Received November 5, 2018, accepted December 25, 2018, date of publication January 7, 2019, date of current version January 29, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2891019

MOCA: Multi-Objective, Collaborative, and Attentive Sentiment Analysis

JIA-DONG ZHANG, (Member, IEEE), AND CHI-YIN CHOW, (Senior Member, IEEE)
Department of Computer Science, City University of Hong Kong, Hong Kong

Corresponding author: Chi-Yin Chow (chiychow@cityu.edu.hk)

This work was supported in part by the National Natural Science Foundation of China under Grant 61772445, in part by the Innovation and Technology Fund (HKSAR) under Grant UIM/334 (9440187), and in part by the City University of Hong Kong under Grant 7004684, Grant 9678132, and Grant 9680221.

ABSTRACT Document-level sentiment analysis aims to predict the overall ratings of texts (e.g., reviews) written by users for items. Most current works model this problem as a supervised learning task, i.e., classification or regression. Recent studies argue that user preferences and item characteristics also have significant influences on ratings that are modeled by learning user-item-specific text embeddings based on neural networks. However, these studies only use the explicit influence observed in the texts and fail to model the implicit influence that cannot be observed in the texts. To this end, in this paper, we propose a multi-objective, collaborative, and attentive framework called MOCA for document-level sentiment analysis. Our MOCA has three important characteristics: 1) attentive model for explicit influence; MOCA applies a bidirectional recurrent neural network with attention mechanism to learn user-item-specific text embeddings for exploiting the explicit influence the of users and items; 2) collaborative model for implicit influence; MOCA devises a new neural collaborative filtering model based on multilayer perceptron to capture the implicit influence that is implied in the highly personalized interactions between the users and items; and 3) multi-objective optimization; MOCA models this problem as both classification and regression tasks and simultaneously optimizes the two objectives to reinforce one another. The experimental results show that our MOCA significantly outperforms other state-of-the-art techniques on three real-world datasets collected from IMDB and Yelp.

INDEX TERMS Document-level sentiment analysis, semantic learning, attentive model, neural collaborative filtering, multi-objective optimization.

I. INTRODUCTION

Sentiment analysis, also called opinion mining, aims to analyze people's sentiments, opinions, evaluations, attitudes and emotions towards entities according to their generated texts [1], [2]. With the rapid growth of e-commerce sites and social networks, people are used to writing texts (e.g., reviews, posts, comments or tips) to share their opinions on products, services, events, or topics. The sentiments expressed in the texts are potentially useful for downstream applications, e.g., recommender systems, financial services, and political elections. As such, sentiment analysis attracts increasing attention of researchers and becomes one of the most active research areas in natural language processing.

Sentiment analysis can be investigated at various levels of granularity, namely, document-level, sentence-level, and aspect-level. In this work, we study the problem of document-level sentiment analysis, the goal of which is to predict the overall rating of a text (e.g., a review) written by a user for an item. Most existing studies apply supervised machine learning algorithms to model this problem as a classification or regression task by extracting features from texts and treating ratings as labels [3]. Early works often use bag-of-words vectors as text features [4], [5] or with additional sentiment lexicons [6]–[8]. Recent works employ deep neural networks to learn distributed text embeddings (i.e., low-dimensional and continuous text representations) without any feature engineering and complete competitive performance [9]–[11].

These works only utilize *the text information* but ignore the important influence of *the user and item information*, i.e., the user who writes the text and the item which is evaluated in the text. **In reality, the user and item information significantly influences the rating of the user giving to the item.** In general, the influence can be divided into two types based on whether it can be observed in natural languages. (1) **Explicit influence.** This type of influence can be observed in *the user-item-specific words of texts*. For example, users often express the same opinion with user-specific words that depend on their preferences, habits, and experiences; each item has a set of item-specific evaluating words that rely on the item's characteristics. (2) **Implicit influence.** This type of influence cannot be observed in textual words, but is implied in *the interactions between users and items*. For instance, a lenient user often gives a higher rating than a critical user even if they post the same review for an item. Similarly, some items may receive higher ratings than others even though they have the same reviews.

There are also a few studies that consider the explicit influence for sentiment analysis. For example, Tang *et al.* [12] utilize user information to catch user-specific word embeddings and then extend the work to incorporate both user and item information to capture user-item-specific word embeddings via convolutional neural networks (CNNs) [13]. The recent works [14]–[16] achieve better performance by applying recurrent neural networks (RNNs) with the attention mechanism [17] to weigh the user-item-specific words to generate the text embedding. However, **the current studies have at least two major limitations**: (1) **Unable to model implicit influence.** Although these studies [12]–[16] intensively model the explicit influence by learning user-item-specific word or text embeddings, they are unable to model the implicit influence which is not reflected in the texts. Actually, the implicit influence is implied in the highly personalized interaction behaviors of users on items and has been extensively investigated in the research area of recommender systems. (2) **Single objective optimization.** The current research works [12]–[16] model document-level sentiment analysis as one single task, i.e., the classification or regression task. They may suffer from over-fitting training data and trapping in local optimal solutions due to optimizing the single objective. A better way is to develop a multi-task learning framework to optimize multiply objectives and enable these objectives to complement each other.

To alleviate the two above-mentioned limitations, this paper proposes a *M*ulti-*O*bjective, *C*ollaborative and *A*ttentive framework (*MOCA*) for document-level sentiment analysis. **MOCA contains three key characteristics**: (1) **Attentive model for explicit influence.** In line with the current state-of-the-art studies, MOCA applies a bidirectional RNN for learning embeddings and hidden states of each word in a text and exploits the attention mechanism for choosing important words for the user and item associated with the text, so as to capture the user-item-specific text embedding. (2) **Collaborative model for implicit influence.** MOCA models the interactions between users and items by collaborative filtering techniques which are widely used in recommender systems. In particular, MOCA devises a new neural collaborative filtering (NCF) model based on multilayer perceptron, in order to catch the highly personalized interactions of users on items. (3) **Multi-objective optimization.** MOCA considers the document-level sentiment analysis as both classification and regression tasks to mutually reinforce each other. It optimizes the two objectives at the same time by minimizing the sum of classification and regression losses.

The main contributions of this study are listed below:

- To the best of our knowledge, this is the first study to differentiate and exploit both explicit and implicit influences of users and items on ratings for sentiment analysis.
- We devise a new NCF model to learn the implicit influence, i.e., the interactions between users and items. This is also the first study to explore NCF techniques for sentiment analysis.
- We fuse both classification and regression tasks into one unified framework and optimize two objectives simultaneously to improve the performance of rating prediction.
- We conduct extensive experiments to evaluate the performance of MOCA using three review datasets collected from IMDB and Yelp. Experimental results show that MOCA achieves significantly superior performance compared to other state-of-the-art techniques.

II. RELATED WORK

This section briefly reviews the advanced techniques for document-level sentiment analysis, including *traditional methods*, *deep neural methods*, and *methods using user and item information*.

A. TRADITIONAL METHODS

The traditional methods apply supervised machine learning algorithms by extracting features from texts and treating ratings as labels. They concentrate on designing effective textual features, since the performance of a rating predictor heavily relies on the choice of features of texts. For example, the work [4] applies bag-of-words vectors as text features for support vector machines, while other studies extract bag-of-opinions features for ridge regression [5] or naive Bayes classifier [18], where an opinion consists of a root word, modifier words, and negation words. Further, some methods apply sentiment lexicons to extract bag-of-sentiments features [6]–[8], while other methods cluster words into topics and represent texts as topic features [19]. However, all these methods essentially employ a bag-of-words model to represent textual features and hence ignore word orders in texts. As a result, they fail to fully extract semantic information from texts and their improvement on sentiment analysis is very limited even though they use sentiment lexicons or topics.

B. DEEP NEURAL METHODS

Deep neural networks are utilized for sentiment analysis due to its great ability of text embedding learning without any feature engineering. For instance, the work [9] uses stacked denoising autoencoder in sentiment classification for the first time. The research works [10], [13], [20] employ CNNs to learn text embeddings by taking word orders into account, in which CNNs first embed each word into a vector, then apply convolution operation over word embedding sequences with a given window size to capture semantic information of a word in the context, and finally take max or average pooling operation on semantic information to get semantic text embeddings. The studies [11], [14]–[16] catch text embeddings based on RNNs which learn word meanings depending on contextual words and are inherently suitable for learning semantics from sequential texts. Moreover, all these studies [11], [14]–[16] exploit the RNN with long short-term memory (LSTM) [21] that is developed to enhance the ability of primitive RNNs to model long sequences, since primitive RNNs often suffer from the gradients vanishing and exploding problem in long sequences. In particular, the studies [11], [14], [16] build two-level hierarchical LSTMs to model texts as embeddings, in which the word-level LSTM learns sentence embeddings over word embedding sequences while the sentence-level LSTM generates text embeddings from sentence embedding sequences. In addition, Socher *et al.* [22], Li *et al.* [23], Bhatia *et al.* [24], and Tai textitet al. [25] present recursive neural networks to learn sentence embeddings. However, these recursive neural networks depend on expansive parsing trees of sentences and are not developed for text embeddings, so they are not suitable for document-level sentiment analysis.

C. METHODS USING USER AND ITEM INFORMATION

Since ratings are influenced by the preferences of users and characteristics of items, this information is also used in sentiment analysis. Diao *et al.* [26] and Li *et al.* [27] perform topic modeling on users and items. More sophisticatedly, the works [12], [13] model user-item-specific word embeddings in CNNs by modifying original word vectors with user and item information, in which each user (or item) has a continuous vector representation for capturing user-sentiment (or item-sentiment) consistencies and a continuous matrix representation for capturing user-text (or item-text) consistencies. Another work [20] catches review embeddings by CNNs over word sequences, learns the representations of a user or item by RNNs over the temporal order of review embeddings of the user or item, and concatenates the user and item representations with review embeddings as features for training a support vector machine. Recently, a few studies leverage the attention mechanism [17] to adaptively choose important words from texts for users and/or items. For example, the research [15] applies deep memory networks to learn the embedding of a text by taking the weighted sum of other text embeddings of the same user or item. Both studies [14], [16] exploit hierarchical LSTMs to generate two-level embeddings for sentences and texts, in which the user and item information is utilized to derive the importance weight of each word in a sentence (or the importance weight of each sentence in a text) in order to better capture semantic meanings in the sentence (or text). The difference is that the study [14] models user attentions and item attentions at the same hierarchical LSTMs whereas the study [16] models them separately.

Nonetheless, these studies [12]–[16] only utilize the explicit influence observed in the texts and cannot model the implicit influence implied in the interactions between users and items. Actually, the implicit influence is widely used to predict the preferences of users on new items without any texts of the users for these new items in recommender systems [28], [29]. Although the historical text information has also been applied in recommender systems [30], it is often employed to learn user and/or item representations in order to recommend new items for users rather than directly capture text embeddings for sentiment analysis. It is not desirable to simply adopt these recommendation methods [28]–[30] for sentiment analysis, because the two research problems are essentially different. The research problem on sentiment analysis aims to predict the overall rating of a given text, whereas the research problem on recommendation intends to suggest new items for a user and there are not any available texts for the user on the new items, because the user has never interacted with these new items. Note that it is not meaningful to recommend old items for users. Further, existing studies [12]–[16] model sentiment analysis as one single task and may suffer from over-fitting training data. This paper proposes a new framework to alleviate these two limitations.

III. THE PROPOSED MOCA

In this section, we define the research problem, followed by the architecture of MOCA. Then we detail the four components of MOCA: semantic learning, attentive model, neural collaborative filtering, and multi-objective optimization.

A. PROBLEM STATEMENT

For the sake of clarity, we generally use lowercases for elements in sets, and calligraphic uppercases for sets, bold lowercases for vectors, and bold uppercases for matrices.

Definition 1 (Word and Text Embeddings): A text y is a sequence of l words $(x_1, \dots, x_l)$ coming from a fixed vocabulary $\mathcal{X}$, where each word is embedded as a vector $\mathbf{x} \in \mathbb{R}^{\mathrm{D}_1}$, and the text is embedded as a vector $\mathbf{y} \in \mathbb{R}^{2\mathrm{D}_2}$. D_1 and $2\mathrm{D}_2$ are the dimension of the embedding vector. For presentation, we use $2\mathrm{D}_2$ as dimension due to the bidirectional RNN.

Definition 2 (User and Item Representations): User $u \in \mathcal{U}$ and item $v \in \mathcal{V}$ are represented as a vector $\mathbf{u} \in \mathbb{R}^{\mathrm{D}_3}$ and $\mathbf{v} \in \mathbb{R}^{\mathrm{D}_3}$ respectively, where D_3 is the dimension of the representation vector.

Definition 3 (Explicit Influence of User and Item Information): Explicit influence can be observed in the user-item-specific words of texts. To model explicit influence for sentiment analysis, we concentrate on predicting the rating of a given text y with associated user u and item v information.

Definition 4 (Implicit Influence of User and Item Information): Implicit influence cannot be observed in texts but is implied in the interactions between users and items. To model implicit influence for sentiment analysis, we focus on

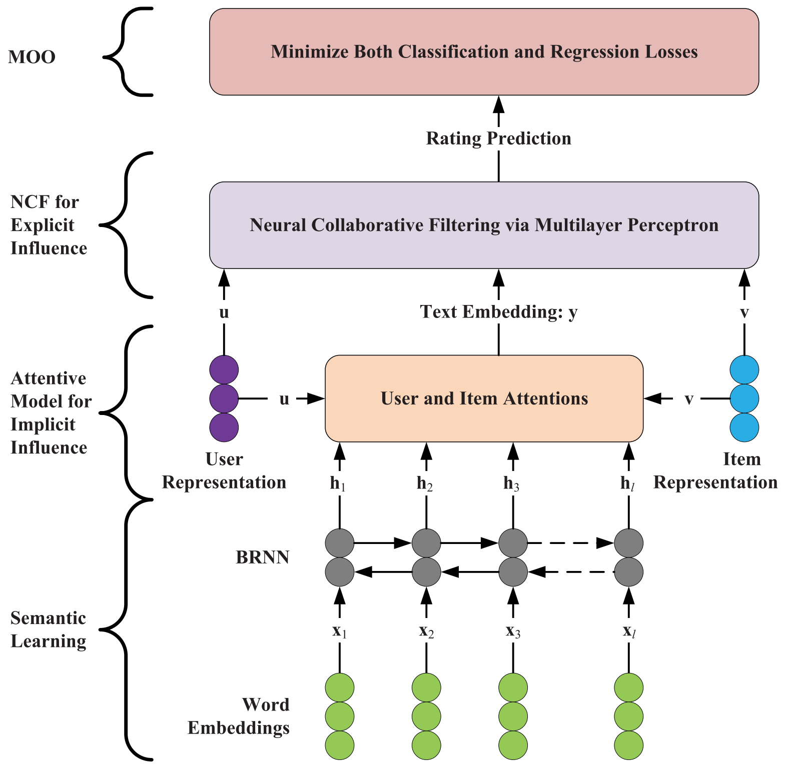
MOO
Minimize Both Classification and Regression Losses
Rating Prediction
NCF for Explicit Influence
Neural Collaborative Filtering via Multilayer Perceptron
u
Text Embedding: y
v
Attentive Model for Implicit Influence
u
User and Item Attentions
v
User Representation
h1
h2
h3
hl
Item Representation
BRNN
Semantic Learning
x1
x2
x3
xl
Word Embeddings

FIGURE 1. The architecture of MOCA with four components: semantic learning, attentive model, neural collaborative filtering (NCF), and multi-objective optimization (MOO).

predicting the rating of a user u on a new item v without the text of the user for the new item.

Definition 5 (Research Problem): Given a text y consisting of words $(x_1, \ldots, x_l)$ written by a user u for an item v, the goal is to predict the overall rating of text y, by taking full advantage of semantic meanings of text y (i.e., explicit influence) and personalized interactions between user u and item v (i.e., implicit influence).

B. ARCHITECTURE

The architecture of the proposed MOCA is depicted in FIGURE 1. It consists of four components: semantic learning, attentive model, neural collaborative filtering (NCF), and multi-objective optimization (MOO) from bottom to top.

(1) **Semantic learning.** This component aims to extract semantic meanings from texts by two steps: (i) It converts each word into an embedding vector $\mathbf{x}_i$ in a latent semantic space and enables to compare two different words, e.g., calculating their semantic similarity or distance on sentiment words, which benefits for the follow-up analysis on texts. (ii) A bidirectional recurrent neural network (BRNN) [31] is adopted to capture the semantic meanings of a text consisting of the word sequence $(\mathbf{x}_1, \ldots, \mathbf{x}_l)$ in both forward and backward directions. The BRNN generates the hidden state $\mathbf{h}_i$ for each word which represents the real meaning of the word in the current context.

(2) **Attentive model.** This component learns a text's embedding $\mathbf{y}$ from the hidden states $(\mathbf{h}_1, \ldots, \mathbf{h}_l)$ of all words in the text. A simple and common method views all words equally important and takes the average of their states as the text embedding $\mathbf{y}$. However, a better method is to calculate the weighted sum of these states based on the attention mechanism [17]. Specifically, this component exploits the attentions of user $\mathbf{u}$ and item $\mathbf{v}$ to discover important words for them, because different users have different preferences on words for expressing their opinions, while different items also have different suitable words for evaluating them.

(3) **Neural collaborative filtering (NCF).** This component intends to predict the rating of a text written by user u for item v. Besides the text embedding $\mathbf{y}$ from the attentive model, the component also takes into account user u's preferences and item v's characteristics, which may not be reflected in the text and greatly influence the rating. This component devises a new NCF method to model the user representation $\mathbf{u}$, item representation $\mathbf{v}$, and their interactions, since users show personalized preferences on items.

(4) **Multi-objective optimization (MOO).** This component models the document-level sentiment analysis as both classification and regression tasks in one unified framework, by minimizing the classification and regression losses on the rating prediction at the same time. The multi-objective optimization enables the two tasks to reinforce one another.

C. SEMANTIC LEARNING FROM TEXT INFORMATION

To capture the semantic meanings of words in a text, we adopt the BRNN [31] that learns the real meaning of each word

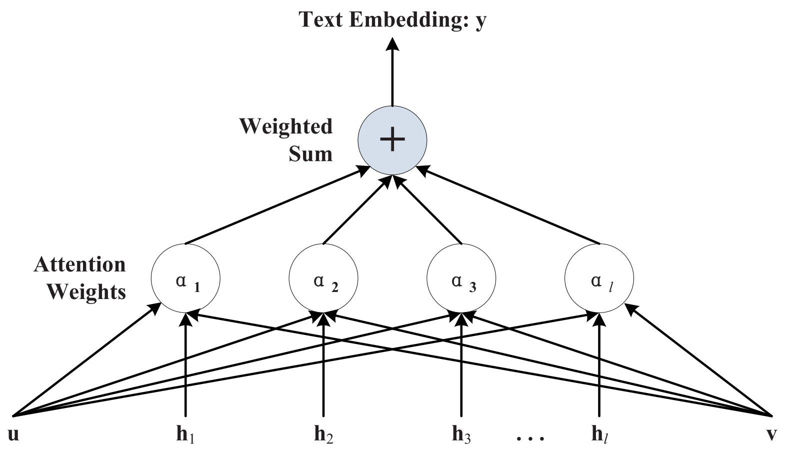

FIGURE 2. User-item based attentive model: user representation **u**, item representation **v**, and LSTM hidden state **h**.

in the text and generates the hidden state for the word by taking its context into account, i.e., looking neighbor words forwardly and backwardly. Moreover, the BRNN is mounted with the long short-term memory (LSTM) [21] that utilizes forget gate, input gate, output gate and memory cell to control the passing of information at each position along the sequence and thus the long-range dependencies in the sequence can be caught.

BRNN symmetrically models the semantics of words in both the forward and backward directions. Following, we take the forward direction as the example. The forward LSTM processes the word embedding sequence $(\mathbf{x}_1, \dots, \mathbf{x}_l)$ from $\mathbf{x}_1$ to $\mathbf{x}_l$ sequentially. At each position, given current word embedding $\mathbf{x}_t \in \mathbb{R}^{D_1}$, forward cell state $\overrightarrow{\mathbf{c}}_{t-1} \in \mathbb{R}^{D_2}$ and hidden state $\overrightarrow{\mathbf{h}}_{t-1} \in \mathbb{R}^{D_2}$, the current hidden state $\overrightarrow{\mathbf{h}}_t \in \mathbb{R}^{D_2}$ is generated by

$$\overrightarrow{\mathbf{f}}_t = \sigma\left(\overrightarrow{\mathbf{W}}_f[\mathbf{x}_t, \overrightarrow{\mathbf{h}}_{t-1}] + \overrightarrow{\mathbf{b}}_f\right), \tag{1}$$

$$\overrightarrow{\mathbf{i}}_t = \sigma\left(\overrightarrow{\mathbf{W}}_i[\mathbf{x}_t, \overrightarrow{\mathbf{h}}_{t-1}] + \overrightarrow{\mathbf{b}}_i\right), \tag{2}$$

$$\overrightarrow{\mathbf{o}}_t = \sigma\left(\overrightarrow{\mathbf{W}}_o[\mathbf{x}_t, \overrightarrow{\mathbf{h}}_{t-1}] + \overrightarrow{\mathbf{b}}_o\right), \tag{3}$$

$$\overrightarrow{\hat{\mathbf{c}}}_t = \tanh\left(\overrightarrow{\mathbf{W}}_c[\mathbf{x}_t, \overrightarrow{\mathbf{h}}_{t-1}] + \overrightarrow{\mathbf{b}}_c\right), \tag{4}$$

$$\overrightarrow{\mathbf{c}}_t = \overrightarrow{\mathbf{f}}_t \odot \overrightarrow{\mathbf{c}}_{t-1} + \overrightarrow{\mathbf{i}}_t \odot \overrightarrow{\hat{\mathbf{c}}}_t, \tag{5}$$

$$\overrightarrow{\mathbf{h}}_t = \overrightarrow{\mathbf{o}}_t \odot \tanh\left(\overrightarrow{\mathbf{c}}_t\right), \tag{6}$$

where σ is the sigmoid function, $\odot$ stands for element-wise multiplication, all $\mathbf{W} \in \mathbb{R}^{D_2 \times (D_1 + D_2)}$ and $\mathbf{b} \in \mathbb{R}^{D_2}$ are the model parameters, and $[\mathbf{x}_t, \overrightarrow{\mathbf{h}}_{t-1}]$ is the concatenation of two vectors $\mathbf{x}_t$ and $\overrightarrow{\mathbf{h}}_{t-1}$.

Symmetrically, the backward LSTM processes the word embedding sequence $(\mathbf{x}_1, \dots, \mathbf{x}_l)$ from $\mathbf{x}_l$ to $\mathbf{x}_1$. Given current word embedding $\mathbf{x}_t$, backward cell state $\overleftarrow{\mathbf{c}}_{t+1}$ and hidden state $\overleftarrow{\mathbf{h}}_{t+1}$, the current hidden state $\overleftarrow{\mathbf{h}}_t$ can be concisely written as

$$\overleftarrow{\mathbf{h}}_t = \mathrm{LSTM}_{\mathrm{backward}}(\mathbf{x}_t, \overleftarrow{\mathbf{c}}_{t+1}, \overleftarrow{\mathbf{h}}_{t+1}). \tag{7}$$

Finally, the forward and backward states of each word are concatenated into:

$$\mathbf{h}_t = [\overrightarrow{\mathbf{h}}_t, \overleftarrow{\mathbf{h}}_t]. \tag{8}$$

The concatenated state $\mathbf{h}_t \in \mathbb{R}^{2D_2}$ represents the latent and real meaning of the t-th word in the text and will be used to learn the text embedding based on the attentive model.

D. ATTENTIVE MODEL FOR EXPLICIT INFLUENCE

Traditional methods usually take the average of all word hidden states $(\mathbf{h}_1, \dots, \mathbf{h}_l)$ as the text embedding $\mathbf{y} \in \mathbb{R}^{2D_2}$. However, these methods ignore the effect of the intrinsic meanings of words and the contexts of texts, i.e., the associated user and item. Therefore, this study applies an attentive model to learn the semantics of a text by considering the attentions of users and items over words, as depicted in FIGURE 2. The attentive model defines the text embedding as the weighted sum of these states, given by

$$\mathbf{y} = \sum\nolimits_{t=1}^{l} \alpha_t \mathbf{h}_t, \tag{9}$$

where weight α_t measures the importance of the t-th word for the current user $\mathbf{u} \in \mathbb{R}^{D_3}$ and item $\mathbf{v} \in \mathbb{R}^{D_3}$, and is normalized by the softmax function:

$$\alpha_t = \mathrm{softmax}(e(\mathbf{u}, \mathbf{h}_t, \mathbf{v})), \tag{10}$$

where $e(\cdot)$ is a score function that aims to discover the preferred words for the user $\mathbf{u}$ who writes the text and the suitable words that evaluate the item $\mathbf{v}$, rather than view all words equally important. It is usually defined by a two-layer neural network:

$$e(\mathbf{u}, \mathbf{h}_t, \mathbf{v}) = \mathbf{w}_a^{\mathsf{T}} \tanh\left(\mathbf{W}_a[\mathbf{u}, \mathbf{h}_t, \mathbf{v}] + \mathbf{b}_a\right), \tag{11}$$

in which $\mathbf{W}_a \in \mathbb{R}^{D_4 \times (2D_2 + 2D_3)}$, $\mathbf{b}_a \in \mathbb{R}^{D_4}$, and $\mathbf{w}_a \in \mathbb{R}^{D_4}$ are the attentive model parameters, $^{\mathsf{T}}$ denotes transpose, and D_4 is the dimension of the attentive model.

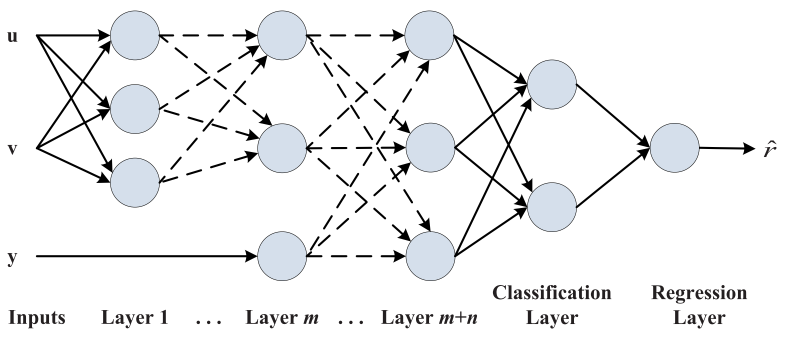

FIGURE 3. Neural collaborative filtering: user representation **u**, item representation **v**, and text embedding **y**.

E. NCF FOR IMPLICIT INFLUENCE

The attentive model focuses on utilizing the explicit influence, i.e., learning the user-item-specific text embedding $\mathbf{y}$ for sentiment analysis. However, not all user preferences and item characteristics can be observed in the text. The implicit interactions between users and items are highly personalized and may not be captured by the attentive model. Thus, we devise a new NCF which integrates user representation $\mathbf{u}$, item representation $\mathbf{v}$ and text embedding $\mathbf{y}$ into a multilayer perceptron for rating prediction, as depicted in FIGURE 3.

First, the NCF takes as inputs the concatenation of $\mathbf{u}$ and $\mathbf{v}$ at the first layer, and outputs $\mathbf{z}_m$ at the m-th layer:

$$\begin{aligned}\mathbf{z}_1 &= \phi(\mathbf{W}_1[\mathbf{u}, \mathbf{v}] + \mathbf{b}_1),\\ \mathbf{z}_2 &= \phi(\mathbf{W}_2\mathbf{z}_1 + \mathbf{b}_2),\\ &\cdots\\ \mathbf{z}_m &= \phi(\mathbf{W}_m\mathbf{z}_{m-1} + \mathbf{b}_m), \end{aligned} \tag{12}$$

where the output of one layer serves as the input of the next layer, and each layer is able to discover certain implicit structures on the interactions between user $\mathbf{u}$ and item $\mathbf{v}$. Then, the NCF combines $\mathbf{z}_m$ with $\mathbf{y}$ to get the output $\mathbf{z}_{m+n}$:

$$\begin{aligned}\mathbf{z}_{m+1} &= \phi(\mathbf{W}_{m+1}[\mathbf{z}_m, \mathbf{y}] + \mathbf{b}_{m+1}),\\ \mathbf{z}_{m+2} &= \phi(\mathbf{W}_{m+2}\mathbf{z}_{m+1} + \mathbf{b}_{m+2}),\\ &\cdots\\ \mathbf{z}_{m+n} &= \phi(\mathbf{W}_{m+n}\mathbf{z}_{m+n-1} + \mathbf{b}_{m+n}), \end{aligned} \tag{13}$$

where $\mathbf{z}_{m+n}$ incorporates both explicit and implicit influences. Further, the output $\mathbf{z}_{m+n}$ is fed into the classification layer to derive the score $\mathbf{s}$ for each class (e.g., five classes from one to five stars):

$$\mathbf{s} = \phi(\mathbf{W}_{m+n+1}\mathbf{z}_{m+n} + \mathbf{b}_{m+n+1}). \tag{14}$$

Finally, the score $\mathbf{s}$ is fed into the regression layer to predict a real-valued rating $\hat{r}$:

$$\hat{r} = \phi(\mathbf{w}_{m+n+2}^{\mathsf{T}}\mathbf{s} + b_{m+n+2}). \tag{15}$$

The activation function ϕ can be set to sigmoid σ, hyperbolic tangent (tanh), or rectifier (relu). All $\mathbf{W}$, $\mathbf{w}$ and $\mathbf{b}$ are the NCF model parameters, and their dimension depends on the number of neurons of the layers with which they connect. It is a good practice to design the network structure to follow a tower pattern, where each successive layer has a smaller number of neurons than its previous layer. In this paper, we set the number of neurons to $\lceil 2^{-j+2}\mathrm{D}_3 \rceil$ $(1 \leq j \leq m)$ and $\lceil (\lceil 2^{-m+2}\mathrm{D}_3 \rceil + 2\mathrm{D}_2)\, 2^{-j+m+1} \rceil$ $(m+1 \leq j \leq m+n)$ for the j-th layer, in which $2\mathrm{D}_2$ is the dimension of text embedding $\mathbf{y}$, and D_3 is the dimension of user representation $\mathbf{u}$ and item representation $\mathbf{v}$, as presented in Definitions 1 and 2. The classification layer has the same number of neurons as classes, while the regression contains only one neuron for predicting a scalar value.

F. MULTI-OBJECTIVE OPTIMIZATION

In this study, the document-level sentiment analysis is modeled as both classification and regression tasks. Thus, it is required to minimize the losses of classification and regression at the same time.

For classification, the score $\mathbf{s}$ in Equation (14) is normalized into the probability distribution $\hat{\mathbf{r}}$ through the softmax function:

$$\hat{\mathbf{r}} = \mathrm{softmax}(\mathbf{s}). \tag{16}$$

Then the cross entropy function is applied to calculate the classification loss, given by

$$L_1 = -\sum\nolimits_{j=1}^{\mathrm{C}} \mathbf{r}(j)\log(\hat{\mathbf{r}}(j)), \tag{17}$$

where C is the number of classes, (j) denotes the j-th element, and $\mathbf{r}$ is the ground-truth rating class distribution. Note that $\mathbf{r}$ is an one-hot binary vector representation of the scalar rating r, by setting the element for the true class to 1 and all other elements to 0. For regression, given the ground-truth rating r and the predicted rating $\hat{r}$ in Equation (15), the square error is considered as the regression loss:

$$L_2 = (r - \hat{r})^2. \tag{18}$$

It is worth emphasizing that Equations (14) and (15) compute the loss for only one rating. Given a set of ratings $\mathcal{R}$, we minimize the weighted sum of classification and regres-

TABLE 1. Statistics of the three datasets.

	IMDB	Yelp 2013	Yelp 2014
No. of classes ($\mathbb{C}$)	10	5	5
No. of docs	84,919	78,966	231,163
No. of users ($\lvert\mathcal{U}\rvert$)	1,310	1,631	4,818
No. of items ($\lvert\mathcal{V}\rvert$)	1,635	1,633	4,194
No. of docs per user	64.82	48.42	47.97
No. of docs per item	51.94	48.36	55.11
No. of words per doc	394.6	189.3	196.9
Vocabulary size ($\lvert\mathcal{X}\rvert$)	61,579	33,626	54,043

sion losses, given by

$$\min_{\Omega} \frac{1}{|\mathcal{R}|} \sum_{r \in \mathcal{R}} (\lambda L_1 + (1-\lambda)L_2), \tag{19}$$

where Ω denotes all model parameters learned via the stochastic gradient descent algorithm [32]; $\lambda \in [0, 1]$ controls the weight of two losses and enables both classification and regression tasks to reinforce each other.

IV. EXPERIMENTS

In this section, we conduct extensive experiments on three real-world datasets to evaluate the effectiveness of our MOCA and analyze experimental results by comparing MOCA with the state-ofthe-art models.

A. EXPERIMENTAL SETTINGS

1) DATASETS

We use three real-world datasets which are collected from IMDB and Yelp Dataset Challenge in 2013 and 2014 [13]. TABLE 1 summarizes the statistics of the three datasets. Each dataset is already split into three parts: 80% for training, 10% for validation, and the remaining 10% for testing. The three datasets are the benchmark for evaluating the task of document-level sentiment analysis with the effect of users and items [14]–[16], [20].

2) EVALUATED MODELS

We compare our **MOCA** with the following models for document-level sentiment analysis.

1) **Trigram** trains a support vector machine (SVM) with unigrams, bigrams and trigrams as features.
2) **TextFeature** trains an SVM with word and character n-grams, sentiment lexicon, negation features, and cluster features [8].
3) **UPF** extracts user-leniency and product features [33] and then concatenates them with the features in **Trigram** and **TextFeature**.
4) **AvgWordvec + SVM** averages word embeddings in a text to obtain the text embedding for training an SVM.
5) **SSWE + SVM** learns sentiment-specific word embeddings and uses max/min/average pooling to generate text embeddings for training an SVM [34].
6) **Paragraph Vector** implements the distributed memory model of paragraph vectors for document-level sentiment classification [35].
7) **RNTN + Recurrent** represents sentences with recursive neural tensor networks [22], generates text embeddings with recurrent neural networks, and averages hidden vectors as features for sentiment classification.
8) **JMARS** is a recommendation algorithm that uses the information of users and aspects with collaborative filtering and topic modeling [26].
9) **UPNN** utilizes user and product information to capture user-item-specific word embeddings for sentiment classification via CNNs [13].
10) **CNN + RNN + SVM** learns user and item representations by RNNs over the temporal order of text embeddings from CNNs and concatenates the representations with text embeddings as features for training an SVM [20].
11) **UPDMN** applies deep memory networks to learn the embedding of a text by taking the weighted sum of other text embeddings of the same user or item and feeds the text embedding into the softmax layer for sentiment classification [15].
12) **NSC + UPA** is a neural sentiment classification with user production attention in the hierarchical word-level and sentence-level RNNs [14].
13) **HUAPA** models user attentions and item attentions separately in the hierarchical word-level and sentence-level RNNs [16].

We report the results of these methods in [13]–[16] and [20] since we use exactly the same datasets.

3) PERFORMANCE METRICS

We adopt standard *Accuracy* to measure the overall sentiment classification performance and *RMSE* to measure the divergences between ground truth r_j and prediction $\hat{r}_j$:

$$Accuracy = T/N, \tag{20}$$

$$RMSE = \sqrt{\frac{\sum_{j=1}^{N}(r_j - \hat{r}_j)^2}{N}}, \tag{21}$$

where N is the number of reviews in the testing set and T is the number of reviews with correct predicted sentiment labels. Note that our MOCA evaluates *Accuracy* on the classification task and *RMSE* on the regression task.

4) TRAINING SETTINGS

We tokenize texts into words with NLTK[1] and implement our MOCA based on Google's TensorFlow[2] which is an open source software library for deep learning. TABLE 2 lists the hyperparameter settings of **MOCA** in the experiments. We pre-train the 256-dimensional word embeddings ($\mathbb{D}_1 = 256$) on each dataset with CBOW [36]. The word embeddings

[1]https://www.nltk.org
[2]https://www.tensorflow.org

TABLE 2. **Hyperparameter settings.**

Hyperparameter	Value
Dimension of word embeddings	$D_1 = 256$
Dimension of hidden states in LSTM	$D_2 = 128$
Dimension of text embeddings	$2D_2 = 256$
Dimension of representations of users and items	$D_3 = 256$
Dimension of attentive model	$D_4 = 256$
Layer sizes in NCF	$m = 3$ and $n = 3$
Activation function in NCF	$\phi = relu$
Loss weight by default	$\lambda = 0.7$
Learning rate of stochastic gradient descent	0.02
Batch size of stochastic gradient descent	64

TABLE 3. **The results of document-level sentiment classification on the three datasets. Accuracy (Acc., higher is better) and RMSE (lower is better) are the evaluation metrics. The best results are in bold.**

Models	IMDB		Yelp 2013		Yelp 2014	
	Acc.	RMSE	Acc.	RMSE	Acc.	RMSE
Models without user and item information						
Trigram	0.399	1.783	0.569	0.814	0.577	0.804
TextFeature	0.402	1.793	0.556	0.845	0.572	0.800
AvgWordvec+SVM	0.304	1.985	0.526	0.898	0.530	0.893
SSWE+SVM	0.312	1.973	0.549	0.849	0.557	0.851
Paragraph Vector	0.341	1.814	0.554	0.832	0.564	0.802
RNTN+Recurrent	0.400	1.764	0.574	0.804	0.582	0.821
Models with user and item information						
Trigram+UPF	0.404	1.764	0.570	0.803	0.576	0.789
TextFeature+UPF	0.402	1.774	0.561	0.822	0.579	0.791
JMARS	N/A	1.773	N/A	0.985	N/A	0.999
UPNN	0.435	1.602	0.596	0.784	0.608	0.764
CNN+RNN+SVM	0.488	1.451	0.639	0.694	0.639	0.688
UPDMN	0.465	1.351	0.639	0.662	0.613	0.720
NSC+UPA	0.533	1.281	0.650	0.692	0.667	0.654
HUAPA	0.550	1.185	0.683	0.628	0.686	0.626
MOCA	**0.569**	**1.060**	**0.706**	**0.596**	**0.718**	**0.578**

are fixed when training other model parameters. The dimension of hidden states in LSTM is set to 128 ($D_2 = 128$), so a bidirectional LSTM gives 256-dimensional text embeddings. The dimension of user and item representations is set to 256 ($D_3 = 256$) and the representations are randomly initialized with the uniform distribution $(-0.01, 0.01)$. The dimension of the attentive model is also set to 256 ($D_4 = 256$). In NCF, the layer sizes m and n are set to 3 and the activation function ϕ is set to rectifier (relu). For multi-objective optimization, the loss weight λ is set to 0.7 by default. We use stochastic gradient descent algorithm [32] with exponential moving average to update parameters and set initial learning rate to 0.02 and batch size to 64 when training. Finally, we select the best model parameters based on the performance on the validation set and evaluate the parameters on the testing set. We do not use any regularization to improve the performance of our model.

B. EXPERIMENTAL RESULTS

1) MODEL COMPARISONS

TABLE 3 depicts the results of all evaluated models which are divided into two groups: the models only using the text information, and the models incorporating both the text information and the user and item information. Obviously, the second group performs better than the first group. For example, Trigram + UPF and TextFeature + UPF obtain at least 1.1% improvement on RMSE across the three datasets, compared to Trigram and TextFeatur, respectively. These comparisons indicate that the user and item information is helpful for document-level sentiment analysis. The results of the first group have been discussed in detail from [13]–[16] and [20]. Thus, we concentrate on analyzing the results of the models in the second group.

We can conclude the following four important findings: (1) Our MOCA achieves the best performance in terms of accuracy and RMSE across all the three datasets by multi-task learning with both explicit and implicit influences of users and items. As opposed to the second best performance given by HUAPA, MOCA boosts accuracy from 3% to 5% and reduces RMSE from 5% to 10%. These results demonstrate the effectiveness of MOCA for modeling the user and item information, i.e., learning user-item-specific text embeddings and capturing personalized user-item interactions. (2) UPDMN, NSC + UPA and HUAPA are competitive, because they all apply the attention mechanism to catch user-item-specific text embeddings for sentiment classification. However, they fail to model the personalized interactions of users on items and cannot use the implicit influence. (3) Both UPNN and CNN + RNN + SVM result in worse accuracy and RMSE in comparison with UPDMN, NSC + UPA and HUAPA, since they do not apply the attention mechanism to discover the user-item-specific words. (4) JMARS is evaluated in terms of RMSE because it outputs real-valued ratings. JMARS reports much higher RMSE, although it adopts collaborative filtering for the user and item information. JMARS is a recommendation algorithm and applies conventional collaborative filtering rather than NCF developed in this paper. This proves that it is not desirable to simply borrow recommendation models for sentiment analysis.

TABLE 4. **Effect of model components on Accuracy (Acc., higher is better) and RMSE (lower is better). AM denotes Attentive Model for explicit influence and NCF denotes Neural Collaborative Filtering for implicit influence.**

Models	IMDB		Yelp 2013		Yelp 2014	
	Acc.	RMSE	Acc.	RMSE	Acc.	RMSE
BRNN	0.526	1.296	0.669	0.638	0.673	0.633
BRNN+AM	0.539	1.223	0.682	0.615	0.684	0.614
BRNN+NCF	0.546	1.180	0.672	0.627	0.693	0.604
BRNN+AM+NCF	0.558	1.091	0.690	0.611	0.702	0.585
MOCA	0.569	1.060	0.706	0.596	0.718	0.578

2) ANALYSIS ON COMPONENTS

TABLE 4 shows the effect of the components of MOCA on accuracy and RMSE over three datasets. BRNN is the baseline of semantic learning by taking the average of all word hidden states as the text embedding and feeding it into the classification layer for accuracy or the regression layer for

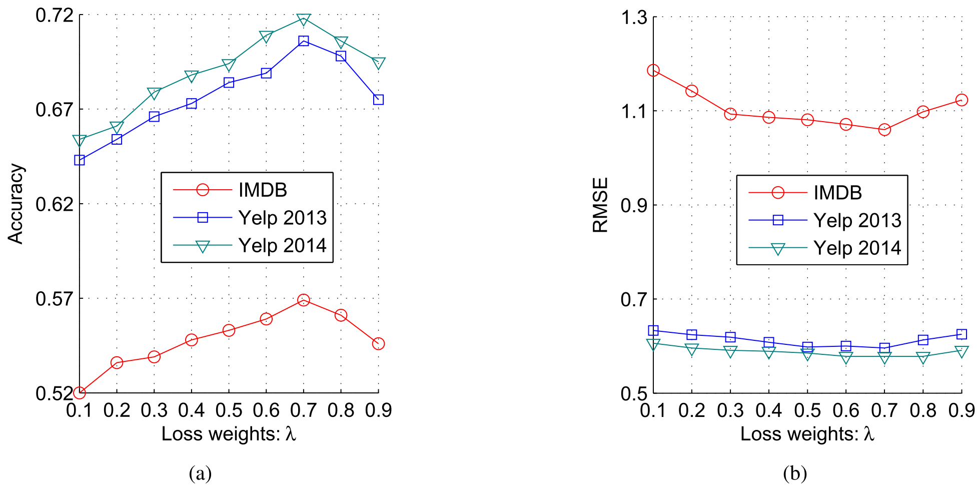

FIGURE 4. **Effect of loss weights on multi-objective optimization. Higher Accuracy or lower RMSE is better. (a) Accuracy (b) RMSE.**

RMSE. By integrating BRNN with attentive model (AM) for the explicit influence of user-item-specific words or neural collaborative filtering (NCF) for the implicit influence of user interactions on items, the performance is improved significantly. The improvement of NCF is larger than that of AM on the IMDB and Yelp 2014 datasets, but reverse on the Yelp 2013 dataset. Moreover, by fusing AM and NCF together, BRNN + AM + NCF reduces RMSE further. Finally, with multi-objective optimization, MOCA achieves the best results on the three datasets, which shows that all components are necessary and play important roles.

3) ANALYSIS ON LOSS WEIGHTS

FIGURE 4 depicts the effect of varying loss weights λ in Equation (19) from 0.1 to 0.9 on multi-objective optimization. As λ for the classification task increases (i.e., the loss weight for the regression task decreases), both accuracy and RMSE become better at first and then get worse on all the datasets. This trend of accuracy and RMSE is the same, which verifies that the classification and regression tasks form mutually beneficial cooperation rather than excluding one another in our proposed framework. In other words, the multi-objective optimization is a cooperative game in which both objectives obtain optimal rewards at the equilibrium state, instead of the zero-sum game where one objective gets revenue at the cost of the other one.

V. CONCLUSION

In this paper, we proposed a new framework called MOCA for document-level sentiment analysis. First, MOCA applies BRNN with the attentive model to catch user-item-specific text embeddings for utilizing the explicit influence of users and items. Then, MOCA develops a new NCF model to exploit the user's and item's implicit influence that is implied in the personalized interactions between them. Finally, MOCA models the sentiment analysis as two tasks and optimizes both classification and regression objectives simultaneously to reinforce one another. Experimental results show that MOCA achieves the best performance compared to other state-of-the-art methods on the three real-world datasets for IMDB and Yelp.

REFERENCES

[1] B. Pang and L. Lee, "Opinion mining and sentiment analysis," *Found. Trends Inf. Retr.*, vol. 2, nos. 1–2, pp. 1–135, 2008.
[2] B. Liu, *Sentiment Analysis and Opinion Mining*. San Rafael, CA, USA: Morgan & Claypool Publishers, 2012.
[3] B. Pang and L. Lee, "Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales," in *Proc. ACL*, 2005, pp. 115–124.
[4] G. Paltoglou and M. Thelwall, "A study of information retrieval weighting schemes for sentiment analysis," in *Proc. ACL*, 2010, pp. 1386–1395.
[5] L. Qu, G. Ifrim, and G. Weikum, "The bag-of-opinions method for review rating prediction from sparse text patterns," in *Proc. COLING*, 2010, pp. 913–921.
[6] X. Ding, B. Liu, and P. S. Yu, "A holistic lexicon-based approach to opinion mining," in *ACM WSDM*, 2008, pp. 231–240.
[7] M. Taboada, J. Brooke, M. Tofiloski, K. Voll, and M. Stede, "Lexicon-based methods for sentiment analysis," *Comput. Linguistics*, vol. 37, no. 2, pp. 267–307, 2011.
[8] S. Kiritchenko, X. Zhu, and S. M. Mohammad, "Sentiment analysis of short informal texts," *J. Artif. Intell. Res.*, vol. 50, no. 1, pp. 723–762, 2014.
[9] X. Glorot, A. Bordes, and Y. Bengio, "Domain adaptation for large-scale sentiment classification: A deep learning approach," in *Proc. ICML*, 2011, pp. 513–520.
[10] R. Johnson and T. Zhang, "Effective use of word order for text categorization with convolutional neural networks," in *Proc. HLT-NAACL*, 2015, pp. 103–112.
[11] Q. Hu, J. Zhou, Q. Chen, and L. He, "SNNN: Promoting word sentiment and negation in neural sentiment classification," in *Proc. AAAI*, 2018, pp. 3255–3262.
[12] D. Tang, B. Qin, T. Liu, and Y. Yang, "User modeling with neural network for review rating prediction," in *Proc. IJCAI*, 2015, pp. 1340–1346.
[13] D. Tang, B. Qin, and T. Liu, "Learning semantic representations of users and products for document level sentiment classification," in *Proc. ACL*, 2015, pp. 1014–1023.
[14] H. Chen, M. Sun, C. Tu, Y. Lin, and Z. Liu, "Neural sentiment classification with user and product attention," in *Proc. EMNLP*, 2016, pp. 1650–1659.
[15] Z.-Y. Dou, "Capturing user and product information for document level sentiment analysis with deep memory network," in *Proc. EMNLP*, 2017, pp. 521–526.

[16] Z. Wu, X.-Y. Dai, C. Yin, S. Huang, and J. Chen, "Improving review representations with user attention and product attention for sentiment classification," in *Proc. AAAI*, 2018, pp. 5989–5996.
[17] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," in *Proc. ICLR*, 2015.
[18] J.-D. Zhang, C.-Y. Chow, and Y. Zheng, "ORec: An opinion-based point-of-interest recommendation framework," in *Proc. ACM CIKM*, 2015, pp. 1641–1650.
[19] G. Ganu, Y. Kakodkar, and A. Marian, "Improving the quality of predictions using textual information in online user reviews," *Inf. Syst.*, vol. 38, no. 1, pp. 1–15, 2013.
[20] T. Chen, R. Xu, Y. He, Y. Xia, and X. Wang, "Learning user and product distributed representations using a sequence model for sentiment analysis," *IEEE Comput. Intell. Mag.*, vol. 11, no. 3, pp. 34–44, Aug. 2016.
[21] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.
[22] R. Socher *et al.*, "Recursive deep models for semantic compositionality over a sentiment treebank," in *Proc. EMNLP*, 2013, pp. 1631–1642.
[23] J. Li, R. Li, and E. Hovy, "Recursive deep models for discourse parsing," in *Proc. EMNLP*, 2014, pp. 2061–2069.
[24] P. Bhatia, Y. Ji, and J. Eisenstein, "Better document-level sentiment analysis from RST discourse parsing," in *Proc. EMNLP*, 2015, pp. 2212–2218.
[25] K. S. Tai, R. Socher, and C. D. Manning, "Improved semantic representations from tree-structured long short-term memory networks," in *Proc. ACL*, 2015, pp. 1556–1566.
[26] Q. Diao, M. Qiu, C.-Y. Wu, A. J. Smola, J. Jiang, and C. Wang, "Jointly modeling aspects, ratings and sentiments for movie recommendation (JMARS)," in *Proc. ACM SIGKDD*, 2014, pp. 193–202.
[27] F. Li, S. Wang, S. Liu, and M. Zhang, "SUIT: A supervised user-item based topic model for sentiment analysis," in *Proc. AAAI*, 2014, pp. 1636–1642.
[28] J.-D. Zhang, C.-Y. Chow, and J. Xu, "Enabling kernel-based attribute-aware matrix factorization for rating prediction," *IEEE Trans. Knowl. Data Eng.*, vol. 29, no. 4, pp. 798–812, Apr. 2017.
[29] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, and T.-S. Chua, "Neural collaborative filtering," in *Proc. WWW*, 2017, pp. 173–182. [Online]. Available: https://dl.acm.org/citation.cfm?id=3052569
[30] J.-D. Zhang and C.-Y. Chow, "SEMA: Deeply learning semantic meanings and temporal dynamics for recommendations," *IEEE Access*, vol. 6, pp. 54106–54116, 2018.
[31] M. Schuster and K. K. Paliwal, "Bidirectional recurrent neural networks," *IEEE Trans. Signal Process.*, vol. 45, no. 11, pp. 2673–2681, Nov. 1997.
[32] A. Y. LeCun, L. Bottou, G. B. Orr, and K.-R. Müller, "Efficient backprop," in *Neural Networks: Tricks of the Trade*. Berlin, Germany: Springer, 1998, pp. 9–50.
[33] W. Gao, N. Yoshinaga, N. Kaji, and M. Kitsuregawa, "Modeling user leniency and product popularity for sentiment classification," in *Proc. IJCNLP*, 2013, pp. 1107–1111.
[34] D. Tang, F. Wei, N. Yang, M. Zhou, T. Liu, and B. Qin, "Learning sentiment-specific word embedding for twitter sentiment classification," in *Proc. ACL*, 2014, pp. 1555–1565.
[35] Q. Le and T. Mikolov, "Distributed representations of sentences and documents," in *Proc. ICML*, 2014, pp. 1188–1196.
[36] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Proc. NIPS*, 2013, pp. 3111–3119.

JIA-DONG ZHANG received the M.Sc. degree from Yunnan University, China, in 2009, and the Ph.D. degree from the City University of Hong Kong, in 2015, where he is currently a Research Fellow with the Department of Computer Science. His research has been published in premier conferences, including the ACM SIGIR, CIKM, and SIGSPATIAL, in transactions, including the ACM TIST, IEEE TKDE, TDSC, TSC, and TITS, and in journals, including the IEEE ACCESS, *Pattern Recognition*, and *Information Sciences*. His research interests include deep learning, data mining, recommender systems, and location-based services.

CHI-YIN CHOW received the M.S. and Ph.D. degrees from the University of Minnesota, Twin Cities, USA, in 2008 and 2010, respectively. He is currently an Associate Professor with the Department of Computer Science, City University of Hong Kong. His research interests include big data analytics, data management, GIS, mobile computing, location-based services, and data privacy. He was a recipient of the VLDB 10-year Award, in 2016. He received best paper awards at ICA3PP 2015 and IEEE MDM 2009. He is a Co-Founder and a Co-Chair of the ACM SIGSPATIAL MobiGIS, from 2012 to 2016, and an Editor of the *ACM SIGSPATIAL Newsletter*.