IEEE Access
Multidisciplinary ⁝ Rapid Review ⁝ Open Access Journal

Received February 3, 2018, accepted March 1, 2018, date of publication March 12, 2018, date of current version April 23, 2018.

Digital Object Identifier 10.1109/ACCESS.2018.2813419

Semantic Clustering-Based Deep Hypergraph Model for Online Reviews Semantic Classification in Cyber-Physical-Social Systems

XU YUAN, MINGYANG SUN, ZHIKUI CHEN, (Member, IEEE), JING GAO, AND PENG LI
School of Software, Dalian University of Technology, Dalian 116620, China

Corresponding author: Zhikui Chen (zkchen@dlut.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 61672123 and Grant 61602083, in part by the Fundamental Research Funds for the Central Universities under Grant DUT2017TB02, in part by the Dalian University of Technology Fundamental Research Fund under Grant DUT15RC(3)100, in part by the Doctoral Scientific Research Foundation of Liaoning Province under Grant 20170520425, and in part by the Key Laboratory of Vibration and the Control of Aero-Propulsion System, Ministry of Education, Northeastern University, under Grant VCAME201705.

ABSTRACT Sentiment classification of online reviews is playing an increasingly important role for both consumers and businesses in cyber-physical-social systems. However, existing works ignore the semantic correlation among different reviews, causing the ineffectiveness for sentiment classification. In this paper, a word embedding clustering-based deep hypergraph model (ECDHG) is proposed for the sentiment analysis of online reviews. The ECDHG introduces external knowledge by employing the pre-training word embeddings to express reviews. Then, semantic units are detected under the supervision of semantic cliques discovered by an improved hierarchical fast clustering algorithm. Convolutional neural networks are connected to extract the high-order textual and semantic features of reviews. Finally, the hypergraph can be constructed based on high-order relations of samples for the sentiment classification of reviews. Experiments are performed on five-domain data sets including movie, book, DVD, kitchen, and electronic to assess the performance of the proposed model compared with other seven models. The results validate that our model outperforms the compared methods in classification accuracy.

INDEX TERMS CNNs, hypergraph, sentiment classification, online reviews, short text.

I. INTRODUCTION

With the recent rapid development of e-commerce, online word-of-mouth such as forms of community, microblogs and social networking site created by cyber-physical-social systems users have become an important information source for consumers to select goods or services and for businesses to improve the quality of commodities or services with big data analytics and processing [1]–[4]. Online review is a representative form of word-of-mouth and generally refers to the positive or negative viewpoints on goods or services posted by latent or actual consumers on websites such as e-commerce or third-party reviews. Online reviews are playing a remarkable role for consumers in online purchasing decisions. According to the survey by Jupiter Research, a leading international market research firm, more than 75% of consumers refer to online review information before online shopping. Especially in the environment of information overload today, valuable online reviews can help consumers make better decisions. As for businesses, valuable online reviews can help them improve the quality of services and products. However, it is a challenging issue to acquire the valuable online reviews by recognizing the review sentiment. Therefore, this paper focuses on the online reviews sentiment classification.

Conventional classification methods for online reviews generally expand texts with latent semantics, which is based on the Latent Dirichlet Allocation (LDA) or its variants. Phan *et al.* [5] proposed a novel framework for expanding online reviews by attaching the topic's names that are obtained by adopting the LDA model. Sahami and Heilman took advantage of web resource search by the text segment to enrich the text representations [6]. More recently, Yan *et al.* [7] proposed the Biterm Topic Model to model online reviews. These methods are based on the bag-of-words algorithm, however, they ignore the order and semantic information between words. Therefore, they cannot effectively acquire the fine-grained semantics for online reviews.

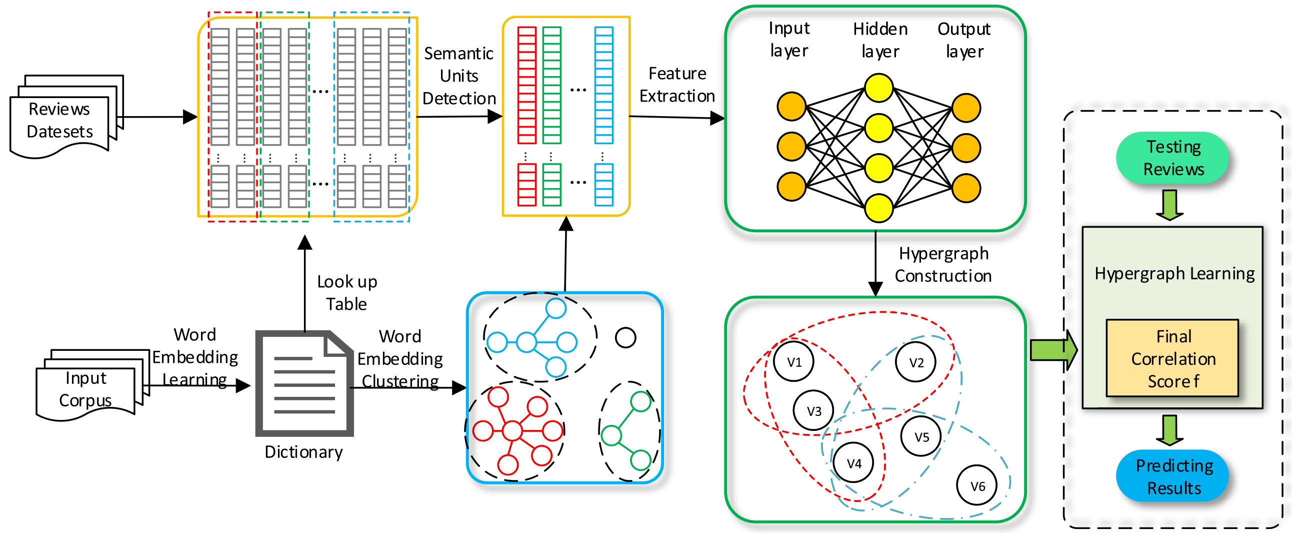


FIGURE 1. **The overview of a word embedding clustering based deep hypergraph model.**

For sentiment classification of online reviews, another representative method is the machine-learning method. Pang *et al.* [8] firstly used the Naïve Bayes (NB), Maximum Entropy (ME), and Support Vector Machines (SVM), for sentiment polarity classification. Furthermore, an important deep learning model, convolutional neural networks (CNN) [9]–[15], has been used for online reviews sentiment classification. However, the previous works cannot intuitively reflect the relevance between different reviews, since they are adopted softmax function at the last layer of CNN to calculate the probability distribution. Subsequently, Wang *et al.* [11] applied semantic cliques discovered by using fast clustering algorithm to detect semantic units. However, the clustering algorithm has lower efficiency and higher memory cost in processing large-scale word embedding data. And only the distances of latent semantic units with clustering center and the nearest word embedding in the cluster are calculated, without taking the meaning of central word in the semantic units into account, which causes the loss of original information. Some words alone are of little significance, which can be replaced by combinations with other words completely. Introducing expanded matrices lead to more cost of time and space.

In this paper, a deep hypergraph model based on the word embeddings clustering is proposed for sentiment polarity classification of online reviews. In this work, the proposed model will keep the composition of the texts as much as possible and acquire the semantic representations of each text. Particularly, an improved hierarchical clustering algorithm, which is an improvement on the fast clustering algorithm based on density peaks searching, is firstly used to discover the semantic cliques, which provide label information for detection of the latent semantic units, in embedding space. As shown in Fig. 1, the text can then be represented as a matrix composed by embedding representations of semantic units. After getting the projected matrices, the deep CNN is trained to capture the feature and acquire the feature vector of each text. Finally, the hypergraph model can be constructed based on feature vectors, where each text can be treated as a vertex and different polarity relations can be used to construct hyperedges based on the similarity among texts.

The key contributions of our paper are summarized as follows:

- A deep hypergraph model based on word embeddings clustering and CNN proposed, which can capture the high-level features and reflect the high-order relations among samples.
- We propose an improved task specific hierarchical clustering algorithm based on density peaks searching for semantic clustering of word embeddings.
- Semantic units are detected with considering the central words, which maximally preserves original information of reviews for improving sentiment classification accuracy.
- The reviews are represented by projected matrices, which are constructed by more meaningful semantic units than some single words. This can improve the expressing ability of semantic matrix for reviews

The rest of this paper is organized as follows. Some related works are introduced in section II. Section III describes the details of ECDHG, including word embedding clustering, latent semantic units detection, feature extraction and hypergraph learning. Section IV shows the experimental setup and results. Finally, this paper is concluded in Section V.

II. RELATED WORK

Sentiment analysis is playing an important role in many fields such as user reviews of product or service. Sentiment classification, as one of the most significant branches in sentiment analysis, has been studied widely. The earlier work of sentiment analysis utilized many pre-developed sentiment lexicons, such as WordNet-Affect, Subjectivity

Lexicon and SentiWordNet [16], for sentiment classification. The WordNet-Affect is a linguistic resource, which is developed by the selection and labeling of the synonym sets representing affective concepts. The subjectivity lexicon is constructed by a set of artificially selected subjectivity words and an online dictionary. The SentiWordNet is a lexicon for sentiment classification, which grades each synonym set in WordNet along three affective dimensions: positivity, negativity, and neutrality. Subsequently, Taboada *et al.* [17] proposed a method SO-CAL based on sentiment lexicon and sentiment extracted from text in combination with linguistic rules such as intensification and negation. The lexicon includes not only adjectives but also nouns, verbs and adverbs. Nevertheless, the lexicon based methods show low level of reliability as dictionaries are constructed automatically or hand-ranked by humans [18]. In addition, such methods [17], [19], [20] are to some extent limited by a satisfactory sentiment dictionary, which is difficult to obtain.

Three ensemble approaches i.e. bagging, boosting and random subspace on ten public datasets are adopted in [21] when using five learners including NB, ME, K Nearest Neighbors (KNN), SVM and Decision Tree (DT). Turney [22] propose using Point-wise Mutual Information (PMI) and Information Retrieval to measure the similarity between words. For example, ‘‘excellent’’ and ‘‘poor’’ are considered as positive and negative reference terms, so the sentiment orientation of samples can be obtained by calculating the difference of PMI using ‘‘excellent’’ and ‘‘poor’’ respectively. However, these studies only focus on textual representations and ignoring the semantic correlation among samples.

Recently, neural network-based methods have been employed to model language [23] and learn word embeddings [24]. Mikolov *et al.* [24] presented a continuous skip-gram model, which is an efficient approach to learn high-quality word embedding from large scale unstructured corpus. Furthermore, various composition-based methods for sentiment analysis of short texts have been proposed. Online reviews can be treated as a kind of short text. Zhang *et al.* [9] exploited character- to sentence-level information to perform sentiment analysis of short texts with deep CNN. Zhang *et al.* [10] trained a simple CNN with one layer of convolution for sentence-level classification tasks and then make a simple modification to allow for the use of both pre-trained and task-specific vectors with multiple channels. Furthermore, Wang *et al.* [11] trained CNN with both projected matrix and expanded matrices, which are consist of semantic units, as input in parallel for sentiment classification of short texts. More recently, Johnson and Zhang [25] proposed deep pyramid CNN architecture for text classification, which can obtain the best accuracy by increasing the depth of network without increasing too much computational cost. However, these previous works adopted softmax function at the last layer of CNN to calculate the probability distribution, which cannot intuitively reflect the relevance between texts.

III. DEEP HYPERGRAPH MODEL

In this section, a deep hypergraph sentiment classifier used for reviews modeling and classification is described in detail, as shown in Fig. 1. In this work, we introduce external knowledge by taking advantage of semantic units consisted of the pre-training word embedding representations and contextual information to improve classification performance for user reviews. After transforming the texts into matrix representations, a deep CNN model is used to extract each input feature. And then we construct hypergraph model where each review can be considered as a vertex for sentiment classification. There are different types of hyper-edges to connect the reviews with similar polarity.

A. WORD EMBEDDING CLUSTERING

In embedding space, the neighbors of one word are usually semantically related [24]. Therefore, we can discover the semantic cliques by using clustering methods. In this work, we adopt the fast clustering algorithm based on searching density peaks to achieve word embedding clustering. The clustering centers discovered by fast clustering algorithm are surrounded by neighbors with lower local density and have a relatively large distance from the points with a higher local density, which precisely satisfies the distribution property of word embeddings.

We will compute two values for data point i, local density ρ_i and some sort of distance δ_i to complete the clustering. Specially, we adopt cosine similarity to measure the distance, that is, the similarity between two points. Specifically, local density ρ_i of i is obtained by:

$$\rho_i = \sum_j \chi(d_{ij} - d_c) \tag{1}$$

where d_{ij} denotes the cosine similarity between data points i and j, d_c demotes the cutoff cosine similarity that effect is similar to cutoff distance, and

$$\chi(\cdot) = \begin{cases} 1, & \textit{if } d_{ij} < d_c \\ 0, & \textit{otherwise} \end{cases} \tag{2}$$

Thus, ρ_i is equal to the number of neighbors that closer than d_c to point i. And distance δ_i is obtained by:

$$\delta_i = \begin{cases} \min\limits_{j:\rho_j > \rho_i} (d_{ij}), & \textit{if } \rho_i < \rho_{\max} \\ \max\limits_j (d_{ij}), & \textit{otherwise} \end{cases} \tag{3}$$

According to Eq.3 when i has the maximum local density, δ_i is the distance between i and the point j with maximum distance from i. Otherwise, δ_i indicates the distance between i and the point with the smallest distance from i among all points with higher local density than i.

However, the clustering algorithm has low efficiency and high memory cost due to large-scale word embedding data. In order to solve these problems, we present an improved algorithm for clustering of word embeddings, hierarchical fast clustering algorithm based on density peaks searching (H-CFS). The improved algorithm first divides word embedding data into sub-datasets containing N samples, where N needs to be specified by the user. Then, clustering results of sub-datasets can be obtained by performing cluster process on each sub-dataset in parallel. Because the words in the same cluster are semantically close, a cluster can be represented by the clustering center as the sample for second clustering. As for outliers, it can be viewed as an independent clustering center. Thus, the semantic clustering results of word embeddings can be obtained by layer-by-layer clustering.

B. WORD EMBEDDING CLUSTERING

For a review text T, there are mainly two problem to extract the feature representation: one is that the length of T is variable; the other is that a phrase in T may be more meaningful than a single word contained in the phrase and these meaningful phrases could appear at any position of T. Thus, simply combining the embedding representations of each word of T may affect the validity and efficiency of all semantic representations. Therefore, the detection of the meaningful semantic units is helpful for improving the capacity of local feature of symptoms of the matrix representation of T.

For a review text $T = \{w_1, w_2, \ldots, w_n\}$, an ordered sequence of n words, the model first transforms these words into real-valued word vectors by looking up in pre-trained word vectors dictionary D. Thus, the matrix representation $TM \in \mathbb{R}^{n\times d}$ of the text is obtained, which can be performed using matrices product as follows: $TM = D \times index\,(T)$, where the dictionary $D \in \mathbb{R}^{v\times d}$ is initialized by *Word2Vec*, v is the size of vocabulary, d is the dimension of embedding, and $index\,(.)$ is the function that transform each word in T into real-valued vectors, which is corresponding to the vocabulary of the dictionary D. Then, the model detects semantic units with cosine similarity meeting the set threshold with the clustering center of the words contained in the semantic units and replaces the words with them, which is very effective in reducing the scale of input matrix.

Particularly, in order to get the representations of the semantic units, we use a window matrix $E_{win} \in \mathbb{R}^{s\times d}$, where $\boldsymbol{s}$ is variable, with all weights equal to one to perform the operation on matrix $\boldsymbol{TM}$ similar to the filter. The essence of the operation is similar to a one-dimensional convolution, which can be defined as:

$$[su_1, su_2, \ldots, su_m] = TM \cdot E_{win} \tag{4}$$

where su_i can be obtained by:

$$su_i = \sum_{j=1}^{s} TM_j^{win,i} = \sum \left(TM\left[i_w^{cur}\right], TM\left[i_w^{nei}\right], \ldots \right) \tag{5}$$

$TM_j^{win,i}$ is the j-th row from the sub-matrix $TM^{win,i}$, which is consist of current word with position i in $\boldsymbol{TM}$ and its neighbors in the range of window size, s is the width of the window matrix E_{win}. As shown in Eq.5, the i-th semantic unit $su_i \in \mathbb{R}^d$ with the same dimension as each word embedding is summation of the component from the columns in $TM^{win,i}$. However, the operation of detecting semantic units will not repeat the calculation of a word, that is, a word is contained only in a unique semantic unit.

In dense embedding space, semantically close words are likewise close in Euclidean or cosine distance. Since using cosine similarity to indicate the semantic similarity between words in *Word2Vec*, we compute cosine similarity to measure the similarity between semantic units and semantic cliques center for recognizing precise semantic units. A threshold τ is used as a constraint to fine-tune the detection of semantic units. The detection algorithm for semantic unit is shown in Algorithm 1. Firstly, the algorithm initializes the position of current word: index, that is, the current line of the text matrix. Then, the algorithm detects all the meaningful semantic units. For the word in current position, we calculate the cosine similarity between the clustering center of the word and the semantic unit constructed by the word and its neighbors in range of the window size s. The semantic unit with cosine similarity to the clustering center of current word being above the threshold τ is selected to replace the words, and update the current position. We set s to be 2, because the process is repeated until all meaningful semantic units are detected.

Algorithm 1 The Algorithm for Detecting Semantic Units and Replacing the Words With the Scale of Window Matrix $\boldsymbol{s}$ Equal to 2

Input: The reviews $R = \{r_1, r_2, \ldots, r_n\}$, Clustering Center CV, The threshold τ

Output: Simplified Text Matrix STM

```
Transform each review $r_i$ into projected matrix $TM_i$ based on dictionary $D$
for Each review $r_i$ do:
  while semantic units exist in $r_i$:
    Initialize current position index.
      while index < length ($TM_i$):
        Detect latent semantic units $su_m$ around index in
          range of $s$.
        if the cosine between $su_m$ and $CV$ meet the
          threshold $\tau$:
            Let $su_m$ append to $STM_i$.
        update index
          $TM_i \leftarrow STM_i$
       Let $TM_i$ append to $STM$.
end for
```

C. FEATURE EXTRACTION

After detection for all semantic units of an input review text, review text can be represented as matrix constituted by the embedding representations of these semantic units, which is used as the input to CNN model.

In this work, a convolutional layer is obtained by convolving a matrix of weights $F \in \mathbb{R}^{d\times t}$ with the matrix of activations at the layer below. We let the operations be one-dimensional convolutions on the input matrices. The one-dimensional convolution is an operation between a filter vector of weights $F \in \mathbb{R}^{t}$ with width t and the input text matrices with detection of semantic units $X = (x_1, x_2, \ldots, x_m)^T$, where $x_i \in \mathbb{R}^{d}$ associated with the i-th meaningful semantic unit in text. The one-dimension convolution is defined as taking the dot product of the filter F with t-gram in the text X and nonlinear transformation to obtain the feature map C consisted of c_j. The feature c_j is generated by:

$$c_j = f(x_{j-t+1:j} \cdot F + b) \tag{6}$$

where f is a non-linear activation function, and b is a bias term. The number of filter vectors and their width t are hyper-parameters of the network. As shown in Fig. 2. The resulting feature matrix C has dimensions $d \times (m + t - 1)$. The weights of the filter F learned in training could be regarded as a linguistic feature detector that learns to identify a specific class of n-grams, which has been shown useful for sentiment analysis [9].

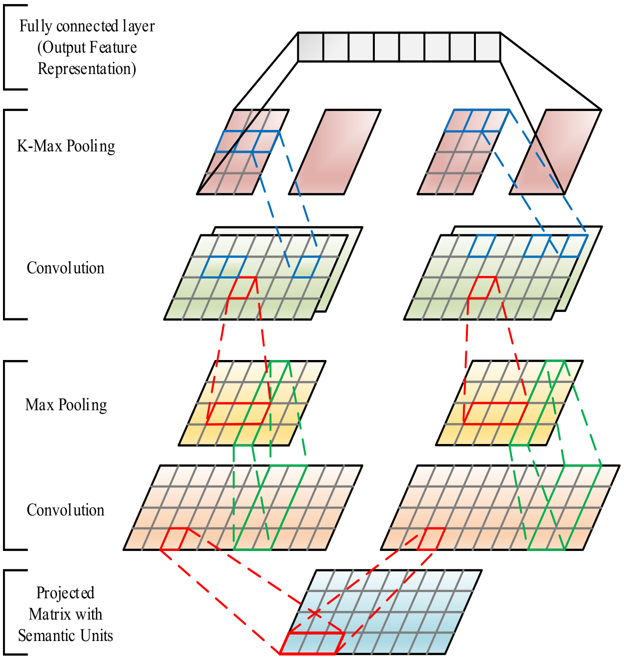


FIGURE 2. **Convolutional Neural Network for Feature Extraction.**

In this work, we construct a deep neural network for feature extraction with three convolutional layers to extract higher level features. After each convolution, the max-over-time pooling layer over the feature map is connected to capture the most useful local features for sentiment classification and reduce the size of input to the next convolutional layer, thereby reducing the complexity of the model. However, because of the variable size of the input texts, the size of output feature map is variable after two pooling operations. In order to capture the most meaningful global features with fixed size, and enable the output feature map meet the needs of hypergraph construction, a k-max pooling operation is used after the third convolution. This fixed sized global feature map can be then used to construct hypergraph model.

D. HYPERGRAPH LEARNING FOR CLASSIFICATION

We can obtain the feature representations of each input review text after the layer of feature extraction performed by operations of convolution and max-pooling. In this work, we construct a hypergraph model for sentiment classification, instead of softmax function used to get the probability distribution of output feature representations. Thus each review text can be represented as a vertex in hypergraph, and the similarity relationships among reviews can be treated as different hyperedges. Unlike the simple graph that an edge can only connect two vertices, the edge in a hypergraph can connect more than two vertices, known as hyperedge, each of which is assigned a weight.

In this paper, we represent a hypergraph as $G = (V, E, w)$, where V is a finite set of vertices, E is the hyperedge set and w is weight vector of the hyperedge. In the hypergraph, each hyperedge e_i is assigned a weight $w(e_i)$. Assume there are n reviews in the dataset, the hypergraph will contain n vertices. Let $V = \{v_1, v_2, \ldots, v_n\}$ and $E = \{e_1, e_2, \ldots, e_m\}$ represents n vertices and m hyperedges respectively. In our method, a hyperedge is generated by a centroid vertex and its k nearest neighbors, thus, a hyperedge can connect $k + 1$ vertices. The hypergraph can be denoted as an incidence matrix $H \in \mathbb{R}^{|V|\times|E|}$ with the entry as follows:

$$H(v, e) = \begin{cases} 1, & if\ v \in e \\ 0, & if\ v \notin e \end{cases} \tag{7}$$

The degree of each vertex based on H is defined as:

$$d(v) = \sum_{e\in E} \omega(e)H(v, e) \tag{8}$$

and the degree of a hyperedge is defined as:

$$\delta(e) = \sum_{v\in V} H(v, e) \tag{9}$$

where $d(v)$ is the weighted sum of a row and $\delta(v)$ is the sum of a column in H. D_v and D_e represent the diagonal matrices of the degrees for vertex and hyperedge, respectively. And W represents the diagonal matrix where the entries are the weights of hyperedges.

We measure the affinity for review i and review j by calculating the cosine similarity as follows:

$$S_{i,j} = \begin{cases} \exp(\frac{C(i,j)}{\overline{C}}), & if\ i \neq j \\ 0, & else \end{cases} \tag{10}$$

where C is the cosine similarity matrix computed on each two reviews, and $\overline{C}$ is the median value of matrix C. Subsequently, we can get the weight of each hyperedge by summing the similarity of the centroid vertex with other vertex connected by the hyperedge as follows:

$$w\left(e_i\right) = \sum_{v_i \in e_i} S_{i,j} \tag{11}$$

Intuitively, a hyperedge should be assigned a higher weight if the reviews within the hyperedge are close to each other.

In hypergraph learning stage, we define a testing vector $y \in \mathbb{R}^{|V|}$ to denote a definite testing review, in which only the item corresponding to the testing vertex is set to 1, otherwise 0. And we define a vector $f \in \mathbb{R}^{|V|}$ to represent the final correlation scores on domain $(-1, 1)$. In order to get the final correlation scores, we employ a regularization framework as follows:

$$\arg\min_{f} \{\Omega(f) + \lambda\Phi(f)\} \tag{12}$$

where the $\Omega(f)$ is a regularizer on the hypergraph,$\Phi(f)$ is an empirical loss, and $\lambda > 0$ is a parameter controls the trade-off between the regularizer and the loss. The purpose is to minimize the cost function, that is, the regularization framework. The regularizer is defined as follows:

$$\begin{aligned}\Omega(f) &= \frac{1}{2}\sum_{e \in E}\sum_{u,v \in V}\frac{\omega(e)h(u,e)h(v,e)}{\delta(e)}\left(\frac{f(u)}{\sqrt{d(u)}} - \frac{f(v)}{\sqrt{d(v)}}\right)^2 \\ &= \sum_{u \in V} f^2(u) - \sum_{e \in E}\sum_{u,v \in V}\frac{f(u)\omega(e)h(u,e)h(v,e)f(v)}{\sqrt{d(u)d(v)}\delta(e)} \\ &= f^T f - f^T D_v^{-\frac{1}{2}} HWD_e^{-1} H^T D_v^{-\frac{1}{2}} f \end{aligned} \tag{13}$$

It is a constraint that can make vertices sharing much more hyperedges in common to have more similar correlation scores. That is mean, two reviews will get a similar correlation score if they are similar to a lot of common reviews. Let $\Theta = D_v^{-(1/2)} HWD_e^{-1} H^T D_v^{-(1/2)}$ and $L = I - \Theta$, where I is the identity matrix, the above equation can be rewritten as:

$$\Omega(f) = f^T Lf \tag{14}$$

where L is the normalized hypergraph Laplacian and it is a positive semidefinite matrix. The classification loss is defined as follows:

$$\Phi(f) = \sum_{u \in V}(f(u) - y(u))^2 = (f - y)^T(f - y) \tag{15}$$

of which the function is to enforce the final correlation score is as close as possible to the initial testing vector. Thus, the regularization framework then can be rewritten as:

$$\arg\min_{f}\left\{f^T Lf + \lambda \|f - y\|^2\right\} \tag{16}$$

We get the partial derivative of the above objective function with respect to f, and the final correlation score, i.e.,

$$\begin{aligned}&\frac{\partial}{\partial f}\left[f^T\left(I - D_v^{-1/2} HWD_e^{-1} H^T D_v^{-1/2}\right)f + \lambda \|f - y\|^2\right] = 0 \\ &\Rightarrow f = \frac{\lambda}{1+\lambda}\left(I - \frac{\Theta}{1+\lambda}\right)^{-1} y \end{aligned} \tag{17}$$

The algorithm for sentiment classification of reviews on deep hypergraph model is summarized in Algorithm 2.

Algorithm 2 Sentiment Classification on Deep Hypergraph

```
Input: Simplified Text Matrix $STM$
Output: The final correlation score $f$ of reviews
(1) Extract features of each sample $STM_i$ and generate feature vector
(2) Calculate the reviews cosine similarity matrix $C$
(3) Calculate the affinity matrix $S$ from $C$
(4) for each sample do:
(5)     Construct a hyperedge by connecting its $k$ nearest neighbors based on $S$
(6) end for
(7) Generate the incidence matrix $H$ and the weight matrix $W$
(8) Calculate the matrix $\Theta$
(9) Given a testing review, calculate the final correlation score $f$
```

IV. EXPERIMENT

In this section, we introduce our experiments in detail. We first get word embedding representations by training a large scale English corpus from Wikipedia and semantic cliques by performing H-CFS algorithm. The results are shown in the subsection A. Then, we introduce the datasets for evaluating the presented model's performance in subsection B. Finally, the experimental results are given in subsection C.

A. WORD EMBEDDINGS LEARNING AND CLUSTERING

In our experiments, the pre-training of word embeddings is performed by using *Word2Vec*, which applied the continuous bag-of-words and skip-gram architectures to compute the representations of words [24]. We use the English Wikipedia latest page articles corpus, which is compressed to 14.3G, from [26] as the source data to train word embeddings. The corpus has been processed using the following steps: (1) removal of webpage tabs and paragraphs that are not in English; (2) replacing the special characters with non-western characters; (3) removal of sentences that are less than 30 characters long (including spaces). And we use all the words in lowercase and substitute each numerical digit with 0 similar to the strategy in [27]. Thus, we get a resulting clean corpus of size 14.8G.

In the stage of word embedding training, we set the words included in the vocabulary must appear at least 10 times and ignore all words with total frequency lower than 10. And the skip-gram method with a context window of size 5 is adopted to train our word embeddings. Then a vocabulary of 1422134 entries can be obtained, where each word embedding has a dimension of 128. The training time is about 6hours 3minutes with 24 threads.

After getting the vocabulary with word embeddings of 1422134 times 128, we perform H-CFS algorithm to cluster word embeddings. When running the algorithm, we set the thresholds N of 5000 and p of 0.006 empirically, which indicates the original word embeddings data is divided into sub-datasets containing 5000 words and the average number of neighbors will be 0.6% of the total word embeddings. Particularly, the cut-off distance is obtained by performing on total word embeddings with p of 0.006. Then, the cutoff distance d_c and the highest local density ρ_i of each sub-dataset can be obtained. TABLE 1 shows the results of several sub-datasets in detail. The last column indicates the result of all sub-datasets. After getting the clustering results of all sub-datasets, clustering process continues to execute on clustering center obtained by first-level clustering. And we set the cut-off distance of 0.63, which is slightly above the average of all cut-off distance from first-level clustering, in second-level clustering. The reason for this setting is to ensure that semantically irrelevant samples are partitioned together in second-level clustering. Finally, all the word embeddings are partitioned to 137011 clusters.

TABLE 1. Statistics of datasets.

Sub-Dataset	d_c	ρ_i	Number of clusters
Sub_4	0.5927	313	417
Sub_19	0.5927	290	310
Sub_61	0.5927	314	377
Sub_73	0.5927	477	829
Sub_158	0.5927	398	689
Sub_279	0.5927	297	409
All	0.5927	1057	165834

B. DATASETS

We evaluate the effectiveness of the proposed model by conducting experiments on product reviews [28] and movies reviews [29], among them, the product reviews including four different domain (books, DVD, electronics, kitchen) collected from Amazon. The product reviews of each domain contain 2000 reviews, half of which are labeled as positive and the other half as negative. The movies reviews data used in this work is composed of 5331 positive and 5331 negative processed movie reviews and has been widely used in the field of sentiment polarity classification. The statistics of the datasets are summarized in Table 2, where the second column is the average number of words for each domain dataset.

TABLE 2. Statistics of datasets.

Domain	Mean words	Number of samples
Book	175.82	2000
DVD	168.85	2000
Electronic	113.15	2000
Kitchen	95.42	2000
Movie	18.90	10662
Total	70.09	18662

All the review texts will be processed using the following steps: (1) removal of punctuations; (2) transformation of abbreviations to a full form (e.g., ''we've been'' becomes ''we have been''); (3) removal of numbers. We will randomly select 30% of the data from positive and negative reviews respectively as the training set, and the rest as the testing set.

In our experiments, the words out of the vocabulary in reviews are simple discarded, since they are low-frequency and often meaningless. And the thresholdτ, which is used as a constraint to fine-tune the detection of semantic units, is set to 0.65 to ensure that the semantic units are meaningful. An important factor affecting the results of our hypergraph learning model is the size of hyperedge, which depends on the number of k nearest neighbors. We first set k equals to 45 based on our experience. Besides, we set the value λ to be 9. Thus, the value of $1/1+\lambda$ listed in Eq.17 is equal to 0.1.

C. EXPERIMENTAL RESULTS

We conduct several group experiments to verify the effectiveness and rationality of the proposed model. The several group experimental setups are introduced as follows:

– In the first group of experiments, we simply concatenate each word embedding to construct projected matrix representation of each review, instead of detecting latent semantic units based on the word embedding clustering to replace the words, named EDHG

– In the second group of experiments, we detect latent semantic units by calculating the cosine similarity between the sum of n words and the word embeddings clustering center regardless of the central word for n words, named EDHG^{-1}.

– Then, we use conventional CNN, i.e. softmax function is adopted at the last layer, to get the probability distribution of the raw data, named ECDCNN.

– Finally, we fully implement our proposed method, named ECDHG.

And the results of these experiments are shown in Table 3.

TABLE 3. Classification accuracy of different combinations.

(%)	Book	DVD	Electronic	Kitchen	Movie
EDHG	84.11	84.39	83.63	83.98	82.25
ECDHG^{-1}	79.93	82.11	77.79	76.63	80.14
ECDCNN	83.67	84.04	83.36	83.93	81.33
ECDHG	84.21	85.23	85.57	84.87	85.05

The results show that better results than conventional CNN can be obtained by the deep hypergraph model with concatenation of each word embeddings only. Deep hypergraph model we proposed can capture the high-order relations among samples, which has contributions to the classification accuracy. Since there is some relevance among reviews with same sentiment polarity, syntactic or semantic correlation. And the results with the support of word embeddings clustering and the detection of latent semantic units, the fully experiments, outperforms CNN model with last layer of softmax by about 2% on the average accuracy. Moreover, the average accuracy of the fully experimental results is better than the experiments without the process of word embeddings clustering and latent semantic units detecting. The result

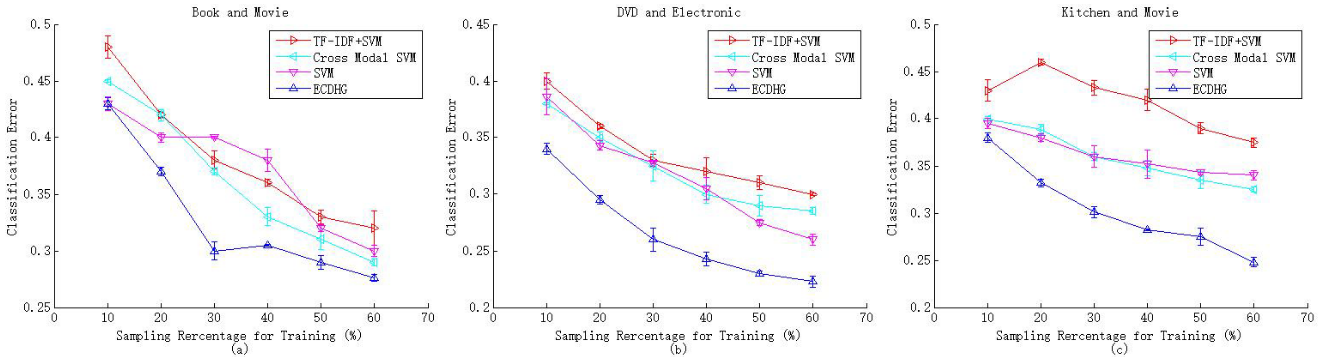


FIGURE 3. Classification error rates of different methods (ECDHG, SVM, TF-IDF + SVM and Cross-Modal SVM) on three mixed datasets (in percentage). Results on the (a) Book and Movie, (b) DVD and Electronic, (c) Kitchen and Movie.

indicates the semantic cliques of word embeddings can assist the model with detecting precise semantic units as the supervision information and improving the ability of projected matrices to express reviews. Nevertheless, the worse results obtained by ECDHG-1 than ECDCNN. This is mainly due to that some original information will be lost by detecting latent semantic units without regard to central word, which has a bad effect on the classification accuracy.

We compare our model with other seven representative methods of sentiment classification. The brief descriptions of these representative methods are as follows:

Lexicon-based: The lexicon-based classifier uses linguistic rules to detect the polarity of reviews [17].

SVM: Support Vector machine (SVM) constructs models that assign new examples to a certain category based on ''margin maximization'' strategy. We apply the features acquired by our feature extractor to train the classifiers.

TF-IDF + SVM: The features acquired by statistics Term Frequency (TF) and Inverse Document Frequency (IDF) and SVM are widely used baseline methods to build sentiment classifiers.

Cross-Modal + SVM: The authors in [28] apply sentiment score as an additional feature to train SVM besides the TF-IDF.

TME: A topic-level maximum entropy (ME) model is proposed in [30], which acquires topic-level features by modeling emotion labels, latent topics and sentiment scores.

CCNN: In [11], multi-scale semantic units are detected under the supervision of semantic cliques discovered by word embedding clustering. Then the projected matrix and expanded matrices are fed into CNN with max-pooling and softmax together.

***NB*(Naïve Bayes):** A simple and powerful probability model is extended from the Bayesian theorem.

As shown in Table 4, the comparisons of the proposed deep hypergraph model against the baselines are listed with the same setup of datasets. The proposed ECDHG performs better than other 7 methods in all cases. And the results of baseline methods perform differently on various datasets. The improvement of constructing hypergraph is quite prominent. This suggests that the word embeddings clustering by H-CFS algorithm can effectively assist the model with detecting meaningful latent semantic units and characterizing the original text accurately. Hypergraph learning can capture the high-order relations among samples, which has contributions to the classification accuracy. Especially, there is syntactic or semantic correlation among user reviews with same sentiment polarity.

TABLE 4. The comparison of different methods over all datasets.

(%)	Book	DVD	Electro nic	Kitche n	Movie
Lexicon-based	70.26	72.7	72.32	73.85	72.63
SVM	77.2	74.43	78.05	78.11	76.3
TF-IDF + SVM	73.07	68.29	73.29	70.57	72.33
Cross-Modal + SVM	78.36	74.93	79.21	78.14	75.58
TME	81.87	84.06	83.67	81.05	81.32
CCNN	82.77	84.1	82.83	83.02	82.5
NB	74.17	76.33	75.17	75.33	74.87
ECDHG	84.21	85.23	85.57	84.87	85.05

To evaluate the scalability of the model, we design experiments with mixing up different topic reviews. In detail, we randomly select 1000 reviews, half positive and half negative, from each two original datasets to build three mixed datasets, which contain Book and Movie, DVD and Electronic, Kitchen and Movie. In the experiment, we vary the percentages of training samples. Fig. 3 demonstrates the classification results of comparison with SVM, TF-IDF + SVM and Cross-Modal SVM. We can see that the ECDHG model outperforms the other three methods in all mixed datasets, especially in the mixed dataset of DVD and electronic. The results indicate that our model can achieve good results on samples of different topic, that is, the deep hypergraph model is independent of the topic of reviews.

We test the sensitivity of parameters p and hyperedge size k (i.e., the number of nearest neighbors) in ECDHG. In this experiment, we fix a parameter to test the effect of another

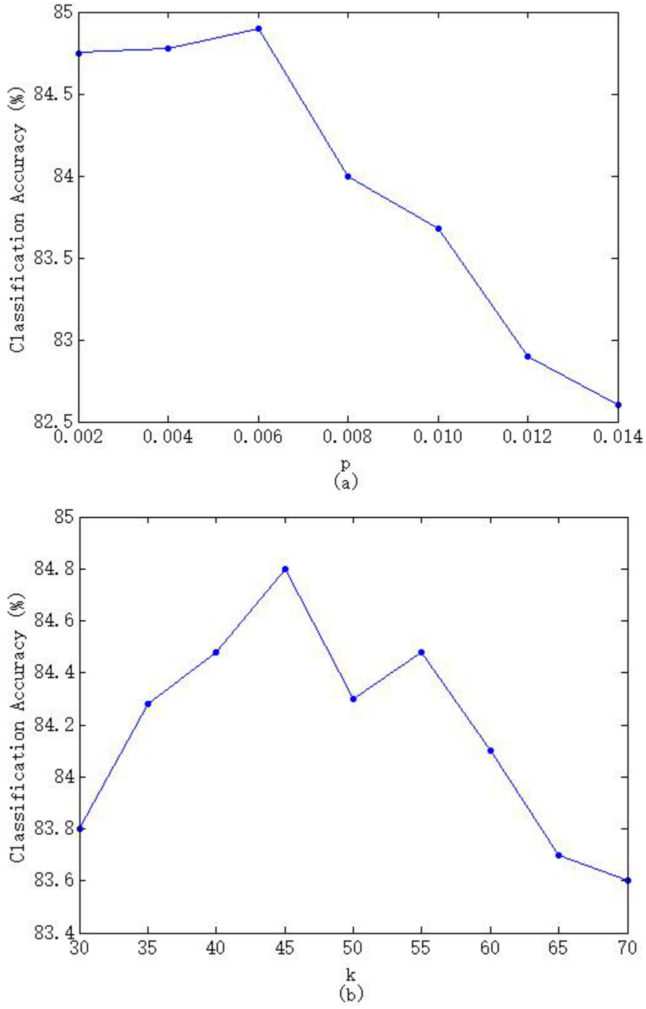


FIGURE 4. **Average classification accuracy with different (a) *p*(k is set to 45) and (b) k (*p* is set to 0.006).**

parameter on classification accuracy. We first set k equals to 45 empirically, and vary p with [0.002, 0.004, 0.006, 0.008, 0.01, 0.012, 0.014]. The average classification accuracy on the five datasets are shown in Fig. 4(a). Afterward, we set p to be 0.006 and vary k with [30, 35, 40, 45, 50, 55, 60, 65, 70]. The average classification accuracy on the five datasets are shown in Fig. 4(b). We can see that the average classification accuracy varies between 0.849 and 0.826 when p varies in the interval [0.002, 0.014]. In addition, as the value of p increases, the accuracy of classification decreases. Because the cut-off distance grows with the increasing of p, which lead to a decrease of relevance among word embeddings in a semantic clique. So that the accuracy of representations for projected matrices to original texts is reduced. And the average classification accuracy varies between 0.848 and 0.836 when k varies in the interval [30, 70]. The results demonstrate that the performance of ECDHG will not seriously degrade when the parameters vary widely. Besides, we can easily observe several fluctuations in the performance curve in Fig. 4. Because k and p, which are obtained by a large number of experiments, may not be optimal for some datasets.

V. CONCLUSION

In this paper, a deep hypergraph scheme is proposed for online reviews modeling and sentiment classification. One property of our presented model is to construct hypergraph to detect high-order relations among different reviews. Another property of the model is to use an improved hierarchical clustering algorithm to discover semantic cliques used for detecting precise semantic units as the supervision information. Extensive results of five domain benchmarks demonstrate the effectiveness and scalability of our ECDHG model. In the future, the fusing multi-modal features and task-specific embedding learning are employed to improve performance of ECDHG [31].

REFERENCES

[1] M. Salehan and D. J. Kim, "Predicting the performance of online consumer reviews: A sentiment mining approach to big data analytics," *Decision Support Syst.*, vol. 81, pp. 30–40, Jan. 2016.

[2] Q. Zhang, L. T. Yang, Z. Chen, P. Li, and M. J. Deen, "Privacy-preserving double-projection deep computation model with crowdsourcing on cloud for big data feature learning," *IEEE Internet Things J.*, to be published, doi: 10.1109/JIOT.2017.2732735.

[3] J. Gao, J. Li, and Y. Li, "Approximate event detection over multi-modal sensing data," *J. Combinat. Optim.*, vol. 32, no. 4, pp. 1002–1016, 2016.

[4] Q. Zhang, L. T. Yang, Z. Chen, and P. Li, "PPHOPCM: Privacy-preserving high-order possibilistic c-means algorithm for big data clustering with cloud computing," *IEEE Trans. Big Data*, to be published, doi: 10.1109/TBDATA.2017.2701816.

[5] X. H. Phan, M. L. Nguyen, and S. Horiguchi, "Learning to classify short and sparse text & Web with hidden topics from large-scale data collections," in *Proc. 17th Int. Conf. World Wide Web, ACM*, 2008, pp. 91–100.

[6] M. Sahami and T. D. Heilman, "A Web-based kernel function for measuring the similarity of short text snippets," in *Proc. 15th Int. Conf. World Wide Web, ACM*, 2006, pp. 377–386.

[7] X. Yan, J. Guo, Y. Lan, and X. Cheng, "A biterm topic model for short texts," in *Proc. 22nd Int. Conf. World Wide Web*, 2013, pp. 1445–1456.

[8] B. Pang, L. Lee, and S. Vaithyanathan, "Thumbs up?: Sentiment classification using machine learning techniques," in *Proc. EMNLP Conf.*, 2002, pp. 79–86.

[9] Q. Zhang, L. T. Yang, Z. Chen, and P. Li, "An improved deep computation model based on canonical polyadic decomposition," *IEEE Trans. Syst., Man, Cybern., Syst.*, to be published, doi: 10.1109/TSMC.2017.2701797.

[10] Q. Zhang, L. T. Yang, Z. Chen, and P. Li, "A survey on deep learning for big data," *Inf. Fusion*, vol. 42, pp. 146–157, Jul. 2018.

[11] P. Wang, B. Xu, J. Xu, G. Tian, C. Liu, and H. Hao, "Semantic expansion using word embedding clustering and convolutional neural network for improving short text classification," *Neurocomputing*, vol. 174, pp. 806–814, Jan. 2016.

[12] Q. Zhang, M. Lin, L. T. Yang, Z. Chen, and P. Li, "Energy-efficient scheduling for real-time systems based on deep Q-learning model," *IEEE Trans. Sustain. Comput.*, to be published, doi: 10.1109/TSUSC.2017.2743704.

[13] P. Li, Z. Chen, L. T. Yang, Q. Zhang, and M. J. Deen, "Deep convolutional computation model for feature learning on big data in Internet of Things," *IEEE Trans. Ind. Informat.*, vol. 14, no. 2, pp. 790–798, Feb. 2018.

[14] Q. Zhang, C. Zhu, L. T. Yang, Z. Chen, L. Zhao, and P. Li, "An incremental CFS algorithm for clustering large data in industrial Internet of Things," *IEEE Trans. Ind. Informat.*, vol. 13, no. 3, pp. 1193–1201, Jun. 2017.

[15] Q. Zhang and Z. Chen, "A distributed weighted possibilistic C-means algorithm for clustering incomplete big sensor data," *Int. J. Distrib. Sensor Netw.*, vol. 10, no. 5, pp. 430814:1–430814:8, 2014.

[16] S. Baccianella, A. Esuli, and F. Sebastiani, "SentiWordNet 3.0: An enhanced lexical resource for sentiment analysis and opinion mining," in *Proc. 7th Int. Conf. Lang. Resour. Eval.*, 2010, pp. 2200–2204.

[17] 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.

[18] A. Andreevskaia and S. Bergler, "When specialists and generalists work together: Overcoming domain dependence in sentiment tagging," in *Proc. ACL Conf.*, 2008, pp. 290–298.

[19] M. Hu and B. Liu, "Mining and summarizing customer reviews," in *Proc. SIGKDD Conf.*, 2004, pp. 168–177.

[20] Y. Lu, X. Kong, X. Quan, W. Liu, and Y. Xu, "Exploring the sentiment strength of user reviews," in *Proc. 11th Int. Conf. Web-Age Inf. Manage.*, Jiuzhaigou, China, 2010, pp. 471–482.

[21] G. Wang, J. Sun, J. Ma, K. Xu, and J. Gu, "Sentiment classification: The contribution of ensemble learning," *Decision Support Syst.*, vol. 57, pp. 77–93, Jan. 2014.

[22] P. D. Turney, "Thumbs up or thumbs down?: Semantic orientation applied to unsupervised classification of reviews," in *Proc. ACL Conf.*, 2002, pp. 417–424.

[23] A. Mnih and Y. W. Teh, "A fast and simple algorithm for training neural probabilistic language models," in *Proc. ICML Conf.*, 2012, pp. 419–426.

[24] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Proc. NIPS Conf.*, 2013, pp. 3111–3119.

[25] R. Johnson and T. Zhang, "Deep pyramid convolutional neural networks for text categorization," in *Proc. ACL Conf.*, 2017, pp. 562–570.

[26] *Wikipedia English Corpus*. [Online]. Available: https://dumps.wikimedia.org/enwiki/

[27] T. Luong, R. Socher, and C. D. Manning, "Better word representations with recursive neural networks for morphology," in *Proc. CoNLL Conf.*, 2013, pp. 104–113.

[28] Z. Chen, F. Lu, X. Yuan, and F. Zhong, "TCMHG: Topic-based cross-modal hypergraph learning for online service recommendations," *IEEE Access*, to be published, doi: 10.1109/ACCESS.2017.2782668.

[29] B. Pang and L. Lee, "Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales," in *Proc. ACL Conf.*, 2005, pp. 115–124.

[30] Y. Rao, H. Xie, J. Li, F. Jin, F. L. Wang, and Q. Li, "Social emotion classification of short text via topic-level maximum entropy model," *Inf. Manage.*, vol. 53, no. 8, pp. 978–986, Dec. 2016.

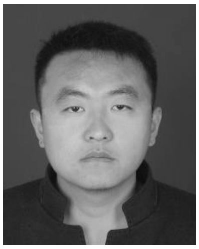
[31] Q. Zhang, L. T. Yang, Z. Chen, and P. Li, "High-order possibilistic c-means algorithms based on tensor decompositions for big data in IoT," *Inf. Fusion*, vol. 39, pp. 72–80, Jan. 2018.

XU YUAN received the B.E. degree in ship electronic and electrical engineering from Dalian Maritime University, China, in 1992, the master's degree in information system and management from Carnegie Mellon University, USA, in 2002, and the Ph.D. degree in management science and engineering from the Dalian University of Technology, China, in 2006. He is currently an Associate Professor and a Supervisor of master's candidates with the Software School of Dalian University of Technology, China. His research interests include big data visualization, industry data fusion, artificial intelligence, education, and talent big data.

MINGYANG SUN received the B.E. degree in software engineering from the Dalian University of Technology, China, in 2016. He is currently pursuing the M.Sc. degree with the Software School of Dalian University of Technology. His current research interests include machine learning, data mining, and natural language processing.

ZHIKUI CHEN received the B.E. degree in mathematics from Chongqing Normal University, Chongqing, China, in 1990, and the Ph.D. degree in solid mechanics from Chongqing University, Chongqing, in 1998.He is currently a Professor with the School of Software Technology, Dalian University of Technology, China. His research interests include Internet of Things and big data.

JING GAO received the B.E. degree in computer science and technology and the Ph.D. degree in computer software and theory from the Harbin Institute of Technology, China, in 2008 and 2015, respectively. She is currently an Assistant Professor with the School of Software Technology, Dalian University of Technology, China. Her current research interests include multi-modal data mining and deep learning.



PENG LI received the B.E. degree in electronic and information engineering from Dezhou University, Dezhou, China, in 2012. He is currently pursuing the Ph.D. degree in software engineering with the Dalian University of Technology, Dalian, China. His research interests include deep learning and big data.