Focus Article

Deep learning for sentiment analysis: successful approaches and future challenges

Duyu Tang, Bing Qin* and Ting Liu

Sentiment analysis (also known as opinion mining) is an active research area in natural language processing. It aims at identifying, extracting and organizing sentiments from user generated texts in social networks, blogs or product reviews. A lot of studies in literature exploit machine learning approaches to solve sentiment analysis tasks from different perspectives in the past 15 years. 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 expert 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 of sentences/documents, sentiment extraction and sentiment lexicon learning. In this paper, we provide an overview of the successful deep learning approaches for sentiment analysis tasks, lay out the remaining challenges and provide some suggestions to address these challenges.

How to cite this article:
WIREs Data Mining Knowl Discov 2015, 5:292–303. doi: 10.1002/widm.1171

INTRODUCTION

Sentiment analysis and opinion mining[1–3] are the field that analyze people's opinions, sentiments, emotions from user generated texts. It is one of the most active research areas in natural language processing,[4,5] and is also widely studied in data mining, web mining, and social media analytics as sentiments are key influencers of human behaviors. With the rapid growth of social media such as Twitter, Facebook, and online review sites such as IMDB, Amazon, Yelp, sentiment analysis draws growing attentions from both research and industry communities.

According to the definition by Liu et al.,[2] sentiment (or an opinion) is a quintuple, ($\underline{e}$, $\underline{a}$, $\underline{s}$, $\underline{h}$, $\underline{t}$), where $\underline{e}$ is the name of an entity, $\underline{a}$ is the aspect of $\underline{e}$, $\underline{s}$ is the sentiment on aspect $\underline{a}$ of entity, $\underline{e}$, $\underline{h}$ is the opinion holder, and $\underline{t}$ is the time when the opinion is expressed by $\underline{h}$. In this definition, the sentiment $\underline{s}$ 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.[6,7] Let us take a toy example to better explain the definition of 'sentiment.' Supposing an Amazon user called Tom posted a review '*The photos from my Samsung camera are not that great, but the battery life is great.*' at June 4, 2015. In this example, there are two sentiment quintuples, namely (*Samsung*, *photo*, *negative*, *Tom*, *June 4*, *2015*) and (*Samsung*, *battery life*, *positive*, *Tom*, *June 4*, *2015*). Based on the definition of 'sentiment,' the objective of 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

*Correspondence to: qinb@ir.hit.edu.cn

Department of Computer Science and Technology, Harbin Institute of Technology, Harbin, China

Conflict of interest: The authors have declared no conflicts of interest for this article.

example, the task of document level sentiment classification[8] targets at the third component (sentiment such as positive, negative, and neutral) while ignoring the other aspects. The task of fine-grained opinion extraction[9] focuses on the first four components of the quintuple. In the past 15 years, machine learning driven methods almost dominate sentiment analysis tasks. As feature representation greatly affects the performance of a machine learner,[10] a lot of studies in literature focus on effective features by hand with domain expertise and careful engineering.[11,12] But this can be avoided by representation learning algorithms, which automatically discover discriminative and explanatory text representations from data.[13] Deep learning is a kind of representation learning approach. It learns multiple levels of representation with nonlinear neural networks, each of which transforms the representation at one level into a representation at a higher and more abstract level.[14] The learned representations can be naturally used as features and applied for detection or classification tasks.

In this study, we introduce successful deep learning algorithms for sentiment analysis. The notation of 'deep learning' in this article stands for learning continuous and real-valued text representation/feature automatically from data, mostly with neural network approaches. We first describe the methods to learn continuous word representation, also called word embedding, as words are the basic computational units of natural language. These word embeddings can be used as inputs to subsequent sentiment analysis tasks, so that we describe how word embeddings are investigated for different sentiment analysis tasks. In particular, we describe semantic compositional methods that compute representations of longer expressions (e.g., sentence or document) for sentence/document level sentiment classification task,[15,16] followed by neural sequential models for fine-grained opinion extraction.[17] As sentiment lexicon is an important resource for many sentiment analysis systems, we also describe neural methods to build large-scale sentiment lexicons.[18] We finally conclude this study and provide some future directions.

WORD REPRESENTATION

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 straight-forward way is to encode a word w_i 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.[19,20] 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 corresponds 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,[21] which states that words in similar contexts have similar meanings. To this end, many matrix factorization methods can be viewed as modeling word representations. For example, Latent Semantic Indexing (LSI)[22] 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[23] 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[24] is also investigated to learn word embeddings over 'term-term' co-occurrence statistics. As 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[25] tackles this problem and takes the supervised information of a targeted task (e.g., information retrieval) into consideration. They learn the embedding model from click-through data with a margin ranking loss.

With the revival of interest in deep learning and neural network,[10,14,26] a surge of studies learn word embeddings with neural network. A pioneered work in this field is given by Bengio et al.,[27] 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 w_i and its preceding context words, the algorithm first maps each context word to its continuous vector with a shared lookup table. Afterward, context word vectors are fed to a feed-forward neural network with softmax as output layer to predict the conditional probability of next word w_i. The parameters of neural network and lookup table are jointly estimated with back propagation. Following Bengio et al.'s[27] work, a lot of approaches are proposed to speed-up training processing or capturing richer semantic information. Bengio et al.[27] 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[28] develop hierarchical softmax to decompose the conditional probability with a hierarchical binary tree. Mnih and Hinton[29] introduce a log-bilinear language model. Collobert and Weston[30] train word embeddings with a ranking-type hinge loss function[31] by replacing the middle word within a window with a randomly selected one. Mikolov et al.[32,33] introduce Continuous Bag-of-Words (CBOW) and Continuous Skip-gram, and release the popular word2vec (https://code.google.com/p/word2vec/) 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[34] accelerate the word embedding learning procedure with Noise Contrastive Estimation.[35] There are also many algorithms developed for capturing richer semantic information, including global document information,[36] word morphemes,[37] dependency-based contexts,[38] word–word co-occurrence,[39] sense of ambiguous words,[40] semantic lexical information in WordNet,[41] and hierarchical relations between words.[42]

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[43] as the two words have similar usages and grammatical roles, but it becomes a disaster for sentiment analysis as they 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.[44] introduce a probabilistic topic model by inferring the polarity of a sentence based on the embedding of each word it contains. Labutov and Lipson[45] re-embed an existing word embedding with logistic regression by regarding sentiment supervision of sentences as a regularization item. Tang et al.[46] extend the C&W model[30] and develop three neural networks to learn sentiment-specific word embedding from Twitter. The tweets containing positive and negative emoticons are used as training data, regarding positive and negative emoticon signals as sentiment indicators. The learned word embeddings are applied for Twitter sentiment classification, and perform comparable performance with the state-of-the-art hand-crafted features[11] on SemEval 2013 dataset. They also build a system named 'Coooolll' and participate in the Twitter sentiment classification evaluation in SemEval 2014.[47] The system yields the second place among 45 participants. Holding a similar idea, Tang et al.[18] extend SkipGram method[33] and leverage sentiment of texts for word embedding learning. Brief illustrations of two neural models for learning sentiment-specific word embeddings are given in Figure 1.

SENTIMENT CLASSIFICATION

Sentiment classification is a fundamental and extensively studied area in sentiment analysis.[1,2] It targets at determining the sentiment polarity (positive or

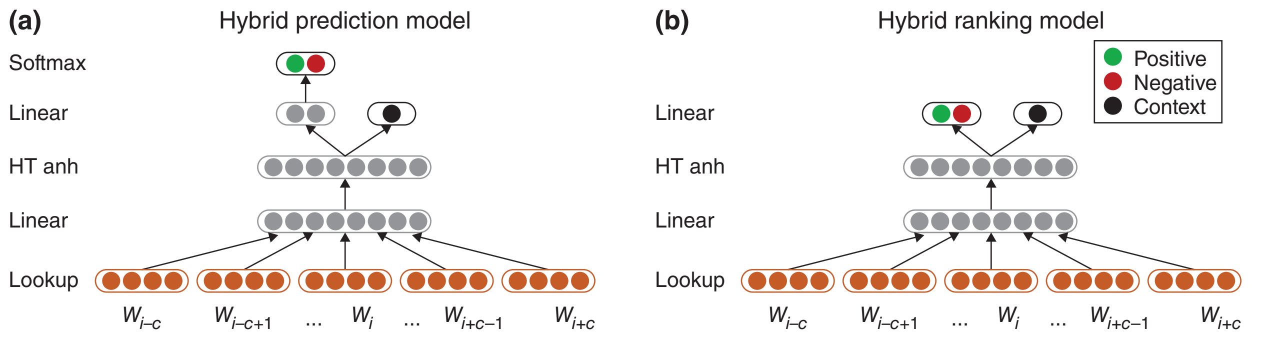

FIGURE 1 | Neural models for learning sentiment-specific word embeddings. The hybrid prediction model is proposed by Tang et al.[46] and the hybrid ranking model is introduced by Tang et al.[18]

negative) of a sentence (or a document) based on its textual content. In the literature, existing studies for sentiment classification are dominated by two mainstream directions: lexicon-based approach and corpus-based approach.

Lexicon-based approaches[48,49] mostly use a dictionary of sentiment words with their associated sentiment polarity, and incorporate negation and intensification to compute the sentiment polarity for each sentence (or document). A representative lexicon-based method is given by Turney,[48] which consists of three steps. He first extract phrases if their postags conform to the predefined patterns. Afterwards the sentiment polarity of each extracted phrase is estimated through point-wised 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 and Liu[50] apply negation words like 'not,' 'never,' 'cannot,' and contrary words like 'but' to enhance the performance of lexicon-based method. Taboada et al.[49] integrate intensifications and negation words with the sentiment lexicons annotated with their polarity and sentiment strength. Thelwall et al.[51] develop SentiStrength using sentiment lexicons and linguistic rules to detect the sentiment strength of tweets. Reckman et al.[52] develop an entirely rule-based system for Twitter sentiment classification. They use lexicalized hand-written rules, each of which is a pattern that matches words or sequences of words.

Corpus-based methods treat sentiment classification as a special case of text categorization problem.[8] They mostly build the sentiment classifier from sentences with annotated sentiment polarity. The sentiment supervision can be manually annotated, or automatically collected by sentiment signals like emoticons in tweets[53] or human ratings in reviews.[44] Pang et al.[8] pioneer to treat the sentiment classification of reviews as a special case of text categorization problem and first investigate machine learning methods. A brief illustration is given in Figure 2. 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 feature. 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[54] present NBSVM, which trades-off between Naive Bayes and NB-feature enhanced SVM. Paltoglou and Thelwall[55] learn feature weights by investigating variants weighting functions from Information Retrieval, such as tf.idf and its BM25 variants. Nakagawa et al.[56] utilize dependency trees, polarity-shifting rules and conditional random fields (CRFs) with hidden variables to compute the document feature. On Twitter, Jiang et al.[57] use lexicon features and syntactic and POS tagging features. NRC-Canada[11] develop a state-of-the-art Twitter sentiment classifier in SemEval 2013 by using a variety of sentiment lexicons and hand-crafted features.

Feature engineering is important but labor-intensive. It is therefore desirable to discover explanatory factors from the data and make the learning algorithms less dependent on extensive feature engineering. With the rapid growing of deep learning (representation learning[10,26]), many recent studies focus on learning the low-dimensional, dense, and real-valued vector as text features for sentiment analysis without any feature engineering. Existing deep learning methods for sentiment classification typically include two stages. In the first stage, they learn word embeddings from text corpus.[33,46] In the second stage, word embeddings are applied to producing the representations of sentences/documents with semantic composition.[58] Embedding learning algorithms have been described in previous Section. Existing composition learning approaches are typically based on the principle of compositionality,[59] which states that the meaning of a longer expression (e.g., a sentence or a document) comes from the meanings of its constituents and the rules used to combine them. Specifically, Bespalov et al.[60] initialize the word embeddings by latent semantic analysis and further represent each document as the linear weighted of ngram vectors for sentiment classification. Glorot et al.[61] use Stacked Denoising Autoencoder in an unsupervised fashion based on reconstruction, and apply it for domain adaptation. Autoencoder is a kind of neural network that is optimized by reconstructing the input itself.

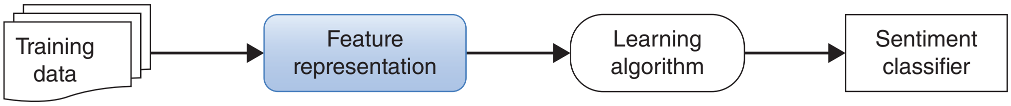

FIGURE 2 | Supervised framework for sentiment classification of sentences/documents.

Denoising Autoencoding randomly masks the values of inputs and tries to reconstruct the noisy inputs. Socher et al.[15] introduce a family of recursive deep neural models including Recursive Autoencoder (RAE),[62] Matrix–Vector Recursive Neural Network (MV-RNN),[63] and Recursive Neural Tensor Network (RNTN)[15] to learn the composition of variable-length phrases based on the representation of its children. Specifically, RAE first learns the structure of a sentence with greedy unsupervised reconstruction, and further conduct compositionality over the learned tree structure. In RAE, each word is encoded as a vector and the calculator is matrix multiplication plus nonlinear hyperbolic tangent function. In MV-RNN, each word is also associated with a matrix representation, and the tree structure is obtained from an external parser (like Stanford parser). In RNTN, they use neural tensor network as the compositional functions to better capture the interactions between elements. An example of RNN is given in Figure 3(b). RNN is one of the successful approaches in NLP community and extended with global feedbackward,[64] feature weight tuning,[65] deep recursive layer,[66] adaptive composition functions,[67] combined with Combinatory Categorial Grammar,[68] and used for opinion relation detection.[69] In particular, Hermann et al.[68] present Combinatory Categorial Autoencoders to learn the compositionality of sentence, which marries the Combinatory Categorial Grammar with Recursive Autoencoder. Dong et al.[67] develop adaptive RNN that uses more than one composition functions and adaptively select them depending on the input vectors. Li[65] extend RNN by tuning feature weight to control how much one specific unit contributes to the higher-level representation.

Another powerful neural network for semantic composition is convolutional neural network (CNN).[70] A brief illustration is given in Figure 3(a). For example, Kalchbrenner et al.[16] develop dynamic CNNs (DCNN) and introduce the dynamic k-max pooling for learning the sentence representation. Kim[71] and Johnson and Zhang[72] also investigate CNNs, and achieve several state-of-the-art performances on some benchmark datasets for sentiment classification. In Johnson and Zhang's[72] model, the word order is effectively incorporated.

Sequential model like recurrent neural network and long short-term memory (LSTM) are also verified as strong approaches for semantic composition. Such method process a sentence in a sequential way from the beginning of a sentence to the end, one word at a time. Neural networks with shared parameters are used for calculation. The basic computational unit could be a simple matrix multiplication or a complex LSTM unit. Li et al.[73] compare the effectiveness of RNN and recurrent neural network on five NLP tasks including sentiment classification. Some recent studies[74,75] investigate tree-structured LSTM for semantic composition.

Beyond sentence level compositionality, a few studies move eyes on document level semantic composition.[76,77] Two recent studies exploit the semantic relationship between sentences for document level sentiment analysis. Tang et al.[78] compose sentence representations to document representation in a sequential way, without using external discourse parser. Bhatia et al.,[79] however, use RST discourse parser and integrate the parsed results with RNN for document level sentiment analysis.

OPINION EXTRACTION

Given a piece of text (e.g., a sentence or a document), fine-grained opinion extraction targets at finding the

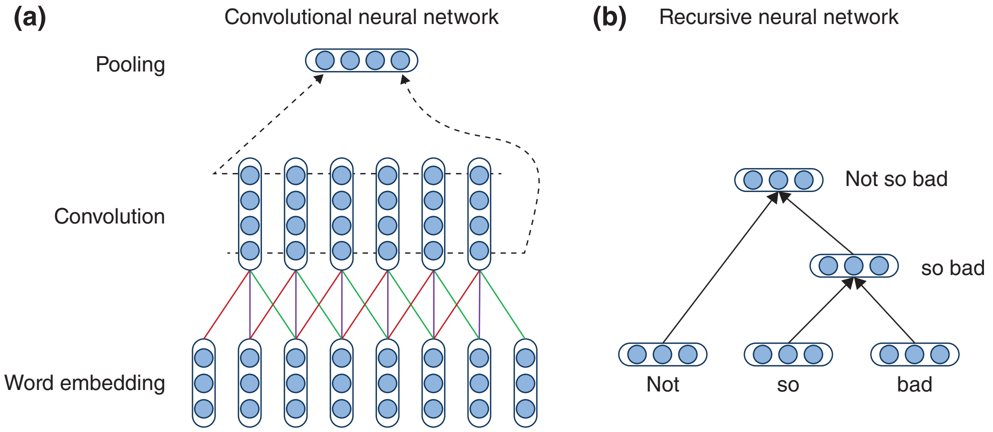

FIGURE 3 | A brief illustration about convolutional neural network and recursive neural network.

elements of a sentiment quintuple including the opinion holder, the entity/aspect, the sentiment of an affective expression, and so on.

This task is typically regarded as a sequential labeling problem.[80,81] For example, the sentence '*the screen of apple is so amazing*' should be tagged with '*O B-Aspect O B-Entity O B-Sentiment I-Sentiment*,' where B indicates the beginning of an expression, I means the tokens inside an expression, and O stands for tokens outside any predefined class. To address this problem, a lot of studies use CRFs or semi-CRF with manually designed discrete features such as word features, phrase features, and syntactic features.[12,80] As an alternative, Irsoy and Cardie[17] use neural network without using hand-crafted features. An illustration of the approach is given in Figure 4. The method proceeds a word at a time from the beginning to the end of a sentence. For each index, the input is word embedding, which will be fed to one or more hidden layers to get more abstractive and discriminative representations of data. Each hidden unit is computed based on its history and the current input. The method can be naturally extended to a bi-directional way, where the 'history' of a hidden unit comes from its surrounding contexts without constraining to the preceding words. The last hidden layer of a position is regarded as the representation of the corresponding word, and is used for classifying the tag label of the word. This method outperforms strong CRF baselines on a benchmark opinion extraction dataset.[17] Paulus et al.[64] investigate an analogous tree structured RNN for fine grained sentiment analysis task.

For fine-grained sentiment analysis, deep learning (neural network) approaches also achieve some promising results recently. For example, Vo and Zhang[82] show that rich features including word embedding perform well on target-dependent sentiment classification. Zhang et al.[83] use word embeddings and integrate them into neural CRF for open-domain sentiment analysis. Liu et al.[84] conduct empirical studies for fine-grained sequential labeling task, and show that LSTM recurrent neural network performs better than feature-based CRF on benchmark datasets.

BUILDING SENTIMENT LEXICON

A sentiment lexicon is a list of words and phrases such as 'excellent,' 'awful,' and 'coooolll,' each of which is assigned with a positive or negative score reflecting its sentiment polarity and strength. Sentiment lexicon is a fundamental component for sentiment analysis, which can be built manually, through thesaurus-based method and corpus-based method. As the manual method is time-consuming, it is mainly combined with automatic methods as the final check.[2] We describe thesaurus-based and corpus-based method.

Thesaurus-based method mainly utilizes semantic relationships (e.g., synonyms, antonyms, hypernyms, etc.) between tokens from an external thesaurus like WordNet. Under this direction, majority of existing studies regard word as basic unit,[6] yet some researchers target at the synset[85] in WordNet. Kim and Hovy[86] use synonym and antonym relations from WordNet to build sentiment lexicon. The hypothesis is that the synonyms of a positive word should have a positive polarity, and vice versa for antonym relation. They manually label a small size of adjective and verb words as sentiment seeds, and then apply a bootstrapping method to expand the seed list. Esuli and Sebastiani[87] take the gloss information of a word in WordNet into consideration. They manually label some sentiment seeds, and use a semi-supervised method to classify the polarity of a word based on its gloss in WordNet. Rao et al.[88] use label propagation algorithm to detect the polarity of words in the graph induced from WordNet. Feng

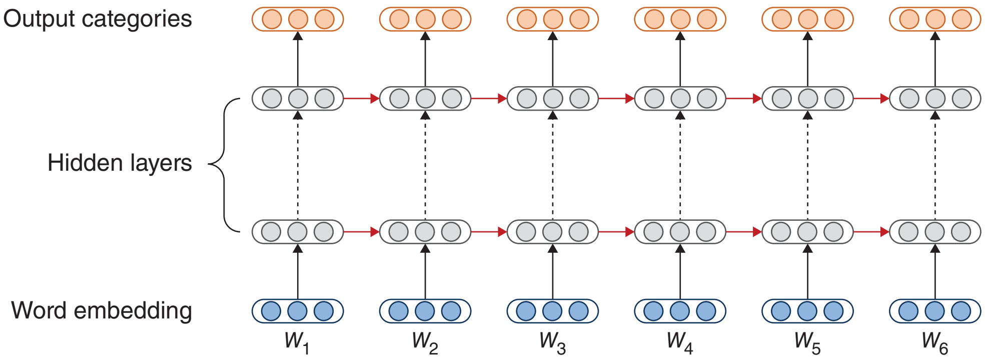

FIGURE 4 | An illustration of recurrent neural network for sequential labeling.

et al.[89] propose learning connotation lexicon, which lists words with connotative polarity, i.e., words with positive connotation (e.g., 'award' and 'promotion') and words with negative connotation (e.g., 'cancer' and 'war'). Feng et al.[90] go a step further and focus on the words that are objective on the surface like 'intelligence,' 'human,' 'cheesecake,' and so forth. Besides detecting the sentiment of each word, many researchers focus on identifying the polarity of a WordNet synset (also referred to as sense). Baccianella et al.[91] release the well-known SentiWordNet, in which each synset is associated with three numerical scores, describing how objective, positive, and negative the terms contained in the synset are. Each score in SentiWordNet is in range (0.0, 1.0) and the summation is 1.0. Esuli et al.[92] use pagerank to rank senses of WordNet in terms of how strongly they are positive or negative. Su et al.[93] use a semi-supervised framework based on mincut to recognize the subjectivity of a word sense in WordNet.

As the thesaurus like WordNet cannot well cover the growing colloquial sentiment expressions on the web, many researches employ corpus-based methods that induce a sentiment lexicon from text corpora. Hatzivassiloglou et al.[94] pioneer this field by extracting the polarity of adjective words. The idea is that words conjoined with 'and' favor to have the same polarity, and words conjoined by 'but' favor to have opposite polarity labels. They start with a list of sentiment seeds, and then identify more subjective adjectives with pre-defined conjunction patterns. Qiu et al.[95] propose a semi-supervised method dubbed double propagation for opinion word expansion and target extraction based on dependency relations between sentiment words and aspect words. An enhanced double propagation approach is given by Liu et al.,[96] and the method shows strong performances on multiple datasets. Velikovich et al.[97] represent words and phrases with their syntactic contexts within a window size from the web documents, and use graph propagation for label inference. Chen et al.[98] utilize the Urban Dictionary and extract the target-dependent sentiment expressions from Twitter. Mohammad et al.[11] use pointwise mutual information between each phrase and hashtag/emoticon seed words, such as '#good,' and '#bad,' ':),' and ':(.' Severyn and Moschitti[99] build sentiment lexicons using scores from an SVM model learned on a distant supervised corpora.

Tang et al.[18] cast sentiment lexicon learning as a phrase-level classification task and use deep learning strategy. The method consists of two part: (1) an embedding learning algorithm to effectively learn the continuous representation of phrases, which are used as features for word-level sentiment classification, (2) a seed expansion algorithm that expands a small list of sentiment seeds to collect training data for building the word-level classifier. The framework is illustrated in Figure 5. The generated sentiment lexicon is evaluated via applied as features for Twitter sentiment classification. The lexicon shows superior performances over traditional sentiment lexicon like MPQA[100] and large-scale sentiment lexicons like Sentiment140.[11]

SOME SUGGESTIONS ON IMPLEMENTATION

In this part, we briefly talk about how to implement deep learning (neural network) approaches for the beginners in this area. Let us take the hybrid prediction model in Figure 1 as an example. There are two options when we implement the model. The first choice is to calculate the gradient of each parameter in terms of the loss function by hand, and use that to update the value of each parameter. However, it is not scalable because we need to calculate the gradients again even if we just slightly modify the neural architecture. In order to pay more attentions to developing powerful neural architecture rather than gradient calculating, we use another choice with layer-wise implementation. For example, we implement lookup table, linear layer, tanh layer, softmax layer, and so on, and we link them together to build up the final model. Once the basic neural layers are completed, we do not need to care about them anymore and we can test variations of neural architecture easily. The implementations by us can be found at http://ir.hit.edu.cn/~dytang. One could also use GPU supported toolkits including Theano (deeplearning.net/software/theano/), Torch (torch.ch), and Caffe (caffe.berkeleyvision.org/).

CONCLUSION

In this study, we introduce successful deep learning approaches for several sentiment analysis tasks involving word embedding learning, sentiment classification, opinion extraction, and sentiment lexicon learning.

We lay out the future challenges of deep learning for aforementioned sentiment analysis tasks in this part. As deep learning is an emerging topic over machine learning and artificial intelligence in both research and industry community and is currently evolving very quickly, so the challenges might soon be outdated.

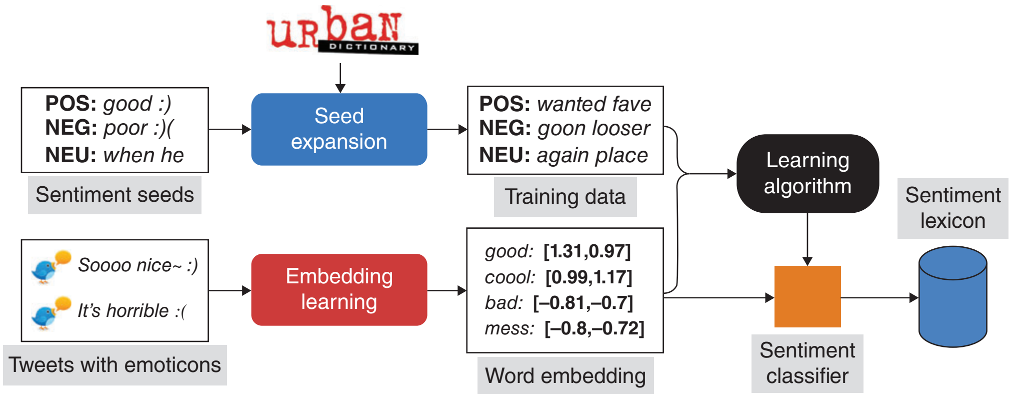

FIGURE 5 | A classification approach leveraging word embedding for building sentiment lexicon.

For learning better word representations for sentiment analysis tasks, Tang et al.[18,46] incorporate sentence level sentiment signals as supervisions. In this line of research, different levels of sentiment signals may be also investigated such as external lexical level sentiment information. In addition, it has been show that the approached introduced by Tang et al.[18,46] do not surpass context-based embeddings for document level classification task,[101,102] which remains a potential challenge about how to use coarse- and fine-grained supervisions. Another interesting direction is how to interpret the meaning of a word embedding because the dense representations are uninterpretable. A recent study given by Faruqui et al.[41] showed that transforming word embeddings into sparse vectors could yield promising results.

For semantic compositionality, most of existing studies focus on sentence level. Toward this goal, a potential challenge is how to learn sentence structure[103–105] together with the composed semantic representation as structure prediction[106] is a big challenge in both natural language processing and machine learning communities. Some studies also claim that the internal connections in CNN form an automatic structure of natural language. Another direction is how to interpret the effects of neural networks with visualization. Li et al.[107] provide several methods to visualize how neural models like LSTM are able to compose meanings in sentiment analysis task.

For building sentiment lexicon, the deep learning approach given by Tang et al.[18] cannot infer the sentiment polarity of the phrases not covered by the existing vocabulary. How to generate new sentiment words/phrases from new corpus is a remained as a challenge. A self-studied life-long framework might be a practicable solution.

ACKNOWLEDGMENTS

We gratefully acknowledge the helpful discussions with Yaming Sun. We thank the editor and anonymous reviewers for their helpful comments and feedbacks. This work was supported by the National High Technology Development 863 Program of China (No. 2015AA015407) and National Natural Science Foundation of China (Nos 61133012 and 61273321). Duyu Tang is supported by Baidu Fellowship and IBM Ph.D. Fellowship.

FURTHER READING

Li J, Li R, Hovy E. Recursive deep models for discourse parsing. In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2014, 2061–2069.

REFERENCES

1. Pang B, Lee L. Opinion mining and sentiment analysis. *Found Trends Inf Retr* 2008, 2:1–135.
2. Liu B. Sentiment analysis and opinion mining. *Synth Lect Human Lang Technol* 2012, 5:1–167.