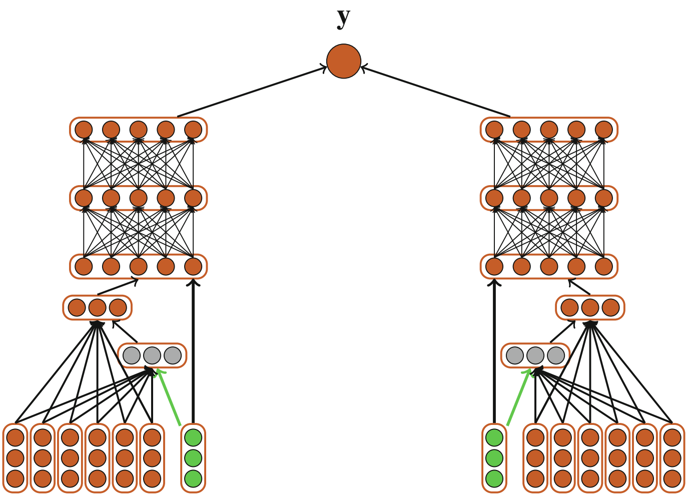


Fig. 8.32 The framework of Xiong et al. (2016)

target/aspect-level sentiment analysis, Thus stance detection is extremely difficult. Figure 8.33 shows several examples of the task.

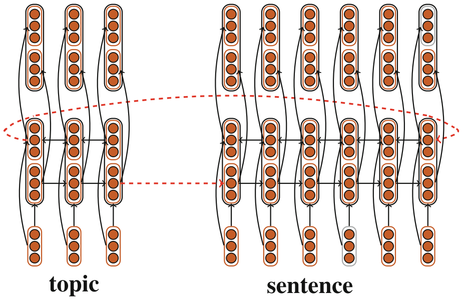
Early work trains independent classifiers for each topic. Thus, the task is treated as a simple 3-way classification problem. For example, Vijayaraghavan et al. (2016) exploit a multilayer CNN model for the task. They integrate both word and character embeddings as inputs in order to solve the unknown words. In the SemEval 2016 task 6 of stance detection, the model of Zarrella and Marsh (2016) achieved the top performance, which builds a neural network based on LSTM-RNN, who has strong capabilities of learning syntactic and semantic features. In addition, motivated by the spirit of transfer learning, they learn the model parameters by the priori knowledge from hashtags in the Twitter, because the raw input sentences of the SemEval task are crawled from Twitter.

The above work models stance classification of different topics independently, which has two main drawbacks. On the one hand, it is not as practical to annotate training examples for each topic, in order to classify the attitudes of a sentence for future topics. On the other hand, several topics may have close relations, for example, "Hillary Clinton" and "Donald Trump" while training the classifiers independently is unable of using this information. Augenstein et al. (2016) propose the first model to train a single model no matter the input topics as a whole, using LSTM neural networks. They model the input sentence and topic jointly, by using the resulting representation of the topics as the input for LSTM over the sentences. Figure 8.34 shows the framework of their method. Their model achieves significantly better performances than the individual classifiers of previous work.

Topic: Climate Change is a Real Concern	
Academy of Science talk Tech solutions for climate change with Barry Brook.	Favor
This just in, an ocean wave just broke an inch further on the beach than normal!	Against
I love this Pope. I don't care what religion you are, this guy is awesome.	NULL
Topic: Feminist Movement	
Because women are seen as "soft," and "emotional" in the eyes of male politicians.	Favor
If the confederate flag offends you, good. Stop making things politically correct.	Against
People say I'm young to be into politics. Honestly, I just stand for what I believe in.	NULL

Fig. 8.33 Examples of stance detection

Fig. 8.34 Conditional LSTM for stance detection



8.5.5 Sarcasm Recognition

In this section, we discuss a special language phenomenon that has close connections with sentiment analysis, namely sarcasm or irony. This phenomenon usually makes change of a sentence's literal meaning, and greatly influence the sentiment expressed by the sentence. Figure 8.35 shows several examples.

Typically, sarcasm detection is modeled as a binary classification problem, which is similar with sentence-level sentiment analysis is essential. The major difference between the two tasks lies in their goals. Ghosh and Veale (2016) study various neural network models for the task in detail, including CNN, LSTM, and deep feed-forward neural networks. They present several different neural models, and investigate their effectiveness empirically. The experimental results show that a combination of these neural networks can bring the best performances. The final model is composed by a two-layer CNN, a two-layer LSTM and another one feed-forward layer, as shown in Fig. 8.36.

For sarcasm detection in social media such as Twitter, author-based information is one kind of useful features. Zhang et al. (2016a) propose a contextualized neural model for Twitter sarcasm recognition. Concretely, they extract a set of salient words from the tweet authors' historical posts, using these words to represent the tweet author. Their proposed neural network model consists two parts, as shown in Fig. 8.37, one being a gated RNN to represent sentences, and the other being a simple pooling neural network to represent tweet author.

Sometimes idiots just brighten my day , incredible !!
I love waking up in the dark and coming home in the dark.
Now I know where you get your manners from.
The only bad thing about weed is getting caught with it #makessense
My life is so exciting......., I just can not believe what have happened.
Glad my dryer has ruined two of my camis now

Fig. 8.35 Sarcasm examples

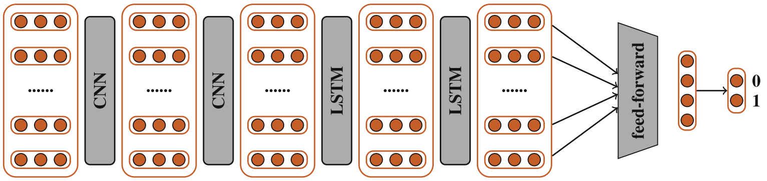


Fig. 8.36 The framework of Ghosh and Veale (2016)

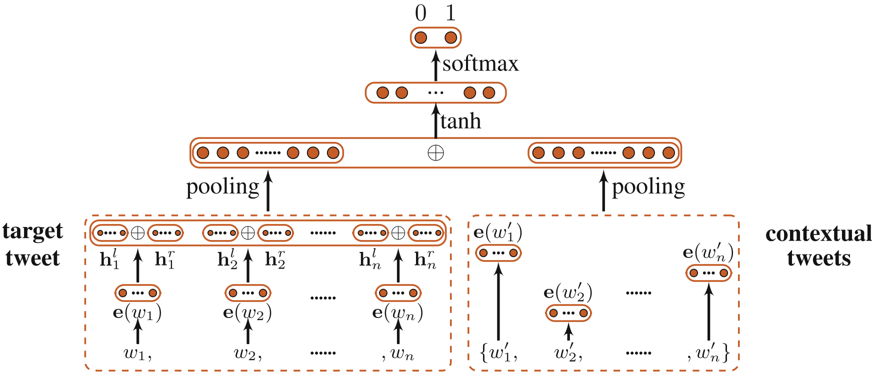


Fig. 8.37 The framework of Zhang et al. (2016a).

8.6 Summary

In this chapter, we give an overview on the recent success of neural network approaches in sentiment analysis. We first describe how to integrate sentiment information of texts to learn sentiment-specific word embeddings. Then, we describe sentiment classification of sentences and documents, both of which require semantic composition of texts. We then present how to develop neural network models to deal with fine-grained tasks.

Despite deep learning approaches have achieved promising performances on sentiment analysis tasks in recent years, there are some potential directions to further improve this area. The first direction is explainable sentiment analysis. The current deep learning models are accurate yet unexplainable. Leveraging knowledge from cognitive science, common sense knowledge, or extracted knowledge from text corpus might be a potential direction to improve this area. The second direction is learning a robust model for a new domain. The performance of a deep learning model depends on the amount and the quality of the training data. Therefore, how to learn a robust sentiment analyzer for a domain with little/no annotated corpus is very challenging yet important for real application. The third direction is how to understand the emotion. Majority of existing studies focus on opinion expressions, targets, and holders. Recently, new attributes have been suggested to better understand the emotion, such as opinion causes and stances. Pushing forward this area requires powerful models and large corpora. The fourth direction is fine-grained sentiment analysis, which receives increasing interests recently. Improving this area requires larger training corpus.

References

Augenstein, I., Rocktäschel, T., Vlachos, A., & Bontcheva, K. (2016). Stance detection with bidirectional conditional encoding. In *EMNLP2016* (pp. 876–885).

Bai, B., Weston, J., Grangier, D., Collobert, R., Sadamasa, K., Qi, Y., et al. (2010). Learning to rank with (a lot of) word features. *Information Retrieval*, *13*(3), 291–314.

Baker, L. D. & McCallum, A. K. (1998). Distributional clustering of words for text classification. In *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (pp. 96–103). ACM.

Bengio, Y., Ducharme, R., Vincent, P., & Jauvin, C. (2003). A neural probabilistic language model. *Journal of Machine Learning Research, 3*(Feb), 1137–1155.

Bespalov, D., Bai, B., Qi, Y., & Shokoufandeh, A. (2011). Sentiment classification based on supervised latent n-gram analysis. In *Proceedings of the 20th ACM International Conference on Information and Knowledge Management* (pp. 375–382). ACM.

Bhatia, P., Ji, Y., & Eisenstein, J. (2015). Better document-level sentiment analysis from rst discourse parsing. arXiv:1509.01599.

Brown, P. F., Desouza, P. V., Mercer, R. L., Pietra, V. J. D., & Lai, J. C. (1992). Class-based n-gram models of natural language. *Computational Linguistics*, *18*(4), 467–479.

Chen, X., Qiu, X., Zhu, C., Wu, S., & Huang, X. (2015). Sentence modeling with gated recursive neural network. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing* (pp. 793–798). Lisbon, Portugal: Association for Computational Linguistics.

Chen, H., Sun, M., Tu, C., Lin, Y., & Liu, Z. (2016). Neural sentiment classification with user and product attention. In *Proceedings of EMNLP*.

Collobert, R. & Weston, J. (2008). A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th International Conference on Machine Learning* (pp. 160–167). ACM.

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

Conneau, A., Schwenk, H., Barrault, L., & Lecun, Y. (2016). Very deep convolutional networks for natural language processing. arXiv:1606.01781.

Deerwester, S., Dumais, S. T., Furnas, G. W., Landauer, T. K., & Harshman, R. (1990). Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, *41*(6), 391.

Deng, L. & Wiebe, J. (2015). MPQA 3.0: An entity/event-level sentiment corpus. In *HLT-NAACL* (pp. 1323–1328).

Denil, M., Demiraj, A., Kalchbrenner, N., Blunsom, P., & de Freitas, N. (2014). Modelling, visualising and summarising documents with a single convolutional neural network. arXiv:1406.3830.

Ding, X., Liu, B., & Yu, P. S. (2008). A holistic lexicon-based approach to opinion mining. In *Proceedings of the 2008 International Conference on Web Search and Data Mining* (pp. 231–240). ACM.

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

Dong, L., Wei, F., Zhou, M., & Xu, K. (2014b). Adaptive multi-compositionality for recursive neural models with applications to sentiment analysis. In *AAAI* (pp. 1537–1543).

dos Santos, C. & Gatti, M. (2014). Deep convolutional neural networks for sentiment analysis of short texts. In *Proceedings of COLING 2014, The 25th International Conference on Computational Linguistics: Technical Papers* (pp. 69–78). Dublin, Ireland: Dublin City University and Association for Computational Linguistics.

Faruqui, M., Dodge, J., Jauhar, S. K., Dyer, C., Hovy, E., & Smith, N. A. (2014). Retrofitting word vectors to semantic lexicons. arXiv:1411.4166.

Gan, Z., Pu, Y., Henao, R., Li, C., He, X., & Carin, L. (2016). Unsupervised learning of sentence representations using convolutional neural networks. arXiv:1611.07897.

Ghosh, A., & Veale, D. T. (2016). Fracking sarcasm using neural network. In *Proceedings of the 7th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis* (pp. 161–169).

Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep learning*. Cambridge: MIT Press.

Gutmann, M. U., & Hyvärinen, A. (2012). Noise-contrastive estimation of unnormalized statistical models, with applications to natural image statistics. *Journal of Machine Learning Research,13*(Feb), 307–361.

Harris, Z. S. (1954). *Distributional structure. Word*, *10*(2–3), 146–162.

He, R., Lee, W. S., Ng, H. T., & Dahlmeier, D. (2017). An unsupervised neural attention model for aspect extraction. In *Proceedings of the 55th ACL* (pp. 388–397). Vancouver, Canada: Association for Computational Linguistics.

Hill, F., Cho, K., & Korhonen, A. (2016). Learning distributed representations of sentences from unlabelled data. In *NAACL* (pp. 1367–1377).

Hu, M. & Liu, B. (2004). Mining and summarizing customer reviews. In *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (pp. 168–177). ACM.

Huang, P.-S., He, X., Gao, J., Deng, L., Acero, A., & Heck, L. (2013). Learning deep structured semantic models for web search using clickthrough data. In *Proceedings of the 22nd ACM International Conference on Information and Knowledge Management* (pp. 2333–2338). ACM.

Huang, E. H., Socher, R., Manning, C. D., & Ng, A. Y. (2012). Improving word representations via global context and multiple word prototypes. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1* (pp. 873–882). Association for Computational Linguistics.

Irsoy, O. & Cardie, C. (2014a). Deep recursive neural networks for compositionality in language. In *Advances in neural information processing systems* (pp. 2096–2104).

Irsoy, O. & Cardie, C. (2014b). Opinion mining with deep recurrent neural networks. In *Proceedings of the 2014 EMNLP* (pp. 720–728).

Johnson, R. & Zhang, T. (2014). Effective use of word order for text categorization with convolutional neural networks. arXiv:1412.1058.

Johnson, R. & Zhang, T. (2015). Semi-supervised convolutional neural networks for text categorization via region embedding. In *Advances in neural information processing systems* (pp. 919–927).

Johnson, R. & Zhang, T. (2016). Supervised and semi-supervised text categorization using LSTM for region embeddings. arXiv:1602.02373.

Joulin, A., Grave, E., Bojanowski, P., & Mikolov, T. (2016). Bag of tricks for efficient text classification. arXiv:1607.01759.

Jurafsky, D. (2000). *Speech and language processing*. New Delhi: Pearson Education India.

Kalchbrenner, N., Grefenstette, E., & Blunsom, P. (2014). A convolutional neural network for modelling sentences. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)* (pp. 655–665), Baltimore, Maryland: Association for Computational Linguistics.

Katiyar, A. & Cardie, C. (2016). Investigating LSTMs for joint extraction of opinion entities and relations. In *Proceedings of the 54th ACL* (pp. 919–929).

Kim, Y. (2014). Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)* (pp. 1746–1751). Doha, Qatar: Association for Computational Linguistics.

Labutov, I., & Lipson, H. (2013). Re-embedding words. In *ACL* (Vol. 2, pp. 489–493).

Lakkaraju, H., Socher, R., & Manning, C. (2014). Aspect specific sentiment analysis using hierarchical deep learning. In *NIPS Workshop on Deep Learning and Representation Learning*.

Le, Q. V. & Mikolov, T. (2014). Distributed representations of sentences and documents. In *ICML* (Vol. 14, pp. 1188–1196).

Lebret, R., Legrand, J., & Collobert, R. (2013). *Is deep learning really necessary for word embeddings?*. Idiap: Technical Report.

LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature*, *521*(7553), 436–444.

Lei, T., Barzilay, R., & Jaakkola, T. (2015). Molding CNNs for text: Non-linear, non-consecutive convolutions. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing* (pp. 1565–1575). Lisbon, Portugal: Association for Computational Linguistics.

Levy, O. & Goldberg, Y. (2014). Dependency-based word embeddings. In *ACL*, (Vol. 2, pp. 302–308). Citeseer.

Li, J. & Jurafsky, D. (2015). Do multi-sense embeddings improve natural language understanding? arXiv:1506.01070.

Li, J., Luong, M.-T., Jurafsky, D., & Hovy, E. (2015). When are tree structures necessary for deep learning of representations? arXiv:1503.00185.

Liu, J. & Zhang, Y. (2017). Attention modeling for targeted sentiment. In *Proceedings of EACL* (pp. 572–577).

Liu, P., Joty, S., & Meng, H. (2015). Fine-grained opinion mining with recurrent neural networks and word embeddings. In *Proceedings of the 2015 EMNLP* (pp. 1433–1443).

Liu, B. (2012). Sentiment analysis and opinion mining. *Synthesis Lectures on Human Language Technologies*, *5*(1), 1–167.

Lund, K., & Burgess, C. (1996). Producing high-dimensional semantic spaces from lexical co-occurrence. *Behavior Research Methods, Instruments, and Computers*, *28*(2), 203–208.

Ma, M., Huang, L., Zhou, B., & Xiang, B. (2015). 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 (Volume 2: Short Papers)* (pp. 174–179), Beijing, China: Association for Computational Linguistics.

Maas, A. L., Daly, R. E., Pham, P. T., Huang, D., Ng, A. Y., & Potts, C. (2011). Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1* (pp. 142–150). Association for Computational Linguistics.

Manning, C. D., Schütze, H., et al. (1999). *Foundations of Statistical Natural Language Processing* (Vol. 999). Cambridge: MIT Press.

Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013a). Efficient estimation of word representations in vector space. arXiv:1301.3781.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., & Dean, J. (2013b). Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems* (pp. 3111–3119).

Mishra, A., Dey, K., & Bhattacharyya, P. (2017). Learning cognitive features from gaze data for sentiment and sarcasm classification using convolutional neural network. In *Proceedings of the 55th ACL* (pp. 377–387). Vancouver, Canada: Association for Computational Linguistics.

Mnih, A. & Hinton, G. (2007). Three new graphical models for statistical language modelling. In *Proceedings of the 24th International Conference on Machine Learning* (pp. 641–648). ACM.

Mnih, A. & Kavukcuoglu, K. (2013). Learning word embeddings efficiently with noise-contrastive estimation. In *Advances in neural information processing systems* (pp. 2265–2273).

Morin, F. & Bengio, Y. (2005). Hierarchical probabilistic neural network language model. In *Aistats* (Vol. 5, pp. 246–252). Citeseer.

Mou, L., Peng, H., Li, G., Xu, Y., Zhang, L., & Jin, Z. (2015). Discriminative neural sentence modeling by tree-based convolution. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing* (pp. 2315–2325). Lisbon, Portugal: Association for Computational Linguistics.

Nakagawa, T., Inui, K., & Kurohashi, S. (2010). Dependency tree-based sentiment classification using CRFs with hidden variables. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics* (pp. 786–794). Association for Computational Linguistics.

Nguyen, T. H. & Shirai, K. (2015). PhraseRNN: Phrase recursive neural network for aspect-based sentiment analysis. In *EMNLP* (pp. 2509–2514).

Paltoglou, G. & Thelwall, M. (2010). A study of information retrieval weighting schemes for sentiment analysis. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics* (pp. 1386–1395). Association for Computational Linguistics.

Pang, B., & Lee, L. (2005). Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics* (pp. 115–124). Association for Computational Linguistics.

Pang, B., Lee, L., & Vaithyanathan, S. (2002). Thumbs up?: Sentiment classification using machine learning techniques. In *Proceedings of the ACL-02 Conference on Empirical Methods in Natural Language Processing-Volume 10* (pp. 79–86). Association for Computational Linguistics.

Pang, B., Lee, L., et al. (2008). Opinion mining and sentiment analysis. Foundations and trends®. *Information Retrieval*, *2*(1–2), 1–135.

Qian, Q., Huang, M., Lei, J., & Zhu, X. (2017). Linguistically regularized LSTM for sentiment classification. In *Proceedings of the 55th ACL* (pp. 1679–1689). Vancouver, Canada: Association for Computational Linguistics.

Qiu, S., Cui, Q., Bian, J., Gao, B., & Liu, T.-Y. (2014). Co-learning of word representations and morpheme representations. In *COLING* (pp. 141–150).

Ren, Y., Zhang, Y., Zhang, M., & Ji, D. (2016a). Context-sensitive twitter sentiment classification using neural network. In *AAAI* (pp. 215–221).

Ren, Y., Zhang, Y., Zhang, M., & Ji, D. (2016b). Improving twitter sentiment classification using topic-enriched multi-prototype word embeddings. In *AAAI* (pp. 3038–3044).

Shen, Y., He, X., Gao, J., Deng, L., & Mesnil, G. (2014). Learning semantic representations using convolutional neural networks for web search. In *Proceedings of the 23rd International Conference on World Wide Web* (pp. 373–374). ACM.

Socher, R., Huval, B., Manning, C. D., & Ng, A. Y. (2012). Semantic compositionality through recursive matrix-vector spaces. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning* (pp. 1201–1211). Jeju Island, Korea: Association for Computational Linguistics.

Socher, R., Perelygin, A., Wu, J., Chuang, J., Manning, C. D., Ng, A., & Potts, C. (2013). Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing* (pp. 1631–1642). Seattle, Washington, USA: Association for Computational Linguistics.

Taboada, M., Brooke, J., Tofiloski, M., Voll, K., & Stede, M. (2011). Lexicon-based methods for sentiment analysis. *Computational Linguistics*, *37*(2), 267–307.

Tai, K. S., Socher, R., & Manning, C. D. (2015). Improved semantic representations from tree-structured long short-term memory networks. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)* (pp. 1556–1566). Beijing, China: Association for Computational Linguistics.

Tang, D., Qin, B., & Liu, T. (2015a). Document modeling with gated recurrent neural network for sentiment classification. In *EMNLP* (pp. 1422–1432).

Tang, D., Qin, B., & Liu, T. (2015b). Learning semantic representations of users and products for document level sentiment classification. In *ACL* (Vol. 1, pp. 1014–1023).

Tang, D., Qin, B., & Liu, T. (2016a). Aspect level sentiment classification with deep memory network. In *Proceedings of the 2016 EMNLP* (pp. 214–224).

Tang, D., Qin, B., Feng, X., & Liu, T. (2016b). Effective LSTMs for target-dependent sentiment classification. In *Proceedings of COLING, 2016* (pp. 3298–3307).

Tang, D., Wei, F., Yang, N., Zhou, M., Liu, T., & Qin, B. (2014). Learning sentiment-specific word embedding for twitter sentiment classification. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)* (pp. 1555–1565). Baltimore, Maryland: Association for Computational Linguistics.

Tang, D., Wei, F., Qin, B., Yang, N., Liu, T., & Zhou, M. (2016c). Sentiment embeddings with applications to sentiment analysis. *IEEE Transactions on Knowledge and Data Engineering*, *28*(2), 496–509.

Teng, Z., & Zhang, Y. (2016). Bidirectional tree-structured lstm with head lexicalization. arXiv:1611.06788.

Teng, Z., Vo, D. T., & Zhang, Y. (2016). Context-sensitive lexicon features for neural sentiment analysis. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing* (pp. 1629–1638). Austin, Texas: Association for Computational Linguistics.

Turney, P. D. (2002). Thumbs up or thumbs down?: Semantic orientation applied to unsupervised classification of reviews. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics* (pp. 417–424). Association for Computational Linguistics.

Vijayaraghavan, P., Sysoev, I., Vosoughi, S., & Roy, D. (2016). Deepstance at semeval-2016 task 6: Detecting stance in tweets using character and word-level CNNs. In *SemEval-2016* (pp. 413–419).

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

Wang, S. & Manning, C. D. (2012). Baselines and bigrams: Simple, good sentiment and topic classification. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Short Papers-Volume 2* (pp. 90–94). Association for Computational Linguistics.

Wang, X., Liu, Y., Sun, C., Wang, B., & Wang, X. (2015). Predicting polarities of tweets by composing word embeddings with long short-term memory. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)* (pp. 1343–1353), Beijing, China: Association for Computational Linguistics.

Xiong, S., Zhang, Y., Ji, D., & Lou, Y. (2016). Distance metric learning for aspect phrase grouping. In *Proceedings of COLING*, *2016* (pp. 2492–2502).

Yang, Z., Yang, D., Dyer, C., He, X., Smola, A., & Hovy, E. (2016). Hierarchical attention networks for document classification. In *Proceedings of NAACL-HLT* (pp. 1480–1489).

Yin, W. & Schütze, H. (2015). Multichannel variable-size convolution for sentence classification. In *Proceedings of the Nineteenth Conference on Computational Natural Language Learning* (pp. 204–214). Beijing, China: Association for Computational Linguistics.

Yogatama, D., Faruqui, M., Dyer, C., & Smith, N. A. (2015). Learning word representations with hierarchical sparse coding. In *ICML* (pp. 87–96).

Zarrella, G. & Marsh, A. (2016). Mitre at semeval-2016 task 6: Transfer learning for stance detection. In *SemEval-2016* (pp. 458–463).

Zhang, R., Lee, H., & Radev, D. R. (2016c). Dependency sensitive convolutional neural networks for modeling sentences and documents. In *Proceedings of the 2016 NAACL* (pp. 1512–1521). San Diego, California: Association for Computational Linguistics.

Zhang, Y., Roller, S., & Wallace, B. C. (2016d). MGNC-CNN: A simple approach to exploiting multiple word embeddings for sentence classification. In *Proceedings of the 2016 NAACL* (pp. 1522–1527). San Diego, California: Association for Computational Linguistics.

Zhang, M., Zhang, Y., & Fu, G. (2016a). Tweet sarcasm detection using deep neural network. In *Proceedings of COLING 2016, The 26th International Conference on Computational Linguistics: Technical Papers* (pp. 2449–2460). Osaka, Japan: The COLING 2016 Organizing Committee.

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

Zhang, M., Zhang, Y., & Vo, D.-T. (2016b). Gated neural networks for targeted sentiment analysis. In *AAAI* (pp. 3087–3093).

Zhang, X., Zhao, J., & LeCun, Y. (2015b). Character-level convolutional networks for text classification. In *Advances in neural information processing systems* (pp. 649–657).

Zhao, H., Lu, Z., & Poupart, P. (2015). Self-adaptive hierarchical sentence model. arXiv:1504.05070.

Zhu, X.-D., Sobhani, P., & Guo, H. (2015). Long short-term memory over recursive structures. In *ICML* (pp. 1604–1612).