Deep User Modeling for Content-based Event Recommendation in Event-based Social Networks

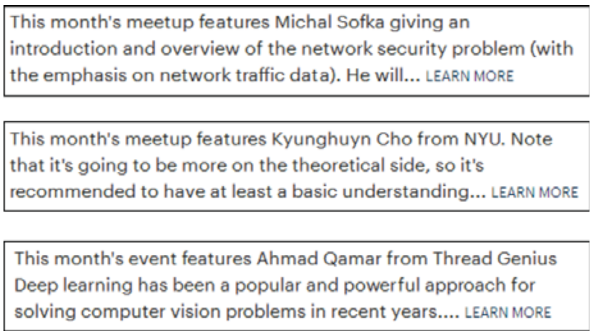
Zhibo Wang[†,‡,≀], Yongquan Zhang[†], Honglong Chen[£], Zhetao Li[¶,*] and Feng Xia[§]
[†]School of Cyber Science and Engineering, School of Computer, Wuhan University, P. R. China
[‡]Jiangsu Key Lab. of Big Data Security & Intelligent Processing, NJUPT, P. R. China
[≀]Key Lab of Aerospace Information Security and Trusted Computing, Wuhan University, P. R. China
[£]College of Information and Control Engineering, China University of Petroleum, P. R. China
[¶]College of Information Engineering, Xiangtan University, P. R. China
[§]School of Software, Dalian University of Technology, P. R. China
Email: {zbwang, dellenzhang}@whu.edu.cn, {honglongchen1984, liztchina}@gmail.com, f.xia@ieee.org

***Abstract*—Event-based social networks (EBSNs) are the newly emerging social platforms for users to publish events online and attract others to attend events offline. The content information of events plays an important role in event recommendation. However, the content-based approaches in existing event recommender systems cannot fully represent the preference of each user on events since most of them focus on exploiting the content information from events' perspective, and the bag-of-words model, commonly used by them, can only capture word frequency but ignore word orders and sentence structure. In this paper, we shift the focus from events' perspective to users' perspective, and propose a Deep User Modeling framework for Event Recommendation (DUMER) to characterize the preference of users by exploiting the contextual information of events that users have attended. Specifically, we utilize convolutional neural network (CNN) with word embedding to deeply capture the contextual information of a user's interested events and build up a user latent model for each user. We then incorporate the user latent model into probabilistic matrix factorization (PMF) model to enhance the recommendation accuracy. We conduct experiments on the real-world dataset crawled from a typical EBSN, Meetup.com, and the experimental results show that DUMER outperforms the compared benchmarks.**

I. Introduction

Event-based social networks (EBSNs), such as Meetup [1] and Eventbrite [2], are the newly emerging social platforms for users to publish events online and attract other users in local city to attend events offline. The special combination of online social networks and offline social interactions provides a new way for users to establish and enhance social ties, which has attracted thousands of users and enabled EBSNs to gain momentum in recent years. According to the statistics of Meetup [1], it has about 30 million registered users from 180 countries and more than 580,000 events are organized per month.

Given the sheer volume of events in EBSNs, personalized event recommendation is required to solve the information overload problem and recommend the most interested events to users. Although recommender systems have been extensively studied, however, the unique features of events make event


This month's meetup features Michal Sofka giving an introduction and overview of the network security problem (with the emphasis on network traffic data). He will... LEARN MORE

This month's meetup features Kyunghuyn Cho from NYU. Note that it's going to be more on the theoretical side, so it's recommended to have at least a basic understanding... LEARN MORE

This month's event features Ahmad Qamar from Thread Genius Deep learning has been a popular and powerful approach for solving computer vision problems in recent years.... LEARN MORE

Fig. 1. Examples of the content of events in EBSNs

recommendation more challenging than traditional recommendation problems (e.g., movie recommendation on Netflix [3] or friend recommendation [4]). The first unique feature is that each event in EBSNs has a short life cycle requiring the event recommendation to be made before the event starts, which is different from items (e.g., books on Amazon) in traditional recommender systems that no time period information is taken in account. More importantly, the newly published events are scheduled in a near future, having little or no historical attendance, and the user feedback information are given only after an event has ended. In other words, the user feedback information (e.g., ratings and comments) commonly used by classic recommender systems becomes useless in EBSNs. Therefore, these unique cold-start natures of events make event recommendation problem in EBSNs very challenging, and the classic recommendation algorithms (e.g., collaborative filtering and matrix factorization) cannot be directly applied for event recommendation in EBSNs.

Several event recommender systems have been developed to recommend the most interested events to users [5]–[9]. The content of events plays an important role in event recommendation as it contains many important information, such as the topics and locations, as shown in Figure 1. Due to its importance, content-based approaches (e.g., LDA [10]) on events are commonly adopted in existing event recommender systems to calculate the similarity score between the events in terms of contents for recommendation. However, the similarity-based

[*]Corresponding author.

approaches actually are not very suitable or effective for event recommendation because of the cold-start nature of a new event in EBSNs. Besides, compared with an event with short life cycle, a user in EBSNs has a much longer life span and usually has attended many events, which encourages us to explore a user's interests for event recommendation instead of calculating the similarity between events. Based on this intuition, we argue that it is more reasonable to exploit the contextual information from the user's perspective to capture the preference of a user on events, than to recommend events based on similarity between new events and historical events.

The bag-of-words model is a common text representation model utilized in existing recommendation systems. However, it cannot fully capture the contextual information of the content, since it only represents word frequency but ignores sentence structure and word orders of the content. Recently, deep learning techniques, such as recurrent neural networks (RNN) [11] and convolutional neural networks (CNN) [12], show great potential for learning effective feature representation from text content [11]–[14]. In particular, RNN has a recurrent network architecture to model sequential input and output, while CNN is named for its convolutional operation that extracts the local feature of the data. Inspired by these work, in this paper, we utilize CNN to deeply capture the contextual information of events from the user's perspective and incorporate them into a probabilistic matrix factorization framework to conduct event recommendation.

In this paper, we exploit the content of events for event recommendation in EBSNs, and shift the focus from event's perspective to user's perspective. We propose a Deep User Modeling for Event Recommendation (DUMER) to characterize the latent preferences of users by deeply exploiting the contextual information of events that users have attended. In particular, we use CNN with word embedding to deeply capture the contextual information of user's interested events and build up a user latent factor model for each user. Then we incorporate the user latent factor into probabilistic matrix factorization (PMF) model to enhance the recommendation accuracy. Note that [15] proposed a similar framework called ConvMF that integrates CNN into PMF to capture contextual information of documents and enhance the rating prediction accuracy. However, ConvMF [15] cannot be directly applied to event recommendation in EBSNs since the unique natures of EBSNs are different from traditional recommender systems. Moreover, different from [15] that focuses on exploiting item documents in an event's perspective, our work shifts the focus to a user's perspective, and applies CNN on user documents to better capture the user preference considering the unique characteristics of EBSNs.

In summary, the main contributions of this work include:

- We utilize the content of events from users' perspective for event recommendation, which characterizes the latent preference of users by deeply exploiting the contextual information of events that users have attended.
- We propose DUMER that integrates CNN into PMF to utilize the contextual information of users' interested events to accurately recommend new events to users. Specifically, we apply CNN to extract high-level features of user documents to form a user latent factor, and then incorporate the user latent factor into PMF model to predict the preference of a user to an event.
- Extensive experiments was conducted on real-word dataset crawled from Meetup.com. We compared DUMER with other item-oriented recommendation algorithms and derivations of them to validate our intuition. Experimental results show that our algorithm outperform the others.

The rest of the paper is organized as follows. We discuss the related work in Section II and describe the content-based event recommendation problem in Section III. We present the high-level overview and the design of DUMER in Section IV and conduct extensive experiments in Section V. Finally, we conclude the paper in Section VI.

II. Related Work

In this section, we first briefly discuss the work of event recommendation in EBSNs, and then review the neural network methods that have been adopted in recommender systems.

Event-based social networks were firstly defined in [16] as a co-existence of both online and offline social interactions and they proposed an event recommendation algorithm based on diffusion model using the topological features of the heterogenous network. Thereafter, Qiao et al. [6] exploited heterogeneous social and geographical information of an EBSN and proposed a Bayesian matrix factorization approach for event recommendation. Du et al. [17] extended Qiao's work by integrating content, location and time into the recommendation algorithm. More thoroughly, Macedo et al. [7] designed a contextual learning algorithm that takes time, location, content, social relations and group features into consideration. Afterwards, Zhang et al. [8] proposed a collective Bayesian poission factorization model to combine content, location, social relation and more importantly event organizer information to infer user latent factors. Recently, Wang et al. [18] proposed a novel event recommendation algorithm that combines EBSN with other classic online social networks to exploit the social information of event hosts to improve the accuracy of event recommendation.

However, content-based approaches utilized in these event recommendation algorithms are mainly LDA or TF-IDF, which suffers from weak semantic understanding of contents since it only represents word frequency but ignores word orders and sentence structure. Moreover, the latent preference of users on events cannot be fully characterized by existing content-based approaches since the content of users' interested events is not exploited from user's perspective, although the latent topic models of events can be captured.

Recently deep learning techniques has been used to effectively exploit contextual features from contents [14], [15], [19]. Wang et al. [19] proposed collaborative deep learning (CDL) that uses Stacker Denosing Auto-Encoder (SDAE) to realize deep representation learning for the content information. Tang et al. [14] designed a neural network to factor in user-specific modification to the meaning of a certain word for review rating prediction. Kim et al. [15] utilized CNN to capture the contextual information of item description documents and integrated it with PMF for accurate rating prediction. However, ConvMF [15] cannot be directly applied to event recommendation in EBSNs because the unique natures of EBSNs are different from traditional recommender systems. Moreover, different from [15] that focuses on exploiting item documents in an event's perspective, our work shifts the focus to a user's perspective, and applies CNN on user documents to better capture the user preference considering the unique characteristics of EBSNs.

Although algorithms mentioned above are designed for recommendation tasks, none of existing recommendation algorithms based on deep learning can be directly applied for event recommendation. This is because the user feedback information (e.g, ratings and comments) serving as the ground truth in traditional recommender systems become useless since users will only rate or comment an event after it ends. However, event recommendation should be made before an event starts since it has a short life time. To overcome this limitation, we propose to exploit the events' content information from a user's perspective to capture user's preferences on events. Inspired by the success of deep learning techniques, we utilize deep learning to extract effective features of contextual information of events to form a user latent factor for event recommendation.

III. Problem Formulation

Give a user u, the set of events he has attended is denoted by $E_u = \{e_1, e_2, \cdots, e_m\}$, where m is the total number of events u has attended. Let $\hat{E} = \{\hat{e}_1, \hat{e}_2, \cdots, \hat{e}_\varphi\}$ denote the set of ongoing events that has not started. Given the user u, event recommendation aims to find out that what events in $\hat{E}$ will interest u and recommend them to u accordingly. Note the the ongoing events in $\hat{E}$ have no rating at all, which makes the event recommendation problem more challenging than traditional recommendation problems.

In this paper, we cast the event recommendation problem as an attendance matrix approximation problem since ratings cannot be utilized for recommendation in EBSNs. The attendance matrix is constructed based on users' RSVP* history. Formally, suppose we have K users, M events consisting all the past events and ongoing events. Let $A \in \mathbb{R}^{K*M}$ stands for the attendance matrix, where its element A_{ij} is 1 when user i attended event j, otherwise 0. Note that for an ongoing event j, $A_{ij} = 1$ if user i responds "yes", otherwise $A_{ij} = 0$. $U \in \mathbb{R}^{K*D}$ and $V \in \mathbb{R}^{D*M}$ are user and event latent factor vectors, respectively, where D is the the number of latent factors and $D \ll min(K, M)$.

*RSVP is a term in meetup.com, means "please respond". A user can RSVP "yes", "no" or "maybe" to an event, indicating that the user "will", "will not", or "maybe" attend this event.

The attendance matrix approximation problem can be formulated as maximizing the posterior probability, which is presented as follows:

$$p(A|U, V, \delta^2) = \prod_{i}^{K} \prod_{j}^{M} \mathcal{N}(A_{ij}|U_i V_j, \delta^2)^{I_{ij}} \quad (1)$$

where $\mathcal{N}(x|\mu, \delta^2)$ is the probability density function of Gaussian distribution, whose mean is μ and variance is δ^2. I is the indicator matrix, the element of which, I_{ij}, is 1 when user i attended or responds "yes" to event j, otherwise 0. After obtaining U and V, we can calculate the preference of a user u_i at an ongoing event $\hat{e}_j$, that is A_{ij}, between users and ongoing events. For the user u_i, the top-N ongoing events with largest preference will be recommended.

IV. Deep User Modeling for Event Recommendation

In this section, we first give a high-level overview of the proposed Deep User Modeling for Event Recommendation (DUMER) algorithm, and then describe its components in detail.

Figure 2 shows the architecture of the proposed DUMER. It utilizes CNN with word embedding to deeply capture the contextual information of users' interested events from users' perspective, and then applies the PMF model to take the convoluted features as the user latent factor to approximate the attendance matrix A. With the predicted attendance matrix, we can conduct top-N recommendation to user u_i according to the value of A_{ij}.

A. Word Embedding based Representation

We observe that it is not events whose descriptions have similar words to event descriptions that a user has attended, but events whose descriptions have similar meaning to them, that interest users. For example, if a user has recently attended an event of traveling to France, it is not likely that he/she would attend an event whose descriptions have the words *traveling* and *France* again, but more likely to attend an event that has the words *traveling*, and *Europe* or *England* in its description. However, traditional bag-of-word model can only capture the similarity of documents with the co-occurrence of words, but fail to reflect word orders and relations. Therefore, we resort to the word embedding technique in [20] to better represent the content of events.

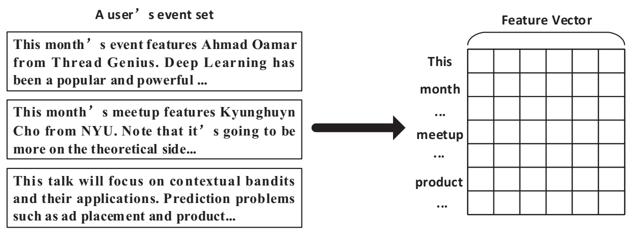
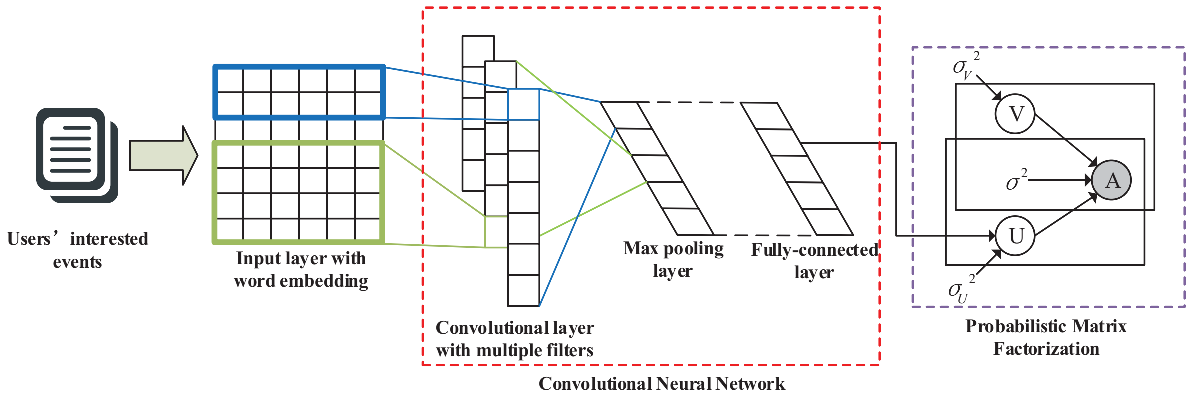

A user's event set
This month's event features Ahmad Oamar from Thread Genius. Deep Learning has been a popular and powerful ...
This month's meetup features Kyunghuyn Cho from NYU. Note that it's going to be more on the theoretical side...
This talk will focus on contextual bandits and their applications. Prediction problems such as ad placement and product...
Feature Vector
This
month
...
meetup
...
product
...

Fig. 3. Word embedding representation

In this paper, we adopt the pre-trained word embedding model Glove [20]. Each word is represented as a feature vector,



Users' interested events
Input layer with word embedding
Convolutional layer with multiple filters
Max pooling layer
Fully-connected layer
Convolutional Neural Network
σ_V^2
V
σ^2
A
U
σ_U^2
Probabilistic Matrix Factorization

Fig. 2. The architecture of DUMER.

and words with similar meanings have a close distance to each other. We utilize the word embedding model to transform each event description of users' interested events into a word embedding matrix, as shown in Figure 3. And then we concatenate them together to form the user word embedding matrix. We use X_i to denote the user word embedding matrix for user u_i.

B. *CNN Architecture of DUMER*

CNN has been widely applied to information retrieval (IR) and natural language processing (NLP) to exploit the local features of documents. Shen et al. utilized CNN to capture both the word n-gram level and sentence-level contextual structures of documents for IR [21]. Kalchbrenner et al. designed a dynamic convolutional neural network for semantic modeling of sentences [12], and achieved excellent performance in sentiment prediction. Therefore, we utilize CNN to deeply capture the contextual information of users' interested events, and in particular characterize the preference of users on events from users' perspective. The CNN of DUMER has three layers in DUMER: the convolutional layer, the pooling layer and the fully-connected layer.

Convolutional Layer: The convolutional layer can be viewed as sliding window based feature extraction [21]. It extracts local features around each word by a window and concatenates them together to form the feature map. Different from image processing, we set the width of the convolutional filters as the width of the input matrix in natural language processing and slide the filters over full rows of the input matrix to obtain the feature maps. As shown in Figure 2, the convolution layer transforms the input matrix into multiple feature maps according to multiple convolutional filters. After this, we add a non-linear operation called rectified linear unit (ReLU) since the real-word data usually are non-linear. Note that there are several non-linear activation functions such as sigmoid, tanh. Here we select ReLU to avoid vanishing gradient during the training process.

The convoluted features can be expressed as follows:

$$h_i^j = f(W_c^j * X_i) \tag{2}$$

where $W_c^j \in \mathbb{R}^{w*d}$ is the jth convolutional filter (w is the window size and d is the dimension of word embedding), X_i is the word embedding matrix of user u_i, and $*$ stands for the convolution operation. $f(\cdot)$ represents the ReLU function and h_i^j is the convoluted feature vector of X_i with the jth convolutional filter. In our experiments, we set the total amount of filters as 300, that is, 100 filters for each word window size (3, 4 and 5), because different convolutional filters can detect different types of features from the document.

Pooling Layer: The convoluted features obtained from the convolutional layer are of variable length. In order to aggregate them into a fixed size vector, we utilize the *1-max-pooling* operation [22] to select the largest number from each rectified feature map. In this way, we extract a scalar from each rectified feature map that are of different length. The blocks that are of different color in pooling layer of Figure (2) are the scalars of different rectified feature maps. Then we concatenate those scalars to form a fixed-length feature vector for the next layer. The feature vector at the pooling layer for user u_i can be expressed as follows:

$$h_p^i = [max({h_i}^1), max({h_i}^2), \ldots, max({h_i}^{n_c})] \tag{3}$$

where n_c is the number of filters, and h_p^i is the pooled feature vector for user document of user u_i. The pooling operation can reduce the dimension of features and speed up the training process, and control overfitting.

Fully Connected Layer: The fully connected layer is a traditional multi-layer perceptron that uses a soft-max activation function to learn non-linear combinations of those high-level representation features. The output of the convolutional and pooling layers represent the high-level features of the input matrix of the contents. The goal of the fully connected layer is to use these features to form the user latent factor for PMF. It can be formulated as follows:

$$S_i = F(W_s * h_p^i) \tag{4}$$

where h_p^i is the global feature vector after max-pooling for user u_i, and W_s is the weight matrix of the multi-layer perceptron. $F(\cdot)$ is the soft-max function and S_i is the final feature vector of user u_i. We concatenate the final feature vector of each user to form the overall feature matrix S. We take the overall feature matrix S as our user latent factor for probabilistic matrix factorization.

C. Probabilistic Matrix Factorization

As shown in Eq. (1), we adopt a probabilistic linear model with Gaussian observation noise to represent the attendance matrix of users. With the user feature matrix S that we obtained from CNN, we further add an Gaussian noise variable to better fit the attendance matrix. Hence, the user latent factor U is

$$U = S + E \tag{5}$$

where E is a Gaussian noise matrix, whose entries E_{ij} follows zero-mean Gaussian distribution whose variance is ${\delta_U}^2$ and S is the feature vector matrix of users from Eq. (4). As for event latent factor, we initialize the item latent factor following zero-mean Gaussian distribution whose variance is ${\delta_V}^2$:

$$\begin{aligned} p(E|{\delta_U}^2) &= \prod_i^K \mathcal{N}(E_i|0, {\delta_U}^2) \\ p(V|{\delta_V}^2) &= \prod_j^M \mathcal{N}(V_j|0, {\delta_V}^2) \end{aligned} \tag{6}$$

In this way, we can maximize the posterior probability to conduct probabilistic matrix factorization.

$$\begin{aligned} &\max_{U,V,W_s,W_c} p(U,V,W_s,W_c|A,\mathcal{X},\delta^2,{\delta_U}^2,{\delta_V}^2) \\ &= \max_{U,V,W_s,W_c} [p(A|U,V,\delta^2)p(V|{\delta_V}^2)p(U|W_s,W_c,X,{\delta_U}^2)] \end{aligned} \tag{7}$$

where $\mathcal{X}$ is the whole event description sets. W_c and W_s are network weight parameters in Eq. (2) and Eq. (4), respectively.

D. Training DUMER

As described above, there are many parameters in our DUMER model. To optimize those parameters, we devise an optimization algorithm to obtain the maximum a posteriori estimates. Maximizing the posterior probability is equivalent to minimizing the joint log-likelihood of U, V, W_s and W_c. So the formula can be expressed as follows.

$$\begin{aligned} \mathcal{L}(U,V,W) = &\min_{U,V,W_s,W_c} [\frac{1}{2}I||A-UV||_2 + \frac{\lambda_V}{2}||V||_2 \\ &+ \frac{\lambda_U}{2}||U-S||_2] \end{aligned} \tag{8}$$

where λ_U and λ_V are the hyper parameters.

We utilize the coordinate descent algorithm to obtain the optimal U and V. It iteratively optimizes a latent variable by fixing the remaining variables until convergence. Therefore, by fixing V, W_s and W_c, we can derive the optimal U by taking derivative of Eq. (8) with respect to U and setting it to zero. In this way, the updating rule of U can be expressed as follows.

$$U_i \leftarrow (VI_iV^T + \lambda_U I_K)^{-1}(VA_i + \lambda_U S_i) \tag{9}$$

where U_i stands for the update value of user latent vector for user i, I_i is a diagonal matrix whose diagonal elements are $I_{ij}, (j = 1, \dots, M)$, R_i is the i-th row of A and S_i is the feature vector of user i obtained from the CNN architecture. With this updating rule of U, we can get our optimal user latent vector U when the convergence is reached.

By fixing U, W_s and W_c, we can do the same with V, and get the optimal V.

$$V_j \leftarrow (UI_jU^T + \lambda_V I_K)^{-1}UA_j \tag{10}$$

where I_i and I_j are diagonal matrices whose diagonal elements are $I_{ij}, (j = 1, \dots, M)$, and $I_{ij}, (i = 1, \dots, K)$ respectively, and A_i and A_j are the ith row and jth column of A respectively.

As for W_s and W_c, we follow the method used in [19]. We fix U and V, and utilize back propagation algorithm to derive the optimal W_s and W_c. We use back propagation to calculate the gradients of the error with respect to all weights in the network, W_s and W_c, and use gradient descent to update W_s and W_c to minimize the output error.

With optimized U, V and W_s and W_c, we can, therefore, calculate the missing values of A between users and ongoing events as follows:

$$\hat{A}_{ij} \approx U_iV_j = (S_i + E_i)V_j \tag{11}$$

Note that S_i is the feature vector of user i obtained from the CNN architecture, and E_i is i-th row of the Gaussian noise matrix E we added in Eq. (5). At last, we can rank events according to the predicted user attendance matrix $\hat{A}_{ij}$ for user u_i and top-N events will be returned as the recommendation list.

V. Performance Evaluation

In this section, we present the experiments we conducted on real-world dataset to evaluate the performance of DUMER and compared it with existing content-based event recommendation algorithms. In particular, we aim to answer the following two questions: 1) Is it more suitable for event recommendation to exploit the content from users' perspective than events' perspective? 2) What is the accuracy of event recommendation of DUMER, and how better is it compared with the state-of-the-art?

A. Dataset

We used the dataset crawled from Meetup in [7] in experiments for performance evaluation. Instead of directly feeding the data to the algorithms, we conducted text preprocessing as follows. We removed all non-vocabulary words for event description documents, such as numbers and non-English words. Then, we removed stop words to eliminate non-meaningful words with high frequency. After this, we selected events whose description document has at least 10 words. To avoid noise, we only select users who have attended at least 5 events for our experiments. After data preprocessing, the statistics of our dataset are shown in Table I.

TABLE I
DATASET STATISTICS

# of Users	154503
# of Events	529979
# of attendances	4113978
Avg. length of event description	216

B. Compared Algorithms and Parameter Setting

We compare DUMER with the following algorithms to demonstrate the effectiveness of the DUMER.

- PMF: Probabilistic Matrix Factorization is a baseline recommendation model in [23], which utilizes a generative model for user and item latent factor to conduct recommendation.
- CTR: Collaborative Topic Regression [24] is a state-of-the-art recommendation model that combines collaborative filtering and topic modeling to exploit ratings and item documents.
- CDL: Collaborative Deep Learning [19] is another state-of-the-art recommendation model that tightly couples deep representation learning for the content information of items and collaborative filtering for the rating matrix.
- ConvMF: Convolutional Matrix Factorization [15] is a document context-aware recommendation model that integrates CNN into PMF to capture contextual information in description documents for the rating prediction.
- DUMER$-$: DUMER$-$ is our algorithm with the one-hot vector[†] instead of the word embedding vector. DUMER$-$ is based on bag-of-word model for comparison.

We conducted our experiments using Python and Keras library with Nvidia Quadro 8000 GPU on Linux Ubuntu platform. For CNN in DUMER, we used mini-batch based RMSprop to train the weight parameters. We set the batch size to 256 considering the size of the dataset. The dimension of latent factor of U and V is set to 50 in our experiments. As for hyper parameters λ_U and λ_V, we utilize the grid search strategy to find their best values. In our experiments, DUMER achieves the best performance when λ_U is set to 10 and λ_V is set to 100. As for PMF, CTR and CDL, we set the dimension of latent factor of U and V as 50 and find the appropriate parameters (λ_U, λ_V) of each model by grid search. The best performing values found for PMF, CTR and CDL are $(10, 10)$, $(10, 0.1)$, $(1, 10)$, respectively.

C. Evaluation Metric

Since most of compared methods are tailored for rating prediction task, we adopt the root mean squared error (RMSE) to compare the performances. RMSE evaluates the error between the attendance matrix A and the predicted UV. The lower RMSE results are, the better the recommendation performance is. In our experiments, we randomly split the dataset into training set, validation set and test set in the proportion of $(0.8, 0.1, 0.1)$. The formula of RMSE is as follows.

[†]one-hot vector is a vector which is 0 in most dimensions, and 1 in a single dimension.

$$RMSE = \sqrt{\frac{\sum_i^K \sum_j^M (A_{ij} - U_i V_j)^2}{||A||}} \tag{12}$$

We generate different ratios of training set to investigate the influence of training set on RMSE, e.g., (0.6, 0.2, 0.2) and (0.4, 0.3, 0.3). The results are shown in Section V-D.

In order to evaluate the recommendation accuracy of DUMER, we randomly mark off $x\%$ of events attended for each user as the test data. We train DUMER with the remaining data and evaluate its performance of recovering these marked-off events in the top-N recommendation list. We select Precision@N and Recall@N, and the normalized discounted cumulative gain (NDCG@N) as our evaluation metrics. However, zero ratings may indicate that a user is not interested in an event or does not know about it. Thus, it is difficult to accurately compute the precision metric. Therefore, we focus on the recall metric to evaluate our algorithm and other comparison algorithms. The recall metric only considers the positively predicted events within top-N recommendation list. For each user, we define recall@N as follows.

$$recall@N = \frac{\#hits}{\#total} \tag{13}$$

where $\#hits$ are the number of events that the user likes among the top-N recommendation list, and $\#total$ are the total number of events that the user has attended. The recall for the entire system will be summarized by the average recall of all users.

The definition of the normalized discounted cumulative gain (NDCG@N) are as follows.

$$NDCG(N) = Z_N \sum_{j=1}^{N} \frac{2^{r(j)} - 1}{log(1+j)} \tag{14}$$

where Z_N is the normalization factor and $r(j)$ is the rank position of event j. In our experiments, we set the mark-off rate as 20% as the default. NDCG@50 assumes that events appearing earlier in a recommendation list are more important and assigns more weight to the ground-truth events that are ranked higher. The higher the NDCG results are, the better the recommendation performance is. We also utilize different mark-off rate to investigate the influence of mark-off rate on NDCG@50, which are shown in Section V-D.

D. Experimental Results

1) Impact of User's Perspective: In order to validate the impact of the users perspective on exploiting content information for event recommendation, we redesigned CTR, CDL and ConvMF into user-oriented models, and named them as CTR-U, CDL-U and ConvMF-U, respectively. To be precise, taking CTR for example, the key property of CTR lies in that the item latent vector is generated according to the topic proportion of item documents, while a user-oriented CTR applies this technique to the user latent vector. That is, the user latent vector is generated according to the topic proportion of user

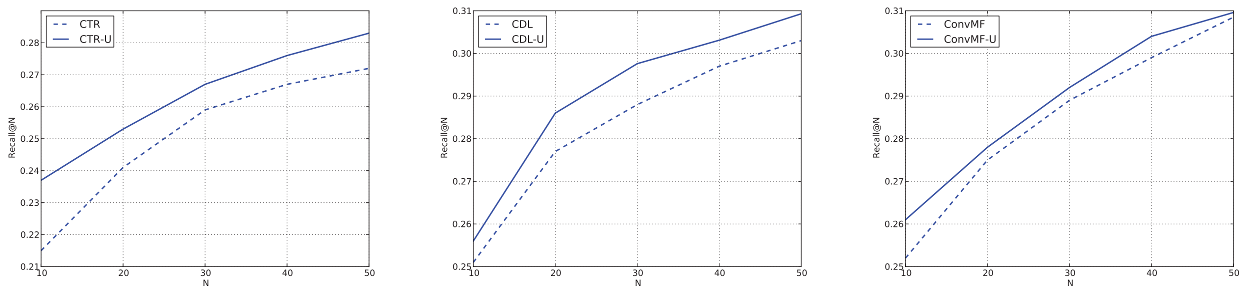


Fig. 4. Comparison of Recall@N of CTR and CTR-U, CDL and CDL-U, and ConvMF and ConvMF-U with different values of N

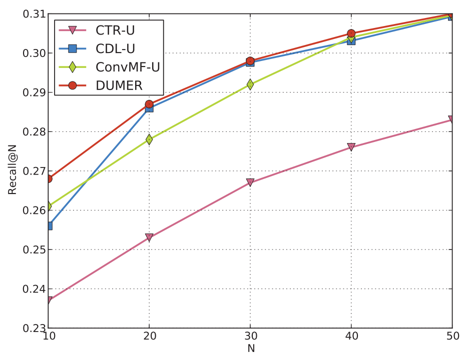


Fig. 5. Recall@N comparison of CTR-U, CDL-U and ConvMF-U with DUMER

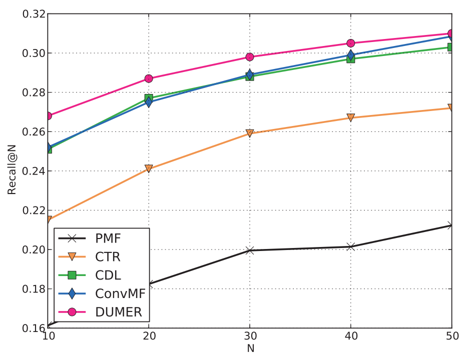


Fig. 6. Recall@N comparison of CTR, CDL and ConvMF with DUMER

documents we created. As for CDL and ConvMF, we did the same conversion.

Figure 4 shows the comparison of Recall@N of CTR and CTR-U, CDL and CDL-U, and ConvMF and ConvMF-U with different values of N. As shown in Figure 4(a), we can see that compared to CTR, CTR-U achieves better recall performance over all N values. The same observation can be drawn from Figure 4(b) for CDL and CDL-U, and Figure 4(c) for ConvMF and ConvMF-U. The results validate our intuition that modeling the user preference of content in a user's perspective is more suitable for event recommendation. The reasons are probably that users' interests play an important role in user's decision making, and exploiting content information in a user's perspective can better model users' interests, and thus is more suitable for event recommendation.

2) Model Comparison: Figure 5 shows the comparison of Recall@N of CTR-U, CDL-U and ConvMF-U with DUMER, in order to compare the learning ability of different models. As shown in Figure 5, DUMER outperforms all the other algorithms. Specifically, compared to ConvMF-U, DUMER achieves better recall results, especially when N is small, which indicates that DUMER's CNN architecture is better at learning high-level features of the content and thus improves the performance of event recommendation. The performance of CDL-U and DUMER are close when N is large, but DUMER still outperforms CDL-U under different values of N. This proves that CNN can better capture effective feature representations of content for event recommendation than the stacked denoising autoencoder model. In addition, another advantage of CNN structure is that the convolutional operation rapidly reduces the amount of network parameters during the training process. Hence, it takes less time and computations to train a DUMER model than a CDL model. CTR-U performs poorly because it utilizes a LDA model to exploit content information, which suffers from lacking of word orders and semantic meanings of words.

3) Overall Performance: Figure 6 shows the comparison of recall@N between PMF, CTR, CDL, ConvMF and DUMER. We can see that DUMER has the best recall results among all the models under different values of N. In particular, PMF has the worst performance among all models, while the performance will be greatly improved by combining it with CNN model in DUMER. The results demonstrate that compared with the state-of-the-art, DUMER, which combines the advantages of users' perspective and CNN learning ability, can deeply capture the semantic meaning of user interests and realize event recommendation more accurately.

Table II shows the RMSE results of DUMER and the compared methods under different ratios of training, validation and test set. The higher ratio of training set to the total dataset, the more samples to be trained for the algorithms.

As shown in Table II, we can observe that DUMER almost achieves the smallest RMSE under different ratios of training,

TABLE II
RMSE RESULTS UNDER DIFFERENT RATIOS OF TRAINING, VALIDATION AND TEST SET

Model	Ratios of training, validation and test set								
	(0.8, 0.1, 0.1)			(0.6, 0.2, 0.2)			(0.4, 0.3, 0.3)		
PMF	0.95734	1.1002	0.97761	0.95958	0.96451	0.99454	0.95928	0.96154	1.0415
CTR	0.35063	0.40234	0.40122	0.36512	0.41252	0.41085	0.38235	0.41658	0.42395
CDL	0.23948	0.29994	0.31379	0.24001	0.31932	0.31824	0.25147	0.32010	0.33029
ConvMF	0.20003	**0.27219**	0.30914	0.21217	0.30192	0.33426	0.22018	0.31283	0.31406
DUMER−	0.20156	0.28719	0.31649	0.20095	0.29425	0.32154	0.20085	0.29812	0.31410
DUMER	**0.18541**	0.27983	**0.29021**	**0.18487**	**0.27994**	**0.29049**	**0.18491**	**0.28055**	**0.29104**

validation and test set. In particular, DUMER achieves significant improvements on RMSE compared to PMF on all the training, validation and test sets under different ratios, e.g., the RMSE on the training set is reduced from 0.95734 for PMF to 0.18541 for DUMER, and the RMSE on the test set is reduced from 0.97761 for PMF to 0.29021 for DUMER when the ratio of the training, validation and test set is (0.8, 0.1, 0.1). This is because we utilized rich contextual features learnt from CNN to form user latent factor. Moreover, the results of DUMER are relatively insensitive to the change of the ratio of training, validation and test set. For example, the RMSE of the test set varies from 0.29021 to 0.29104 when the ratio changes from (0.8, 0.1, 0.1) to (0.4, 0.3, 0.3). That is, DUMER can achieve good performance even with a small set of training samples, which also implies that DUMER can find interesting events for users accurately even when the users have a short event history. To focus more specifically on DUMER, the results of DUMER are consistent over training set, validation set and test set, which indicates that DUMER has good generalization ability and does not suffer from over-fitting.

TABLE III
NDCG@50 RESULTS UNDER DIFFERENT MARK-OFF RATES

Model	Mark-off Rate		
	0.2	0.4	0.6
PMF	0.07094	0.06150	0.06748
CTR	0.13140	0.11392	0.10287
CDL	0.18977	0.16576	0.15214
ConvMF	0.20637	0.17239	0.16199
DUMER−	0.18450	0.12846	0.10924
DUMER	**0.21248**	**0.18015**	**0.17000**

Table III shows the NDCG@50 results of DUMER and compared methods under different mark-off rates. The mark-off rate stands for the percentage of the total dataset that the test set accounts for. Therefore, the higher the mark-off rate, the less the training set to be trained for the algorithms.

As we mentioned, the higher the NDCG@50, the better recommendation performance. As shown in Table III, compared with other algorithms, the proposed DUMER achieves the highest recommendation performance under all different mark-off rates. We can observe that the recommendation performance of these algorithms decreases as the mark-off rate increases since that less data used for training will decrease the learning accuracy in these algorithms. We can also observe that DUMER achieves significant recommendation improvement compared to PMF under different mark-off rates, e.g., NDCG@50 improves from 0.07094 for PMF to 0.21248 for DUMER. This implies that CNN can deeply capture the contextual information of users' interested event to characterize the preference of users on events which is taken into PMF model to improve the recommendation accuracy. The phenomenon of NDCG@50 results are consistent with that of RMSE results, implying that it is feasible to utilize attendance matrix approximation for event recommendation. We can conclude that content information indeed plays an important role in users' event decision makings and utilizing CNN to deeply exploit content from users' perspective can be beneficial to event recommendation.

4) Impact of Word Embedding Model: We compare the performance of our model with one-hot vector and pre-trained word embedding vector from GloVe to investigate the impact of pre-trained word embedding model for event recommendation. As shown in Table II and Table III, DUMER outperforms DUMER− under all scenarios. The RMSE of DUMER are always smaller than that of DUMER−, and the recommendation accuracy of DUMER is better than that of DUMER−. This validates our argument that bag-of-word model fails to capture the word relations and cannot fully capture the contextual information of the content of events, while word embedding model can realize that purpose and help improve event recommendation accuracy.

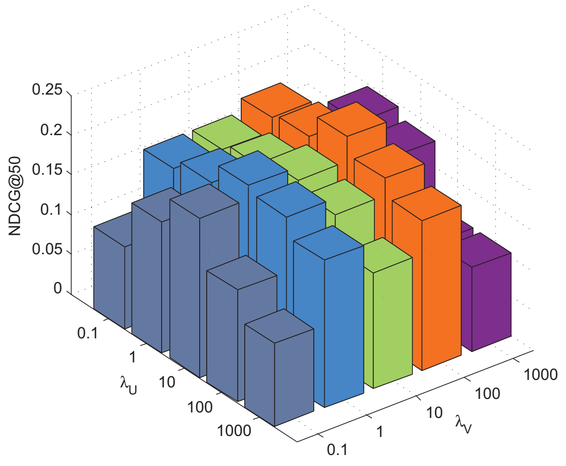


Fig. 7. Parameter Analysis

5) Parameter Analysis: To investigate the influence of the parameters on the performance of our algorithm, we conducted a parameter analysis experiment on hyper parameters λ_U and λ_V. Figure 7 shows the NDCG@N results of the test set under different values of λ_U and λ_V when the mark-off rate is 20%. Note that a high value of λ_U implies that the user latent factor is accounted for more loss in the total loss function. We can see that as λ_U increases, the result becomes better, while the performance deteriorates when λ_U gets too high. It is likely that it would have a bad influence on the training process when you put too much emphasis on the CNN architecture of learning user latent factor. On the other hand, the results fluctuate as λ_V increases. This implies that the effect of λ_V on event recommendation accuracy is not stable due to the randomization. In particular, the best recommendation performance is achieved when $\lambda_U = 10$ and $\lambda_V = 100$.

VI. Conclusion and Future Work

In this paper, we investigated how to deeply exploit the content of events for event recommendation in EBSNs. We proposed DUMER to characterize the latent preference of users from users' perspective by deeply exploiting the contextual information of events that users have attended. In particular, we use CNN with word embedding to deeply capture the contextual information of users' interested events and build up a user latent model for each user. Then we incorporate the user latent factor into PMF model to enhance the recommendation accuracy. As far as we know, this is the first work that utilizes deep learning techniques for event recommendation in EBSNs. The extensive experimental results conducted on real-world data shows that our proposed model did capture the subtle contextual information of contents and improve the recommendation accuracy compared with exiting content-based event recommendation algorithms.

This work focused on exploiting the content of events for event recommendation in EBSNs. Besides the content, an event also have many other useful information, such as the time, location and event host, that could be exploited to improve recommendation accuracy. Therefore, as for the future work, we plan to integrate all these contextual information together to improve recommendation accuracy.

Acknowledgement

This work was supported in part by National Natural Science of China (Grants No. 61502352, 61772551 and 61572106), National Basic Research Program of China (Grant No. 2014CB340600), NSFC-General Technology Foundation Research United Fund (Grants No. U1536204, U1636101 and U1736211), National High-tech Research and Development Program (Grants No. 2015AA016002 and 2015AA016004), Natural Science Foundation of Hubei Province and Jiangsu Province (Grants No. 2017CFB503, 2017CFA007, 2017C-FA047 and BK20150383), Fundamental Research Funds for the Central Universities (Grant No. 413000035), and Jiangsu Key Laboratory of Big Data Security & Intelligent Processing, NJUPT (Grant No. BDSIP1805).

References

[1] Meetup, 2017, http://www.meetup.com/.

[2] Eventbrite, 2017, http://www.eventbrite.com/.

[3] Netflix, 2017, https://www.netflix.com/.

[4] Z. Wang, J. Liao, Q. Cao, H. Qi, and Z. Wang, "Friendbook: a semantic-based friend recommendation system for social networks," *IEEE Transactions on Mobile Computing*, vol. 14, no. 3, pp. 538–551, 2015.

[5] W. Zhang, J. Wang, and W. Feng, "Combining latent factor model with location features for event-based group recommendation," in *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2013, pp. 910–918.

[6] Z. Qiao, P. Zhang, Y. Cao, C. Zhou, L. Guo, and B. Fang, "Combining heterogenous social and geographical information for event recommendation," in *Proceedings of Twenty-Eighth AAAI Conference on Artificial Intelligence*, 2014.

[7] A. Q. Macedo, L. B. Marinho, and R. L. Santos, "Context-aware event recommendation in event-based social networks," in *Proceedings of the 9th ACM Conference on Recommender Systems*, 2015, pp. 123–130.

[8] W. Zhang and J. Wang, "A collective bayesian poisson factorization model for cold-start local event recommendation," in *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2015, pp. 1455–1464.

[9] L. Gao, J. Wu, Z. Qiao, C. Zhou, H. Yang, and Y. Hu, "Collaborative social group influence for event recommendation," in *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*. ACM, 2016, pp. 1941–1944.

[10] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent dirichlet allocation," *Journal of machine Learning research*, vol. 3, no. Jan, pp. 993–1022, 2003.

[11] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[12] N. Kalchbrenner, E. Grefenstette, and P. Blunsom, "A convolutional neural network for modelling sentences," *arXiv preprint arXiv:1404.2188*, 2014.

[13] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa, "Natural language processing (almost) from scratch," *Journal of Machine Learning Research*, vol. 12, no. Aug, pp. 2493–2537, 2011.

[14] D. Tang, B. Qin, T. Liu, and Y. Yang, "User modeling with neural network for review rating prediction," in *Proceedings of the 24th International Conference on Artificial Intelligence*. AAAI Press, 2015, pp. 1340–1346.

[15] D. Kim, C. Park, J. Oh, S. Lee, and H. Yu, "Convolutional matrix factorization for document context-aware recommendation," in *The ACM Conference*, 2016, pp. 233–240.

[16] X. Liu, Q. He, Y. Tian, W.-C. Lee, J. McPherson, and J. Han, "Event-based social networks: linking the online and offline social worlds," in *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2012, pp. 1032–1040.

[17] R. Du, Z. Yu, T. Mei, Z. Wang, Z. Wang, and B. Guo, "Predicting activity attendance in event-based social networks: Content, context and social influence," in *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, 2014, pp. 425–434.

[18] Z. Wang, Y. Zhang, Y. Li, Q. Wang, and F. Xia, "Exploiting social influence for context-aware event recommendation in event-based social networks," in *INFOCOM 2017-IEEE Conference on Computer Communications, IEEE*. IEEE, 2017, pp. 1–9.

[19] H. Wang, N. Wang, and D. Y. Yeung, "Collaborative deep learning for recommender systems," in *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2014, pp. 1235–1244.

[20] J. Pennington, R. Socher, and C. D. Manning, "Glove: Global vectors for word representation." in *EMNLP*, vol. 14, 2014, pp. 1532–1543.

[21] Y. Shen, X. He, J. Gao, L. Deng, and G. Mesnil, "A latent semantic model with convolutional-pooling structure for information retrieval," in *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management*. ACM, 2014, pp. 101–110.

[22] Y.-L. Boureau, N. Le Roux, F. Bach, J. Ponce, and Y. LeCun, "Ask the locals: multi-way local pooling for image recognition," in *Computer Vision (ICCV), 2011 IEEE International Conference on*. IEEE, 2011, pp. 2651–2658.

[23] R. Salakhutdinov and A. Mnih, "Probabilistic matrix factorization." in *Nips*, vol. 1, no. 1, 2007, pp. 2–1.

[24] C. Wang and D. M. Blei, "Collaborative topic modeling for recommending scientific articles," in *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2011, pp. 448–456.