Information Sciences 471 (2019) 269–287

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

Multiobjective e-commerce recommendations based on hypergraph ranking

Mingsong Mao[a], Jie Lu[b,*], Jialin Han[a], Guangquan Zhang[b]

[a] School of Information Technology, Jiangxi University of Finance and Economics, Nanchang, China
[b] Faculty of Engineering and IT, Center for AI, University of Technology Sydney, Australia

ARTICLE INFO

Article history:
Received 26 September 2017
Revised 7 July 2018
Accepted 11 July 2018
Available online 16 August 2018

Keywords:
Recommender systems
E-commerce
User personalization
Hypergraph

ABSTRACT

Recommender systems are emerging in e-commerce as important promotion tools to assist customers to discover potentially interesting items. Currently, most of these are single-objective and search for items that fit the overall preference of a particular user. In real applications, such as restaurant recommendations, however, users often have multiple objectives such as group preferences and restaurant ambiance. This paper highlights the need for multi-objective recommendations and provides a solution using hypergraph ranking. A general User–Item–Attribute–Context data model is proposed to summarize different information resources and high-order relationships for the construction of a multipartite hypergraph. This study develops an improved balanced hypergraph ranking method to rank different types of objects in hypergraph data. An overall framework is then proposed as a guideline for the implementation of multi-objective recommender systems. Empirical experiments are conducted with the dataset from a review site Yelp.com, and the outcomes demonstrate that the proposed model performs very well for multi-objective recommendations. The experiments also demonstrate that this framework is still compatible for traditional single-objective recommendations and can improve accuracy significantly. In conclusion, the proposed multi-objective recommendation framework is able to handle complex and changing demands for e-commerce customers.

1. Introduction

Today, the volume of data on an e-commerce site may become extremely large. For example, Taobao.com announced that there are currently over 0.8 billion daily-active products on this C2C e-commerce site. Hence, it is challenging customers to choose items of interest from the large range of options. To alleviate this information-overload problem, recommender systems have emerged as useful tools to learn the personalized preferences of users and make personalized suggestions. A large number of recommendation applications have reportedly been deployed in specific areas [3,25], and they are thought to enhance e-commerce sales by encouraging potential buyers, increasing cross-selling and building customer loyalty [48,55].

Traditional recommender systems are single-objective: to find those items best fitting an individual user's overall preference refined from historical data. In other words, a recommender system assumes users' preferences are static and they do not have additional requirements from time to time. However, it is common for users to change their preferences fre-

[*] Corresponding author.
E-mail addresses: mingsongmao@jxufe.edu.cn (M. Mao), jie.lu@uts.edu.au (J. Lu), jialinhan@jxufe.edu.cn (J. Han), guangquan.zhang@uts.edu.au (G. Zhang).

https://doi.org/10.1016/j.ins.2018.07.029
0020-0255/© 2018 Published by Elsevier Inc.

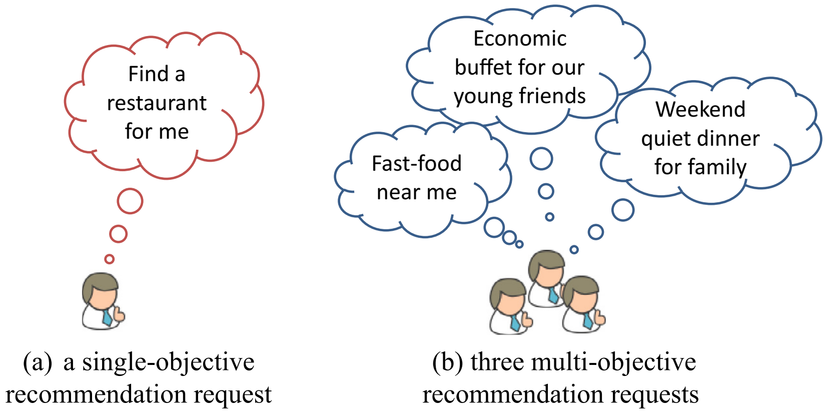
Find a restaurant for me
Economic buffet for our young friends
Weekend quiet dinner for family
Fast-food near me
(a) a single-objective recommendation request
(b) three multi-objective recommendation requests

Fig. 1. Single-objective and multi-objective recommendation requests for restaurants.

quently in e-commerce environments. For example, a customer will not always go to the same restaurant even though it is the “best-fit” for his/her overall preference. Instead, the choice at a particular time is impacted by different factors such as different companions and time-location context, hence, the customer requires a restaurant with specific conditions to satisfy multiple objectives. Thus, the recommendation request becomes multi-objective rather than single-objective. It is worth mentioning that the existing context-aware recommender systems also consider a user’s different preferences in specific situations, such as time and location, but they do not take into account a user’s explicit requirements on item conditions. Fig. 1 shows a simple comparison of single-objective and multi-objective recommendation problems. Fig.1(a) is a traditional single-objective recommender system, which is only, concerned a user’s preference for a restaurant based on a particular type of cuisine. As a result, the suggestions will be the same each time. Fig.1(b) indicates that multi-objective requests, however, are related to extra constraints such as the type of cuisine and restaurant ambiance at different times. Therefore, this study is motivated to propose a multi-objective recommendation framework to analyze the changing and flexible requirements as input and suggest appropriate items as output to satisfy the multi-objective requirements of users.

Another significant challenge for modern e-commerce recommendations is that massive information entities and complex relationships may be involved with the development of Web2.0 applications. In the early stage of recommender system development, the explicit rating data is the major or only input for collaborating filtering (CF) [38]. Today, we can extract more and different types of information in addition to user ratings from an e-commerce site to improve recommendation quality, including item content [26], user social connections [28], textual tags or comments [14,30], environmental context [12], etc. Although these resources have been well studied separately in recent advanced CF models [38], there are few models which integrate all the involved information resources in one generalized model. Motivated by several pioneer studies of graph models [4,8,10,11,16,27,41], this paper introduces hypergraph to handle the possible information entities and complex relationships in e-commerce. Graph models generally consider information entities such as users and items as separate nodes and build connections for these nodes as edges, by which we can handle the recommendation problem as a ranking problem on graph data: to seek the nearest item nodes for a given user node. It is easy to extend a graph by adding new nodes or edges. As a result, some advanced graph structures have been utilized in the field of recommender systems, such as multi-partite graph [11], multigraph [27] and hypergraph [41]. In particular, the hypergraph is the most generalized graph structure that can theoretically handle any types of information entities and high-order relationships. For example, a hypergraph can model the multiple connections between a same pair of nodes or a group connection of multiple nodes.

The hypergraph model is also suitable for multi-objective recommendations. Graph ranking-based models usually transfer the recommendation problem to one that seeks a relative ranking order of item entities for a particular user entity on a graph structure. We also treat the multi-objective recommendation problem as one that ranks item entities according to the overall closeness to a group of entities, which may consist of multiple users and contextual entities. The main contributions of this study are three-fold:

First, we propose a unified User–Item–Attribute–Context (UIAC) data model as a basis for the collection of valuable data sources for e-commerce recommendations. In this model, we summarize four types of information entities and six types of pairwise or group relationships that may appear in the e-commerce environment. The four types of entities are user, item, attribute and context entities. The six types of relationships are user–user, item–item, User–Item, Item–Attribute, user–attribute and user–context–item relationships. This model includes the most input resources adopted in existing recommender systems, such as social networks, context-aware preference, tag information, etc. In particular, we extend the context entities to all third-party information including textual comments and tags as well as traditional environmental information like time and location.

Secondly, we propose a multi-objective recommendation framework to handle the flexible demands of e-commerce customers. We point out that a user in an e-commerce application does not always want to find the same items that best fit the overall preference, but often has extra requirements or constraints on item conditions or contextual information. With the proposed UIAC data model, we can decompose a multi-objective recommendation to requirements to multiple entities from user, item, attribute and context entities. Putting all groups of entities and pairwise or group relationships together, a most generalized graph structure, the multipartite hypergraph, is constructed. Then, we transfer the multi-objective recommendation problem to a ranking problem on a multipartite hypergraph structure. Finally, we develop a complete framework as a guideline to illustrate how multi-objective recommendations are queried by users, addressed using hypergraph ranking, and returned to the users.

Thirdly, we propose a modified hypergraph ranking model called balanced hypergraph ranking (BHR) to improve the ranking performance for the unique multipartite hypergraph built from the UIAC data model. Different to common hypergraphs, the multipartite hypergraph is constructed from a wide range of relationships in an e-commerce application and is special in that the edge degrees (number of connected nodes) vary greatly from two (pairwise edges) to thousands (group edges) or even larger. In this circumstance, we find that the traditional hypergraph ranking models suffer from a ranking bias problem. The modified BHR is able to alleviate this problem to some extent and is more suitable for the proposed unique multi-objective e-commerce recommendations.

The rest of this paper is organized as follows. Section 2 is a brief review on e-commerce recommender systems and the existing related works in the fields of multi-criteria, group, context-aware recommendations. In particular, the developments in graph models for recommendations are introduced. Section 3 is a preliminary study about the hypergraph and hypergraph ranking theory. In Section 4, we propose a UIAC data model to discuss which kinds of information entities and relationships may appear in e-commerce, and how to construct a multipartite hypergraph structure. We also modify the traditional hypergraph ranking method and propose a balanced hypergraph ranking method for the unique multipartite hypergraph. Section 5 presents a multi-objective recommendation framework based on the hypergraph and ranking model. In Section 6, we conduct an analysis using the data on a business review site Yelp.com. Our model is compared to the existing graph models for both single-objective and multi-objective recommendations and is demonstrated to perform well. The last section concludes the findings of this study and suggests future research directions.

2. Related works

This paper is motivated to develop a novel multi-objective framework for e-commerce recommender systems. It is partially related to existing multi-criteria recommendations, group recommendations and context-aware recommendations. We review the related works in these fields in the following sub-sections to analyze the connections and differences between our study and previous studies.

2.1. E-commerce recommender systems

Recommender systems have been deployed successfully in many domains including information retrieval, tourism review, e-learning, e-government and e-commerce [25]. In the field of e-commerce, recommender systems help individual customers to find potentially interesting items by learning personalized preferences from historical data including ratings, browsing and clicking records and so on [50]. Many of the largest commerce sites like Amazon and Taobao already use recommender systems to help their customers find products to purchase [15,36,37]. It has been reported that recommender systems are able to enhance e-commerce sales by encouraging potential buyers, increasing cross selling and building customer loyalty [48,55].

From the perspective of techniques, recommendation models are usually classified as content-based (CB) [32], collaborating filtering (CF) [38] and hybrid models that are combinations of the former two [18]. The key idea of CB is to find items that have similar content or attributes to the items previously chosen by the target user. The CF takes a different approach and extracts user preferences from user ratings that record explicit preferences for known items, and it can be further divided into memory-based and model-based [38]. Memory-based CF predicts user preferences to unknown items by integrating the ratings of neighbor users who share similar preferences [35]. Model-based CF approaches, on the other hand, are based on unique prediction models in which some parameters have been trained with previous rating data as the input. Examples of model-based CF include probabilistic topic models [12], fuzzy models [42], graph models [11], Bayesian network models [23], matrix factorization models [21], etc.

From the perspective of input resources, e-commerce recommendation studies have been widely extended in recent years with the development of the Internet and Web2.0 applications. For example, the online social relations of users have become another important facet of information to enhance pure rating-based CF models [28]. More various user-contributed data like textual reviews [30], social tags [14], social media [51] and more generalized side information [33] are also imported in e-commerce recommendations. Contextual information like temporal, special and weather information associated with user preferences has also motivated many studies in this field recently [9,34]. For systems where it is difficult to collect rating data, knowledge-based recommender systems [6] have also been developed to build functional knowledge rules to describe how items meet the needs of users [5,22].

This review indicates that e-commerce recommender systems have attracted many studies in recent years. The recent developments show that one of the most urgent tasks in this field is to develop new recommendation models that are able to integrate various types of heterogeneous information to enhance performance.

2.2. Multi-criteria and group recommendations

Our work for multi-objective recommendations is also related to multi-criteria and group recommendations to some extent. Generally, multi-criteria recommender systems allow users to issue multi-criteria ratings to evaluate items from a variety of perspectives, while acknowledging that the suitability of a recommended item for a particular user will probably depend on more than one utility-related criterion [17]. The additional information provided by multi-criteria ratings can represent more complex preferences of users and can thus help to improve recommendation quality. Multi-criteria recommendations mainly focus on the utilization of user preferences for different aspects of items but still aim to build the static preferences of users; therefore, they are still single-objective recommendations.

Group recommender systems (GRS) produce suggestions for a group of users when group members are unable to gather for negotiation, or their preferences are not clear in spite of having met each other [7,47]. Group recommender systems are able to generate corresponding suggestions for multiple participant users by combining their preferences based on certain strategies like least misery, average, most pleasure and so on [29]. Group recommender systems are a kind of multi-objective recommender systems because they have to satisfy the preferences of all members rather than a single user. However, existing GRSs are not able to handle the extra demands of group members for item conditions or context.

2.3. Context-aware recommendations

This paper considers the changing user demands in different circumstances, which is partially related to existing context-aware recommender system (CARS) studies [2,45]. One of the most cited definitions of context is the definition of Dey et al. [9] that describes context as any information that can be used to characterize the situation of an information entity in an application. In the review of Adomavicius et al. [1], context in the recommender system field is a multifaceted concept used across various disciplines, with each discipline adopting a certain angle and putting its “stamp” on this concept. With context awareness, the rating (preference) function is no longer a two-dimensional function (R: $User \times Item \rightarrow Rating$) but becomes a multi-dimensional function (R: $User \times Item \times Contex \rightarrow Rating$), where *User* and *Item* are the domains of users and items respectively, and *Context* specifies the domain of other entities associated with users and items. Thus, context-aware recommender systems consider users’ unique preferences in specific environments such as time, location and weather. This is particularly important for some applications in which it is not sufficient to consider only users and items, such as recommendations for tourism [20,31], vacation package [40], or social events [13]. For example, using a temporal context, a travel recommender system might make very different vacation suggestions in winter compared to summer [40]. The contextual information about students in technology enhanced learning environments has also been incorporated into recent e-learning recommendations [44]. From the perspective of techniques, different advanced models have been introduced in dealing the complex relationships between user, item and context entities. Some studies consider context information like “time” and “companion” as extra labels associated with user ratings, and propose a multi-label classification model to tackle the CARS problem [20,52]. Unger et al. applied neural network learning models to learn the “latent” context information from multi-sensor data for recommendations [43]. Enhanced matrix and tensor factorization models have also been introduced to address CARS, such as the hierarchical factorization machine proposed in [46] and the modified tensor model in [49]. Moreover, graph models in CARS have been developed to ease extending different entities as graph nodes and relationships as edges. Because graph models are the basis of our study, a detailed review is given in the following sub-section.

The multi-objective recommendations in our work share similarities with existing CARS by taking into account changing user demands in different contexts. The difference is that the user requirements in different circumstances of CARS is implicitly refined from data and still do not change frequently, but our study also allow users to state explicit requirements for items even in similar situations. For example, a traditional CARS may suggest the same restaurant to customers in the context of “friday evening” by learning the time-aware preferences of users. However, a multi-objective recommender system will also allow users to specify the “topic” of the activity, such as “birthday party”, so more targeted suggestions will be generated. In summary, our work can be seen as an extension of existing context-aware recommendations that consider both explicit and implicit user demands in different circumstances.

2.4. Graph models for recommendations

The goal of a recommender system is to seek an appropriate ranking order for all the candidate items and then select the top listed ones as suggestions to users. If we place users and items as nodes into a graph, then the recommendation problem can be transferred to a ranking problem on graph data. In the early stage, Fouss et al. introduced random walk theory to rank item nodes on a User–Item bipartite graph to generate recommendations [10]. In their model, user ratings are treated as the weighted edges between user nodes and item nodes to perform random walking. Jamali and Ester combined user-user trust relationships and User–Item ratings, and proposed a unique random walk manner on the bipartite graph structure [16]. Moreover, more types of information entities were imported as extra vertices and multipartite graph models were proposed by researchers [4,8] and random walks jump between different types of vertices with different relationships. There are also some advanced graph structures being introduced to deal with high-order relationships. For example, to handle multiple relations between the same pair of nodes, our previous work proposed a multigraph ranking model for multi-relational social recommendations [27]. To handle group relationships between multiple nodes such as user-tag-item connections, Tan et al. introduced a hypergraph and ranking model for music sharing recommendations [41]. In summary, graph models are advantageous to ease modeling and extending. Hence, this study is motivated to introduce a hypergraph model to incorporate different information entities and to handle multi-objective recommendations.

3. Preliminary study of hypergraph ranking

Before constructing our multi-objective recommendation framework using hypergraphs, it is important to introduce the traditional definitions and ranking methods for hypergraphs, and the possible limitations in dealing with our research problem, as follows.

3.1. Hypergraph

An ordinary graph is a representation of a set of vertices in which each edge connects a pair of nodes, while a hypergraph is a generalization of an ordinary graph with hyperedges, a special type of edges that connects an arbitrary number of vertices [53]. In other words, a hyperedge represents a high-order relationship of two or more vertices. The formal definition of a hypergraph is given as follows:

Definition 1. (Hypergraph). A hypergraph $G = (V, E)$ is a pair where $V = \{v_1, v_2, \ldots\}$ is the vertex set representing a finite number of objects, and $E = \{e_1, e_2, \ldots\}$ is the hyperedge set representing the high-order relationships between vertices. A hyperedge $e \in E$ is represented by the connected nodes, i.e., a non-empty subset of V, to which a weighting function $w : E \to \mathbb{R}^+$ is also assigned to denote the strength of this connection.

Let $h(v, e) = 1$ denote that a vertex v is in a hyperedge e and $h(v, e) = 0$ otherwise. We then obtain an incidence matrix $\mathbf{H}$ with size $|V| \times |E|$ in which each element is:

$$H_{ij} = h(v_i, e_j) = \begin{cases} 1, & \text{if } v_i \in e_j \\ 0, & \text{otherwise} \end{cases} \tag{1}$$

The degree of a vertex is defined as $d(v) = \sum_e w(e)h(v, e)$. Unlike the edge in a simple graph, the degree of a hyperedge is defined as the number of connected vertices, i.e., $\delta(e) = \sum_v h(v, e)$. Thus, an ordinary graph is thought of a special case of hypergraph in which the edge degrees are always two. Throughout the rest of this paper, we define the following diagonal matrix forms for $w(e)$, $\delta(e)$ and $d(v)$ as $\mathbf{W} \in \mathbb{R}^{|E|\times|E|}$ $\mathbf{D}_e \in \mathbb{Z}^{|E|\times|E|}$ $\mathbf{D}_v \in \mathbb{R}^{|V|\times|V|}$, respectively.

3.2. Regularization framework for hypergraph ranking

Ranking on graph data is a unique problem to sort most vertices based on the known labels of a small part of vertices as limited given knowledge. This problem is usually given by a weighted graph $G = (V, E)$, together with a *query vector* $\mathbf{y} = [y_1, y_2, \ldots, y_{|V|}]^T$, which denotes the input ranking sores of a small number of vertices. The objective is to seek a ranking function $f : V \to \mathbb{R}$ to label all vertices. There are two constraints for a good ranking. First, it should be “smooth” on the graph, meaning the ranking scores will not vary greatly for highly related nodes. Second, it should be close to the input query vector $\mathbf{y}$ such that the given knowledge is well accepted. The graph-ranking problem is therefore usually formalized to minimize the following cost function:

$$\min_{f: V\to\mathbb{R}} Q(f) = S(f) + \mu\hat{R}(f; y) \tag{2}$$

In the above equation, $\mu > 0$ is a trade-off parameter; the term $S(f)$ measures the smoothness of f; the term $\hat{R}(f; y)$ measures the empirical error of f, which is usually represented by the ℓ_2 norm variance [32,33]: $\hat{R}(f; y) = \|f - y\|^2 = (\mathbf{f} - \mathbf{y})^T(\mathbf{f} - \mathbf{y})$. Note that $\mathbf{f}$ is the vector form of the ranking function f, and we use them interchangeably in our paper.

Zhou et al. proposed the regularization framework for ordinary graph ranking and also extend it to hypergraph environments to rank objects with group relationships [53,54]. In their work, the smoothness cost for hypergraph ranking function f is calculated by:

$$\begin{aligned} S(f) &= \frac{1}{2}\sum_{i,j=1}^{|V|}\sum_{e\in E}\frac{w(e)h(v_i, e)h(v_j, e)}{\delta(e)}\left\|\frac{f_i}{\sqrt{d(v_i)}} - \frac{f_j}{\sqrt{d(v_j)}}\right\|^2 \\ &= \sum_{i,j=1}^{|V|}\sum_{e\in E}\frac{w(e)h(v_i, e)h(v_j, e)}{\delta(e)}\left(\frac{f_i^2}{d(v_i)} - \frac{f_i f_j}{\sqrt{d(v_i)}\sqrt{d(v_j)}}\right) \\ &= \sum_{i=1}^{|V|} f_i^2 \sum_{e\in E}\frac{w(e)h(v_i, e)}{d(v_i)}\sum_{j=1}^{|V|}\frac{h(v_j, e)}{\delta(e)} - \sum_{i,j=1}^{|V|}\sum_{e\in E}\frac{f_i w(e)h(v_i, e)h(v_j, e) f_j}{\sqrt{d(v_i)}\delta(e)\sqrt{d(v_j)}} \\ &= \mathbf{f}^T(\mathbf{I} - \mathbf{D}_v^{-1/2}\mathbf{H}\mathbf{W}\mathbf{D}_e^{-1}\mathbf{H}^T\mathbf{D}_v^{-1/2})\mathbf{f} = \mathbf{f}^T(\mathbf{I} - \mathbf{A})\mathbf{f} \end{aligned} \tag{3}$$

In the above expansions, an intermediate matrix $\mathbf{A} = \mathbf{D}_v^{-1/2}\mathbf{H}\mathbf{W}\mathbf{D}_e^{-1}\mathbf{H}^T\mathbf{D}_v^{-1/2}$ is defined. Taking (3) into (2), the overall cost function is rewritten:

$$\min_{\mathbf{f}} Q(\mathbf{f}) = \mathbf{f}^T(\mathbf{I}-\mathbf{A})\mathbf{f} + \mu(\mathbf{f}-\mathbf{y})^T(\mathbf{f}-\mathbf{y}) \tag{4}$$

Requiring that the gradient of $Q(\mathbf{f})$ vanish will give:

$$\left.\frac{\partial Q}{\partial \mathbf{f}}\right|_{\mathbf{f}=\mathbf{f}^*} = (\mathbf{I}-\mathbf{A})\mathbf{f}^* + \mu(\mathbf{f}^*-\mathbf{y}) = 0 \tag{5}$$

With algebraic steps, then we have the final optimal ranking result as

$$\begin{aligned}\mathbf{f}^*_{\mathrm{REG}} &= \frac{\mu}{\mu+1}\left(\mathbf{I} - \frac{1}{\mu+1}\mathbf{A}\right)^{-1}\mathbf{y} \\ &\triangleq (\mathbf{I}-\alpha\mathbf{A})^{-1}\mathbf{y}\end{aligned} \tag{6}$$

In the above equation, a new parameter $\alpha = 1/(\mu+1) \in (0,1)$ is imported and the positive constant $\mu/(\mu+1)$ is omitted because it does not affect the ranking order.

3.3. Random walks on hypergraph

The graph-ranking problem can be solved in a different way by performing random walks on the graph with appropriate starting points. At the beginning, a starting point is selected randomly from the given labeled vertices according to the input query vector $\mathbf{y}$. A walker is supposed to move randomly to adjacent nodes from the starting point following the edges (hyperedges) in the graph (hypergraph). Finally, the stationary visiting probabilities are the ranking scores for all vertices.

In traditional hypergraph random walk models [41,53], the transition probability from one node to an adjacent node is $p(v|u) = \sum_{u,v\in e} w(e)/d(u)\delta(e)$ and the overall transition matrix for the whole hypergraph will be $\mathbf{T} = \mathbf{D}_v^{-1}\mathbf{H}\mathbf{W}\mathbf{D}_e^{-1}\mathbf{H}^T$. Based on the random walk with restarts theory [19,24], the walker will have two options for the next move each time: (1) continually walking to an adjacent node with a certain probability $\alpha \in (0, 1)$; (2) jumping back to the starting point with probability $1-\alpha$.

Let $\mathbf{p}^{(t)}$ be a column vector reporting the visiting probability of every vertex at a certain time t, and $\mathbf{q}$ the initial probability of being selected as the starting point. The visiting distribution of the whole vertex set updates in the following way according to the random walk with restarts model:

$$\mathbf{p}^{(t+1)} = \alpha\mathbf{T}\mathbf{p}^{(t)} + (1-\alpha)\mathbf{q}. \tag{7}$$

Finally, the stationary distribution when (7) reaches convergence represents the long-term visiting rate of all vertices. Let $\mathbf{p}^{(t+1)} = \mathbf{p}^{(t)} = \mathbf{p}^{(\infty)}$ then we obtain $\mathbf{p}^{(\infty)} = (1-\alpha)(\mathbf{I}-\alpha\mathbf{T})^{-1}\mathbf{q}$. Because the positive constant $(1-\alpha)$ does not change the ranking order, the optimal ranking result will be:

$$\mathbf{f}^*_{\mathrm{RW}} = (\mathbf{I}-\alpha\mathbf{T})^{-1}\mathbf{q}. \tag{8}$$

It is easy to find that (8) has the same structure as (6), and the intermediate matrix $\mathbf{A}$ in the regularization framework is a normalized version of the transition matrix $\mathbf{T}$ in the random walk model. The regularization framework is thus seen to be a normalized solution of the random walk-based ranking model, and some empirical studies report that the former performs better for recommendation applications [41].

3.4. Ranking bias of traditional hypergraph ranking

We notice that the aforementioned traditional hypergraph ranking methods will produce a bias if the hyperedge degree varies significantly. Fig. 2 gives a numerical example and we use random walks to explain the ranking bias.

In Fig. 2, node A, which is supposed to be the starting point of random walks, is linked by two hyperedges e_1 and e_2. The edge strength indicates that the relationship between the nodes in e_1 is equivalent to the relationship between the nodes in e_2, so it is intuitively appropriate that the runner shall have equivalent probability to follow any one of the two hyperedges. With the settings of the traditional ranking in the above, however, the runner will strongly prefer to move via edge e_2. Recall the transition formula $p(v|u) = \sum_{u,v\in e} w(e)/d(u)\delta(e)$ and suppose there is no jumping back, the probability of moving to the nodes in the two hyperedge will be $p(n\in e_1) = \frac{1}{2\times 20} = \frac{1}{40}$ and $p(n\in e_2) = \frac{1}{2\times 2} = \frac{1}{4}$, respectively. We can find that the probability of moving to a node in e_2 is heavily overvalued only because the edge degree is much smaller. Hence, we consider the traditional hypergraph ranking models are not suitable for hypergraphs where the hyperedge degrees vary greatly. In the following section, we describe the building of a multipartite hypergraph model for e-commerce environments, in which the hyperedge consists of six types of pairwise or group relationships. It is our opinion that the traditional ranking models will not perform well in this circumstance, so we propose an improved balanced hypergraph ranking model in Section 4.3.

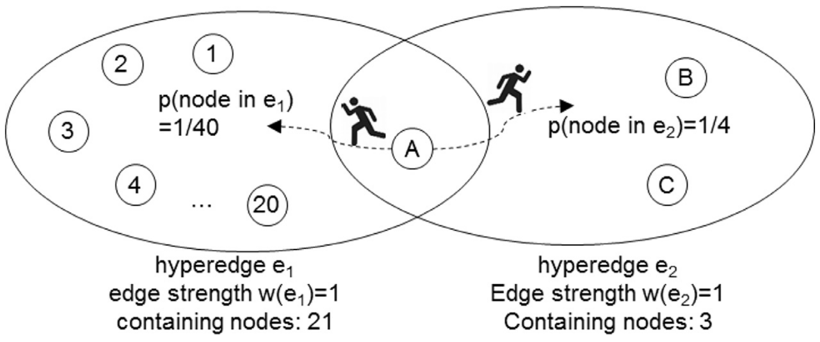

Fig. 2. Ranking bias of traditional hypergraph ranking.

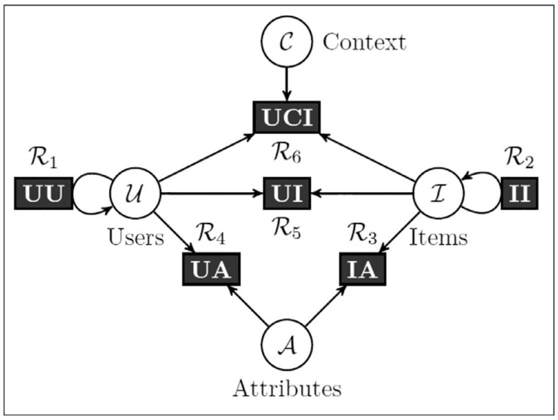

Fig. 3. User–Item–Attribute–Context (UIAC) data model.

4. Multipartite hypergraph construction and ranking

We use restaurant recommendations as an example for e-commerce recommender systems. In this scenario, information such as customer visiting history, restaurant attributes and contextual information is valuable for recommendation making. We propose a generic User–Item–Attribute–Context (UIAC) data model to summarize four types of information entities and six types of pairwise or high-order relationships, as shown in Fig. 3.

4.1. The UIAC data model

Taking a restaurant recommender system (RRS) as an example, the four types of information entities are users, items (restaurants), restaurant attributes and context information.

Users (**U**) are the registered customers and requesters in an RRS. Users are encouraged to review the restaurants they have visited using numerical ratings, as an explicit means of obtaining user preferences.

Items (**I**) are the restaurants included in an RRS as recommended objects. The ultimate goal of an RSS is to suggest a list of restaurants as alternative options to the users' requests.

Attributes (**A**) are a set of taxonomy texts describing the conditions of restaurants, such as restaurant types ("breakfast", "lunch", "dinner", etc.), restaurant ambience ("romantic", "intimate", "touristy", etc.), noise level ("quite" or "loud") and so on. Usually, an e-commerce platform will provide a standard taxonomy for the items and we can collect them directly. For example, Fig. 6 in Section 6 shows the settings of restaurant taxonomy attributes in Yelp.com, a business review application.

Context (**C**) refers to third-party information that may affect user choices [9]. Despite environmental information such as time and location, we also treat tags or short comments as context entities because the reviewing behavior also reflects user preference. Thus, we extend the context entities to all third-party resources beyond users and items that may affect or reflect user preferences for items.

Based on our analysis of the literature and real datasets, the possible pairwise or group relationships in an RRS are specified into the following six forms.

User–User relationships (**UU**). This type of relationship refers to the correlations between users. The general considerable relationships could be user social networks [28], rating similarities [35], or implicit trust refined from common behaviors [39].

Table 1
Structure of the formalized multipartite hypergraph.

Vertex \ Hyperedge	$E^{(1)}$	$E^{(2)}$	$E^{(3)}$	$E^{(4)}$	$E^{(5)}$	$E^{(6)}$	
U	$UE^{(1)}$			$UE^{(4)}$	$UE^{(5)}$	$UE^{(6)}$	Incidence Matrix
I		$IE^{(2)}$	$IE^{(3)}$		$IE^{(5)}$	$IE^{(6)}$	
A			$AE^{(3)}$	$AE^{(4)}$			
C						$CE^{(6)}$	
Source relationships	**UU**	**II**	**IA**	**UA**	**UI**	**UCI**	

Item–Item relationships (**II**). In RSSs, restaurants may share connections because they are located in the same district or they belong to the same company. It is worth mentioning that some connections are multi-to-multi relationships rather than pairwise relationships.

Item–Attribute association relationships (**IA**). In e-commerce applications, items usually come with descriptions of various attributes. Thus, we can collect the association relationships between items and attributes.

User–Attribute preference relationships (**UA**). Users may prefer some items with particular attributes where the preference relationships between users and item attributes can be collected.

User–Item preference relationships (**UI**). Personalized ratings of items are the most explicit preferences of users. In addition, if there are no explicit ratings, the visiting or purchasing history can be seen as implicit acceptance of users for items.

User–Context–Item relationship (**UCI**). We point out that third-party context information is able to affect or reflect user preference. If user acceptance of an item is adjusted by a particular context, we then collect a high-order tripartite relationship for the user, the item and the context entities.

The above elaborates four types of information entities and six types of relationships that may appear in an RSS. Because some relationships are high-order connections between multiple objects, the conventional pairwise graph models are unable to present them well. We therefore introduce a multipartite hypergraph in the following sub-section.

4.2. Multipartite hypergraph construction

With the UIAC data model, a hypergraph $G = \{V, E\}$ will be constructed where the vertex set $V = U \cup I \cup A \cup C$ is the union of four types of objects: user nodes, item nodes, attribute nodes and context nodes, and the hyperedge set E consists of six subsets $E^{(1)}$ to $E^{(6)}$ generated from the six types of relationships, **UU, II, IA, UA, UI** and **UCI**. The general initialization of each type of edge is as follows.

The first subset of edges $E^{(1)}$ is initialized from user–user relationships **UU**. For pairwise social relationships, a pairwise edge will be set to a pair of user nodes. Note that users could be connected by multiple edges if multi-relational connections exist, as studied in our previous research [27]. Hence, a hyperedge will be set to connect multiple user nodes for group social relationships.

The second subset of edges $E^{(2)}$ is initialized from item–item relationships **II**. Similarly, the items bundled by some relationships are connected with hyperedges. For example, we initialize a hyperedge for restaurants belonging to the same catering company or those that are located in the same district of a city.

The third subset of edges $E^{(3)}$ is initialized from Item–Attribute associations **IA**. The Item–Attribute association relationships are usually represented by pairwise connections between item nodes and attribute nodes. With hypergraphs, in another way, we can instead build an overall hyperedge to include an item node and all its attributes.

The fourth subset of edges $E^{(4)}$is initialized from user-attribute preferences **UA**. The same as Item–Attribute associations, user preference to item attributes can be represented using pairwise edges or hyperedges.

The fifth subset of edges $E^{(5)}$ is initialized from User–Item ratings **UI**. Like most recommendation models, the explicit rating given by a user to an item initializes a pairwise edge between them, together with a scaled number to indicate the preference degrees.

The sixth subset of edges $E^{(6)}$ is initialized from user-context-item relationships **UCI**. If a user's preference for an item changes in different contexts, a hyperedge will connect the user node, the item node and the related context nodes. As an extension, textual comments or tags are also third-party context entities that reflect user preferences. Hence, the reviewing or tagging behavior of a user will initialize a hyperedge as a connection between the user, the item and several words.

With these settings, a multipartite hypergraph will be formalized with the structural information as shown in Table 1.

4.3. Balanced hypergraph ranking (BHR)

To overcome the possible ranking bias of traditional ranking for multipartite hypergraphs, we modify the vertex degree $d(v) = \sum_{v\in e} w(e)$ to an enhanced form:

$$d_+(v) = \sum_{v\in e} w(e)\sqrt{\delta(e)} \tag{9}$$

Different from the traditional vertex degree definition, the enhanced vertex degree of (9) also integrates the degree information of the involved edges. An enhanced node degree matrix $\mathbf{D}_+$ can then be denoted with this new setting. Next, the cost function (4) of the hypergraph ranking model will be rewritten to:

$$Q(\mathbf{f}) = \frac{1}{2}\sum_{i,j=1}^{|V|}\sum_{e\in E}\frac{w(e)h(v_i,e)h(v_j,e)}{\sqrt{\delta(e)}}\left(\frac{f_i}{\sqrt{d_+(v_i)}} - \frac{f_j}{\sqrt{d_+(v_j)}}\right)^2 + \mu\sum_{i=1}^{|V|}(f_i - y_i)^2 \tag{10}$$

In the above equation, the first part of the right side, denoting the smoothness function $S(f)$, can be simplified to matrix-vector form by the following steps:

$$\begin{aligned} S(f) &= \frac{1}{2}\sum_{i,j=1}^{|V|}\sum_{e\in E}\frac{w(e)h(v_i,e)h(v_j,e)}{\sqrt{\delta(e)}}\left(\frac{f_i}{\sqrt{d_+(v_i)}} - \frac{f_j}{\sqrt{d_+(v_j)}}\right)^2 \\ &= \sum_{i,j=1}^{|V|}\sum_{e\in E}\frac{w(e)h(v_i,e)h(v_j,e)}{\sqrt{\delta(e)}}\left(\frac{f_i^2}{d_+(v_i)} - \frac{f_i f_j}{\sqrt{d_+(v_i)}\sqrt{d_+(v_j)}}\right) \\ &= \sum_{i=1}^{|V|} f_i^2 \sum_{e\in E}\frac{w(e)h(v_i,e)\sqrt{\delta(e)}}{d_+(v_i)}\sum_{j=1}^{|V|}\frac{h(v_j,e)}{\delta(e)} - \sum_{i,j=1}^{|V|}\sum_{e\in E}\frac{f_i w(e)h(v_i,e)h(v_j,e)f_j}{\sqrt{d_+(v_i)}\sqrt{\delta(e)}\sqrt{d_+(v_j)}} \\ &= \mathbf{f}^T\mathbf{f} - \mathbf{f}^T\mathbf{D}_+^{-1/2}\mathbf{H}\mathbf{W}\mathbf{D}_e^{-1/2}\mathbf{H}^T\mathbf{D}_+^{-1/2}\mathbf{f} \\ &= \mathbf{f}^T(\mathbf{I} - \mathbf{A}_+)\mathbf{f} \end{aligned} \tag{11}$$

Here, a new intermediate matrix $\mathbf{A}_+ = \mathbf{D}_+^{-1/2}\mathbf{H}\mathbf{W}\mathbf{D}_e^{-1/2}\mathbf{H}^T\mathbf{D}_+^{-1/2}$ is introduced to replace the old matrix $\mathbf{A}$ in Section 3.2. Performing a similar calculation to (6), the optimized ranking order of BHR is obtained:

$$\mathbf{f}_{\text{BHR}} = (\mathbf{I} - \alpha\mathbf{A}_+)^{-1}\mathbf{y}, \tag{12}$$

Correspondingly, the random walk version of BHR will be $\mathbf{f}_{\text{BHR(RW)}} = (\mathbf{I} - \alpha\mathbf{T}_+)^{-1}\mathbf{y}$, where $\mathbf{T}_+ = \mathbf{D}_+^{-1}\mathbf{H}\mathbf{W}\mathbf{D}_e^{-1/2}\mathbf{H}^T$. It is easy to find that $\mathbf{A}_+$is the normalized version of $\mathbf{T}_+$.

With the settings of the proposed BHR, the results of the example in Fig. 2 will be recalculated as follows (using the random walk version). This indicates that our settings in BHR will alleviate the ranking bias of traditional methods to some extent.

$$\begin{aligned} p(B\in e_2)^{new} &= \frac{1\times\sqrt{2}}{1\times\sqrt{2}+1\times\sqrt{20}}\times\frac{1}{2} \approx 0.12 < p(B)^{old} = 0.25 \\ p(1\in e_1)^{new} &= \frac{1\times\sqrt{20}}{1\times\sqrt{2}+1\times\sqrt{20}}\times\frac{1}{20} \approx 0.038 > p(1)^{old} = 0.025 \end{aligned}$$

In Section 6, the empirical experiments on large-scale data demonstrate that BHR performs better than the existing ranking models for multipartite hypergraphs consisting of different forms of edges.

5. Multi-objective recommendation framework

With the construction of multipartite hypergraphs, we elaborate how to transfer users' multi-objective requests to computable input and propose the multi-objective recommendation framework.

5.1. Multi-objective request analysis

In traditional graph ranking-based recommendation models, the current active user is the only query node as input, and the ranking aim is to find the closest (item) nodes to this single node. In this study, a multi-objective recommendation request can be seen as multiple requirements to a set of query nodes, and the ranking aim becomes to determine the closest nodes to all members in the query set rather than an individual user. Hence, a multi-objective request can be represented by an input query set containing multiple users, item attributes, context information, etc. Fig. 4 indicates the decomposing process of multi-objective requests, also with a comparison of single-objective requests.

Based on this analysis, a multi-objective request can be decomposed to a query set $V_q \subset V$with a number of vertices in the multipartite hypergraph model. A corresponding input query vector $\mathbf{q}$ is then generated with $q_i = 1$ if the i-th hypergraph

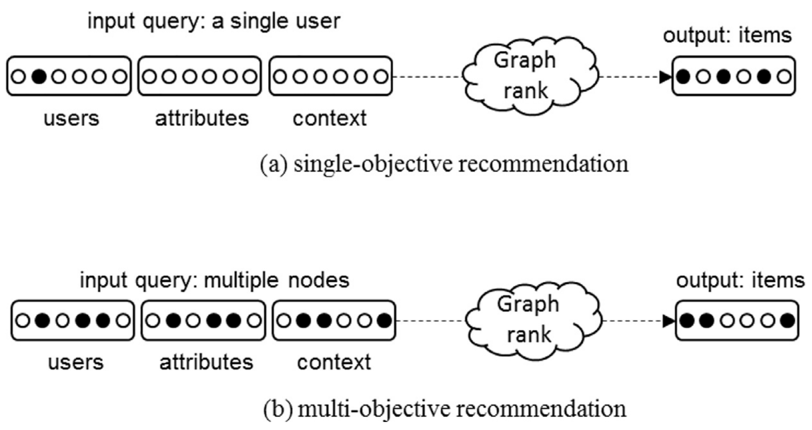
input query: a single user
users
attributes
context
Graph rank
output: items
(a) single-objective recommendation
input query: multiple nodes
users
attributes
context
Graph rank
output: items
(b) multi-objective recommendation

Fig. 4. Decomposing single- or multi-objective requests to input query sets and making recommendations.

vertex is included in the query set and zero otherwise. For a single-objective request, as a special case, the input query set contains only the current user so the input query vector will have only one positive element.

Now we can transfer a multi-objective request to a computable query vector $\mathbf{q}$ as the input for hypergraph ranking. Generally, the query set is far smaller than the whole vertex set such that the input query vector would be very sparse. Hence, we define a new query vector $\mathbf{y} = \mathbf{A}_+^T \mathbf{q}$ to replace $\mathbf{q}$ to enrich the input information.

5.2. Recommendation framework

The overall framework for multi-objective recommendations is proposed in Fig. 5 with restaurant recommendations as an example.

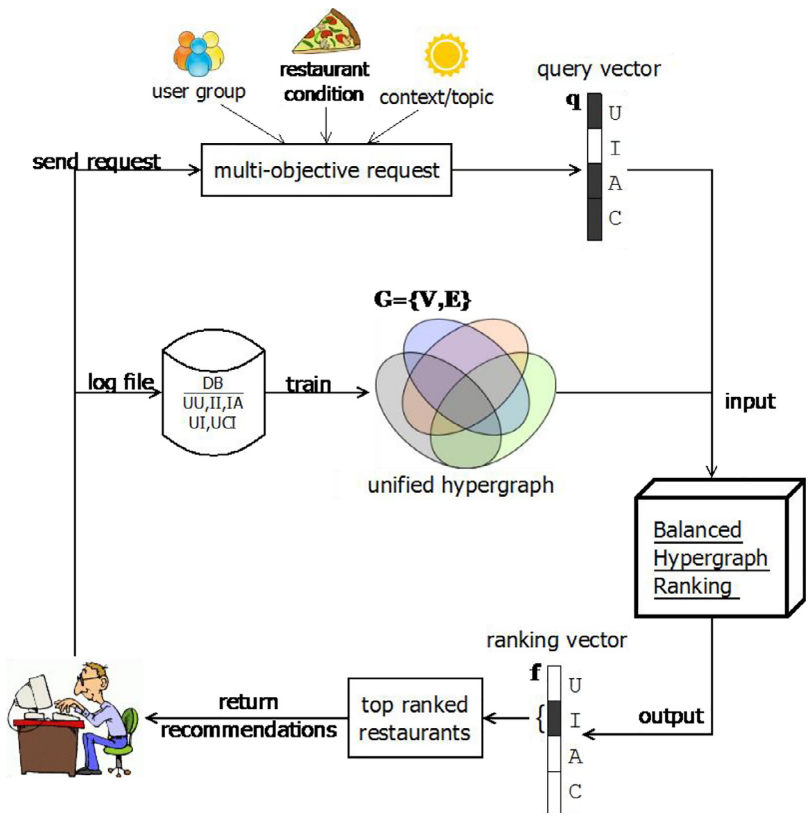

Fig. 5. Multi-objective recommendation framework for restaurants.

This framework consists of four steps to complete a multi-objective recommendation task: query generation, hypergraph construction, hypergraph ranking, and recommendation generation.

Step 1. *Query generation.* In this step, the active user sends a multi-objective recommendation request represented by a query set and the system will generate an input query vector.

Step 2. *Hypergraph construction.* A multipartite hypergraph is constructed from historical data in the background. Different resources in the database will be utilized to generate the diverse relationships between different objects.

Step 3. *Balanced hypergraph ranking.* In this step, the obtained query vector and a multipartite hypergraph constitute the input of the proposed BHR model. Some intermediate matrices like $\mathbf{D}_+$ and $\mathbf{A}_+$ are trained in advance. A ranking vector $\mathbf{f}$ will be computed using BHR to evaluate the closeness of other vertices to the whole input query set.

Step 4. *Recommendation making.* This step extracts the ranking order of only item vertices, and suggests the top ranked ones to satisfy the multi-objective recommendation requests of users.

6. Empirical experiments

Empirical experiments are conducted on a dataset of Yelp.com,[1] which is a local business review site where users can give numerical ratings and also textual comments to the merchants they have known or visited in the past. There are various information entities and complex relations available in this dataset that it can be seen as a representative case of e-commerce recommender systems. The main limitation of a generated test-set for our experiment is that all the mentioned six types of relationships in the UIAC model should be available. Few datasets provide such full information, especially context-aware information. The adopted Yelp dataset has no records about user preferences under different contexts, but it provides user comments, which can be treated as the extended context entities and user-context-item relationships defined in our UIAC data model. Thus, we can collect all types of entities and relationships in the UIAC model from the Yelp dataset for experiments, as explained as follows.

6.1. Data collection

According to the idea of the proposed UIAC data model, we first analyze all the involved information entities in the Yelp dataset.

(1) Restaurants (I): The original dataset contains local restaurants, hotels, grocery stores, etc. in a city, but we only select the restaurants from the dataset for the case of restaurant recommendations. Finally, 4119 restaurants are extracted as the recommended objects for our multi-objective recommendation problem.

(2) Users (U): In the dataset, 1911 users in total have given more than 20 ratings or comments on the selected restaurants. We chose these users for our experiments.

(3) Attributes (A): Based on the taxonomy settings in Yelp, the taxonomy of restaurants contains 24 attributes from five aspects: “type”, “ambience”, “alcohol”, “noise”, and “price range”, as shown in Fig. 6. Therefore, we build an attribute set with 24 attributes as a special group of nodes for the multipartite hypergraph.

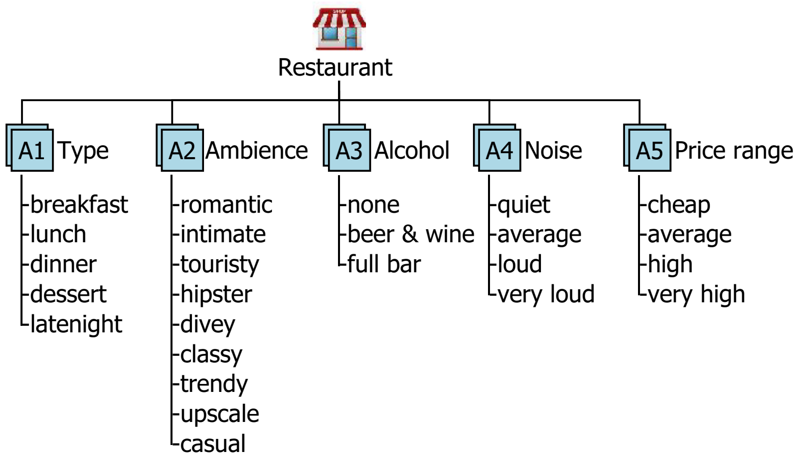

Fig. 6. The taxonomy attributes for restaurants in Yelp.com.

[1] The Yelp dataset challenge: http://www.yelp.com/dataset_challenge.

Table 2
Statistical information of the dataset.

Vertex/hyperedge	Entity/relationship	Sample size*	Avg. degree**
U	users	1911	79.9
I	restaurants	4119	17.7
A	attributes	24	851.6
C	context	7205	19.2
$E^{(1)}$	UU: friendship	43,760	2
$E^{(2)}$	II: neighborhood	16	226.7
$E^{(3)}$	IA: category	4032	6.1
$E^{(4)}$	UA: preference	1899	3.4
$E^{(5)}$	UI: rating	60,688	2
$E^{(6)}$	UCI: tips	24,541	7.6

* The total number of these types of vertices or hyperedges.
** The average vertex degree or edge degree.

(4) Context (C): Traditional context information such as time and user location is missing in the dataset, but it contains many short comments (called “tips” in Yelp) that can be seen as a kind of third-party information entities. There are over 28,000 such “tips” made by users on the restaurants in the dataset. We split them into single words and remove the most common words[2] and meaningless symbols, resulting in the extraction of 7205 single words treated as third-party context entities. A “tip” behavior is thus considered as a high-order relationship between a user, a restaurant and certain words.

These four types of information entities are treated as four sets of nodes, **U, I, A, C** to build a multipartite hypergraph. The basic statistical information is shown in Table 2.

Next, we analyze all the involved pairwise or group relationships for the four types of entities. With the UIAC dataset, the relationships can be divided to six types. For each type of relationship, the collection process is detailed as follows.

(1) User–User relationships (**UU**). We collect this type of relationship from the online friendship information of the Yelp dataset. The dataset contains 43,760 binary friendship relations between the selected 1911 users. Thus, we build a pairwise edge $\langle u_1, u_2\rangle \in E^{(1)}$ if two users are friends and let the weightings be one.
(2) Item–item relationships (**II**). The Yelp dataset contains no precise location information on the restaurants, but it records the 16 business districts where the restaurants are located. Thus, we collect a type of item–item relationship from the neighborhood information of restaurants. Concretely, we build a hyperedge $\langle i_1, i_2, i_3, \ldots\rangle \in E^{(2)}$ for those restaurants in the same business district, and the edge weighting is initialized to one.
(3) Item–Attribute association relationships (**IA**). As previously discussed, 24 taxonomy attributes are used to describe restaurant conditions. By using the hypergraph structure that allows group relationships, we assign each item (restaurant) just one hyperedge $\langle i, a_1, a_2, \ldots\rangle \in E^{(3)}$ of this type to connect the restaurant itself and all the associated attributes. The edge weighting of this type of hyperedge is set to one. Because the taxonomy information of some restaurants is missing, we finally collect 4032 association relationships between items and attributes, as shown in Table 2.
(4) User–Attribute preference relationships (**UA**). As the dataset does not provide explicit information about user preferences for attributes, we extract implicit preferences from user ratings on items. In Yelp, ratings range from 1 to 5. If a single user has visited more than five restaurants that all contain a particular attribute (e.g., “quiet”) and the average rating for these restaurants is over 3, we consider that this user has a positive preference for this attribute. A hyperedge $\langle u, a_1, a_2, \ldots\rangle \in E^{(4)}$is then built for a user to connect this user and the preferred attributes, and the weighting is one. By removing some insignificant results, we build UA relationships for 1899 out of 1911 users, as shown in Table 2.
(5) User–Item preference relationships (**UI**). The dataset contains 60,688 ratings given by the selected users for the selected restaurants, ranging from 1 to 5. We build a pairwise edge $\langle u, i\rangle \in E^{(5)}$for a pair of users and the rated item and normalize the rating value to [0,1] as the edge weighting.
(6) User–Context–Item relationship (**UCI**). As previously discussed, 7205 single words extracted from the “tips” in the dataset are collected as the set of context entities C. Accordingly, the tips of a user on a restaurant can be represented by a hyperedge $\langle u, c_1, c_2, \ldots, i\rangle \in E^{(6)}$ between the user u, the item i and the contained word nodes c_1, c_2, etc. Finally, we collect 24,541 relationships/hyperedges of this type from the dataset.

In summary, Table 2 records the statistical information of the testing dataset with the four types of nodes and six types of relationships.

It is worth mentioning that the edge degree of the constructed multipartite hypergraph varies greatly from two to hundreds. Recall the analysis of Section 3.4 where the traditional hypergraph rankings may suffer ranking bias and the improved BHR is expected to achieve better performance in this circumstance.

[2] Based on the “Long stopword list” of nature language processing provided in http://www.ranks.nl/stopwords

6.2. Compared models

The core task of a recommender system is to guess items of potential interest to the users making a recommendation request. To simulate real usage, we randomly extract half of the positively rated items for every user as hidden knowledge, and test whether a recommendation model is able to predict these items correctly. To guarantee there are sufficient data for both training and testing, only users who have more than 10 positively rated items (rating is 4 or 5) are selected to build the test set. As a result, 1827 out of 1911 users and accumulatively 26,632 preferred restaurants are extracted from the whole dataset to build the test set, of which each user has on average 14.6 preferred items.

We compare our model with three baseline recommendation approaches and three most related graph models. Second, a non-personalized recommendation approach (denoted as AVG) is produced. In this approach, restaurants are ranked by their reputations, i.e., the average ratings given by users. Second, the standard user-based CF is selected and denoted as UCF. This is a memory-based CF model that first predicts the possible ratings of a user to unknown restaurants and then determines the best candidates [35]. The third is a model-based CF: the standard SVD model that applies matrix factorization techniques to predict users' ratings to unknown items [21]. Of the graph models, the first is the contextual graph model proposed by Bogers [4], which is a representative random walk model to handle various information entities and relationships. This model constructs an ordinary graph for random walking, denoted as SGRW. The second graph model is the regularization framework of hypergraph ranking reviewed in Section 3.2 [41], denoted as HGReg. The third one is the random walk version of hypergraph ranking, denoted as HGRW [53]. Our recommendation model using the balanced hypergraph ranking method is denoted as BHR. In addition, the random walk version denoted as BHR(RW) is also evaluated. It is worth mentioning that most CF techniques (both memory-based and model-based) aim to predict the missing ratings on unknown items and hereby rank all candidate items [38], while graph models estimate the closeness ranking order directly based on the various relationships between users and items. A brief summary of all compared models are given below in Table 3.

Table 3
A summary of the compared models.

Model	Description	Predict ratings	Item ranking	Input resources
AVG	Non-personalized rating averaging	Yes	No	single (rating)
UCF [35]	The standard Memory-based CF	Yes	No	single (rating)
SVD [21]	The standard SVD model for CF	Yes	No	single (rating)
SGRW [4]	Simple graph random walk model	No	Yes	multi (context-aware)
HGRW [53]	Hypergraph random walk model	No	Yes	multi (context-aware)
HGReg [41]	Regularized hypergraph ranking	No	Yes	multi (context-aware)
BHR(RW)	Our random walk version of BHR	No	Yes	multi (context-aware)
BHR	Our balanced hypergraph ranking model	No	Yes	multi (context-aware)

6.3. Evaluation scheme and metrics

In our experiments, every model is implemented to recommend the top-N interesting restaurants for each user, and we adjust the recommendation size N to make clearer comparisons. The five graph-based models, SRW, HGRW, BHR(RW), HGReg and BHR share a similar parameter α and we set $\alpha = 0.95$ for each model initially.

For top-N recommendations, we test whether each model is able to precisely classify the items that are of potential interest to the users. However, differing to classical classification problems, we cannot classify all the most preferred items for a user because the rating data is incomplete, but instead we use the limited number of preferred items in the test set as substituted "ground-truth" for verification [25,38]. With this unique circumstance, our evaluation scheme consists of the following aspects:

(1) Classification accuracy: the precision and recall ratios of the top-N recommended items compared to the test set.
(2) Recommendation precision: whether the top-N recommendations are actually preferred by the users. The enhanced precision metric mean average precision (MAP) is used.
(3) Parameter sensitivity of the graph models. We need to test the influence of the graph ranking parameter α on the recommendation performance.
(4) First-page recommendation: can a model successfully find the items of interest sooner? It is important for a recommender system to discover user interest even when the recommendation list is very short. Here, a new metric of user satisfaction is defined.
(5) Multi-objective recommendation ability: can a model successfully predict items when the user demand is multi-objective?

With the restaurant dataset, the precision metric is defined as the proportion of correctly recommended restaurants in the whole recommendation list and recall is the proportion of correctly recommended restaurants in the test set. Because the precision and recall metrics may be inconsistent, we also introduce the F1 metric as an overall evaluation:

$$\mathrm{F1} = \frac{2 \times \mathrm{Precision} \times \mathrm{Recall}}{\mathrm{Precision} + \mathrm{Recall}} \tag{13}$$

Notice that the precision metric varies the recommendation size N. We introduce the MAP metric to test the overall recommendation precision with different recommendation sizes. For a single user, the average precision (AP) is defined as the average of precisions computed at each point of correctly recommended item in the recommendation order. Let $Corr_i$be a Boolean value denoting whether the i-th recommendation is correct, and the AP can be written as:

$$\mathrm{AP} = \underset{i=1,2,\ldots,N}{\mathrm{mean}} (Corr_i \times \mathrm{Precision@i}) \tag{14}$$

The MAP metric is the average AP scores over all tested users.

We also propose a new metric SA to test the overall user satisfaction degree of a recommender system. Simply, we consider that a user's request is satisfied if at least one acceptable item can be found from the recommendation list, and the proportion of satisfied requests is defined as the overall SA degree of a recommendation model, written as:

$$\mathrm{SA} = \frac{\mathrm{Numberofsatisfiedrequests}}{\mathrm{Totalnumberofrecommendationrequests}} \times 100\% \tag{15}$$

For multi-objective recommendations, it is hard to compute the recommendation accuracy because the data set does not provide preference information of a group of users in certain contexts. For this reason, we let each model predict a proper ranking order for the items in the test set, and test if the result is correct. The detail of the experiment settings is presented in Section 6.8. Here, we introduce the Normalized Discounted Cumulative Gain (NDCG) to compare the ranking accuracy of each model. For a list sorted in descending order with N items, let rel_i be the ranking score at the position i, and the Discounted Cumulative Gain (DCG) accumulated at the position N is calculated by (16).

$$\mathrm{DCG@}N = \sum_{i=1}^{p} \frac{2^{rel_i} - 1}{\log_2(i+1)} \tag{16}$$

Let IDCG be the ideal DCG value of the actual ranking order, then the NDCG score of a test ranking order is a normalized value between 0 and 1, as follows.

$$\mathrm{NDCG@}N = \frac{\mathrm{DCG@}N}{\mathrm{IDCG@}N} \tag{17}$$

6.4. Classification accuracy

We let each model recommend 10 restaurants (fixed $N = 10$) and compare these against the ones in the test set. The performance in terms of precision and recall is presented in Fig. 7, which preliminarily shows that our model BHR and the existing regularized hypergraph ranking model HGReg achieve the best performance. For the three random walk models, we find the existing hypergraph model HGRW performs worse than the simple graph model SGRW. With the modification mentioned in 4.3, our balanced hypergraph random walk model BHR(RW) improves accuracy significantly. This demonstrates that the proposed balanced hypergraph ranking models are able to alleviate the ranking bias of traditional models to some extent.

Fig. 7 also demonstrates that the rating prediction-based UCF and SVD suffer poor performance in dealing with top-N recommendation tasks. In our experiment, these models predict the ratings of unknown items first and then rank them according to the prediction. However, it is hard to predict ratings for many items because of the high sparsity of rating data, and the top-N recommendations that are finally selected become inaccurate.

6.5. Recommendation precision

We test the recommendation precision of every model with different recommendation sizes. Fig. 8 shows the variation in the MAP scores along with increasing recommendation sizes. It can be seen that the proposed BHR model achieves the best performance in all cases. Again, the superiority of BHR compared to HGReg indicates the success of our modification of the balanced hypergraph ranking model. This improvement is more significant when comparing the random walk models BHR(RW) and HGRW. Figs. 7 and 8 show that the unique settings of BHR are more suitable for the special multipartite hypergraph environment with various forms of relationships. Except for the graph models, however, the rating-prediction-based models SVD and UCF still suffer poor performance for top-N recommendations due to sparse rating data.

6.6. Parameter sensitive of graph models

Fig. 9 presents the trends of recommendation accuracy of each graph model when parameter α increases from 0.6 to 0.99, showing that the best performance is achieved by BHR at $\alpha = 0.95$, significantly higher than the second best HGReg. Notice that BHR and HGReg are based on the traditional regularization hypergraph ranking framework and they both improve ranking accuracy compared to their pure random-walk versions BHR(RW) and HGRW. For the random walk models,

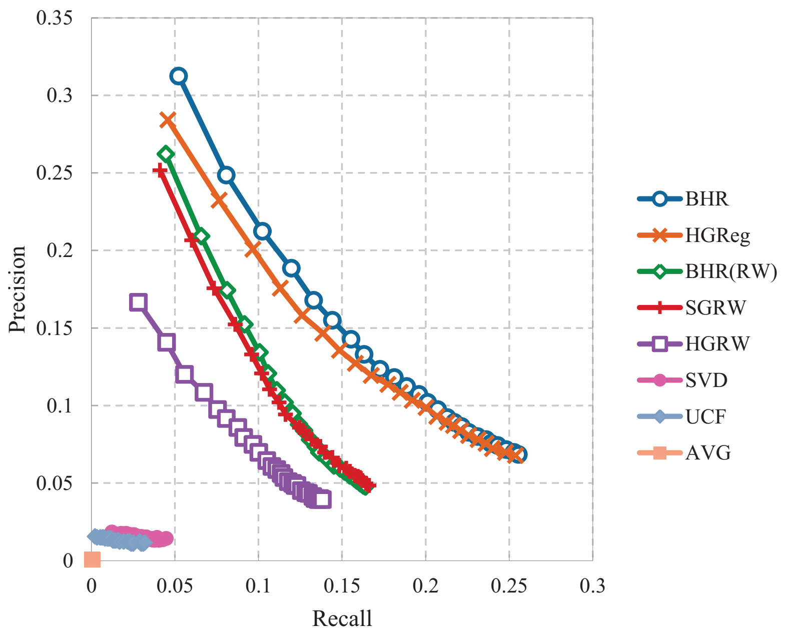

Fig. 7. The precision-recall curve of each model for top-N recommendations (N = 10).

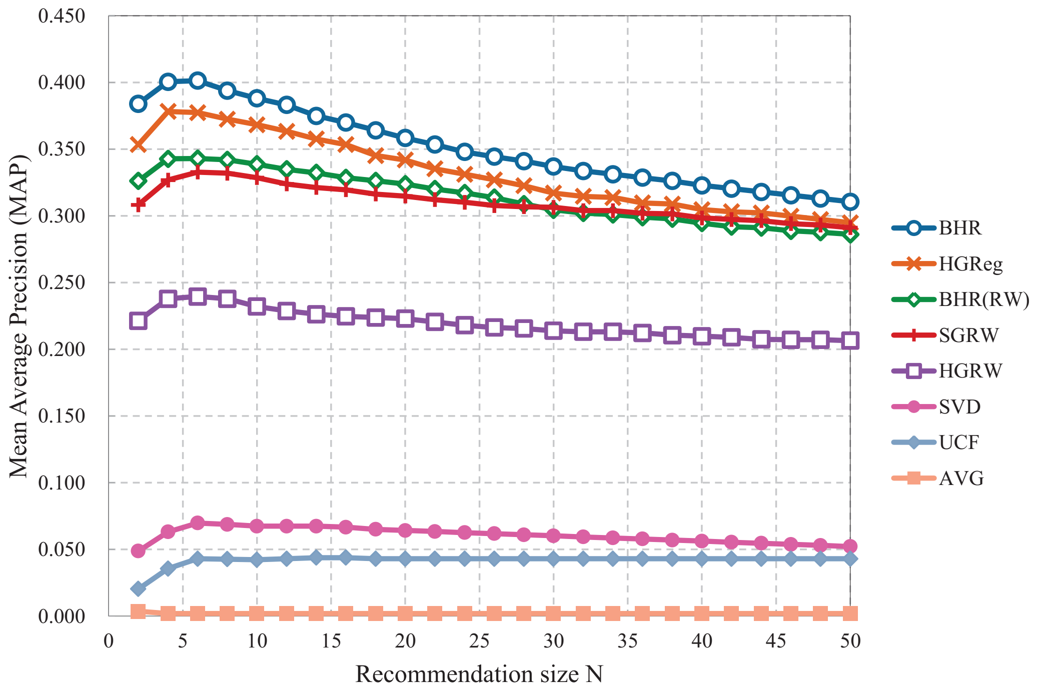

Fig. 8. The MAP values of each model with different recommendation sizes.

the traditional hypergraph model HGRW acquires the lowest precision scores, even worse than the single graph model SGRW. However, with our modification, the balanced hypergraph model BHR(RW) increases accuracy significantly and performs slightly better than SGRW.

These findings demonstrate that if we only simply import a hypergraph structure to deal with high-order relationships (like HGRW), it is hard to improve recommendation accuracy even compared to simple pairwise graph models (like SGRW). Taking the used data set as an example, some group relationships connect a large number of entities while some connect only a few, such that the traditional hypergraph ranking models suffer from the ranking bias problem discussed in

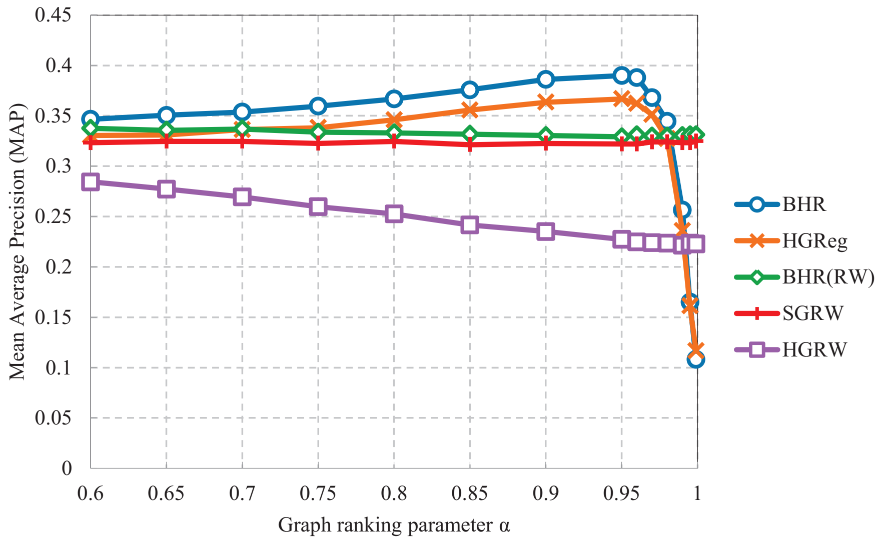

Fig. 9. The recommendation precision trends with different parameter settings (N = 10).

Table 4
One-page recommendation performance comparison.

Recmend. size		(a)	(b)	(c)	(d)	(e)	(f)	(g)	(h)	improvement	
N		AVG	UCF	SVD	SGRW	HGRW	HGReg	BHR(RW)	BHR	(g) vs. (e) (%)	(h) vs. (f) (%)
N = 2	F1	0.000	0.004	0.009	0.093	0.065	0.114	0.097	**0.122**	48.9	6.9
	MAP	0.003	0.020	0.049	0.308	0.221	0.353	0.326	**0.384**	47.4	8.6
	SA(%)	0.2	1.7	4.3	22.0	17.1	27.2	22.2	**28.1**	29.8	3.6
N = 4	F1	0.000	0.006	0.012	0.099	0.069	0.122	0.102	**0.126**	46.8	3.6
	MAP	0.002	0.036	0.063	0.327	0.238	0.378	0.343	**0.400**	44.1	5.9
	SA(%)	0.1	4.3	7.6	32.7	25.8	40.1	33.0	**40.9**	27.8	2.1
N = 6	F1	0.000	0.008	0.013	0.101	0.070	0.122	0.102	**0.127**	45.1	4.1
	MAP	0.002	0.043	0.070	0.333	0.239	0.377	0.343	**0.401**	43.2	6.3
	SA(%)	0.2	7.4	11.9	47.3	37.0	57.6	47.1	**58.4**	27.4	1.5
N = 8	F1	0.000	0.008	0.012	0.100	0.070	0.120	0.102	**0.123**	44.7	2.5
	MAP	0.002	0.043	0.069	0.332	0.238	0.372	0.342	**0.394**	43.8	5.8
	SA(%)	0.2	8.8	14.4	56.8	43.9	67.9	56.9	**69.0**	29.7	1.6
N = 10	F1	0.000	0.008	0.012	0.100	0.068	0.119	0.101	**0.122**	47.9	3.2
	MAP	0.002	0.042	0.067	0.329	0.232	0.368	0.339	**0.388**	45.9	5.4
	SA(%)	0.2	10.1	16.3	65.3	50.1	78.7	64.7	**79.7**	29.0	1.4

Section 4.3. Thus, we propose a unique balanced hypergraph ranking model to deal with the complex environment of e-commerce recommender systems.

6.7. *First-page recommendations*

In general, e-commerce sites have limited space to show recommended items, especially mobile apps on smartphones. It is our opinion that user satisfaction will be increased if they can find interesting items earlier from a shorter recommendation list, such as the first recommendation page. Thus, we compare the “first-page” recommendation performance of each model with small recommendation sizes from 2 to 10. The results are given in Table 4, showing that the BHR model achieves the best performance in all cases.

The F1 and MAP metrics show that our balanced hypergraph models BHR and BHR(RW) significantly improve recommendation accuracy compared to the existing models HGReg and HGRW respectively, especially when recommendation size is smaller. The user satisfaction metric SA demonstrates that our model BHR is able to satisfy users more quickly. In other words, users are able to find at least one preferred items from the few recommended item on the first page using the BHR model.

Moreover, we can see the pure rating prediction-based models ((a)–(c)) suffer poor performance for top-N recommendations when rating data is sparse, while the graph-based models ((d)–(h)) are able to alleviate this rating sparsity problem by utilizing wider connections between users, items and context information.

Table 5
An example of random generated multi-objective request and the query set.

User IDs	Restaurant conditions	Frequently appearing words (topics)	Corresponding vertices in the hypergraph
$\begin{cases} u_{21} \\ u_{640} \\ u_{660} \end{cases}$	$\begin{cases} \textit{Alcohol}: \textit{"none"} \\ \textit{Noise}: \textit{"quiet"} \\ \textit{Price}: \textit{"average"} \end{cases}$	$\begin{cases} \textit{"lunch"} \\ \textit{"pizza"} \\ \textit{"drinks"} \end{cases}$	$\begin{Bmatrix} v_{421}, v_{640}, v_{660}, v_{6045}, v_{6048}, \\ v_{6053}, v_{9435}, v_{10345}, v_{12003} \end{Bmatrix}$

Table 6
Ranking accuracy comparison for multi-objective recommendations.

Recommend. size	(a)	(b)	(c)	(d)	(e)	(f)	(g)	(h)	improvement	
N	AVG	UCF	SVD	SGRW	HGRW	HGReg	BHR(RW)	BHR	(g) vs. (e) (%)	(h) vs. (f) (%)
NDCG@5	0.1980	0.2392	0.2731	0.1348	0.1336	0.4132	0.1392	**0.4916**	4.2	19.0
NDCG@10	0.2652	0.2996	0.3302	0.1921	0.1884	0.5048	0.1961	**0.5688**	4.1	12.7
NDCG@30	0.4826	0.5019	0.5233	0.4317	0.4087	0.6560	0.4142	**0.6963**	1.3	6.1
NDCG@50	0.5521	0.5719	0.5843	0.5121	0.4982	0.6907	0.5021	**0.7274**	0.8	5.3
NDCG@100	0.5789	0.5997	0.6032	0.5453	0.5373	0.7030	0.5403	**0.7396**	0.6	5.2

6.8. *Multi-objective recommendations*

The above section demonstrates the strong performance of our model in dealing with traditional top-N recommendations for a single user. It is also essential to evaluate the ability of our model to deal with multi-objective recommendation problems. Our experiment settings and evaluation scheme are as follows.

A multi-objective recommendation request is not explicitly available in the dataset, but can be manually generated by assigning several request nodes from the different types of entities, according to the analysis in Section 5.1. Specifically, we randomly select three users, three restaurant attributes and three of the most frequently appearing words in the comments to build a multi-objective query set. We repeat this selection 1000 times to generate a test set. Table 5 is an example of a selected query set, which requires that recommended restaurants fit the best for these objectives, e.g., the group's cuisine preferences, restaurant conditions and comment topics.

Next, an ideal solution for each test multi-objective request needs to be generated as the ground truth to test the recommendation models. For each query set consisting of three users, we extract the restaurants preferred by any of these users from the test set to build an initial candidate set. Then it is expected to estimate the overall relevance of each candidate to the whole query set, to generate an ideal ranking order. We denote a query set as Q and the related candidate restaurant set as M. Because a candidate restaurant $m \in M$ and a single object $v \in Q$ may share multiple edges in the hypergraph model, we choose the maximum weighting of these edges as the closeness between them. The average correlation of a candidate m to all objects in Q is then seen as the ideal ranking score for this candidate, defined as $rel(m) = \sum_{v \in Q} \max_{m,v \in e} w(e)/|Q|$. In brief, if a candidate restaurant has connections with all objects in the query set, it will appear at the top of the recommendation list.

The three graph models SGRW, HGRW and HGReg are able to make multi-objective recommendations using similar settings to BHR as introduced in 5.2. The rating prediction-based models AVG, UCF and SVD, however, cannot handle multi-objective recommendations directly. For these models, we use an "aggregating and filtering" method to generate multi-objective recommendations. Taking UCF for example, given a test multi-objective query set that involves three users, we first run UCF to predict the rating of each user to each restaurant and compute the average rating to generate an initial ranking list of the restaurants. Next, we remove the restaurants without any connections to the other six nodes (restaurant attributes and comment topics) in the query set and generate a final ranking list. The top-ranked restaurants in this final ranking list are then returned as multi-objective recommendations.

Table 6 also demonstrates that our modified hypergraph random walk model BHR(RW) is slightly better than the traditional HGRW, but both may lose their advantage compared to the simple pairwise graph model SGRW at certain times. The regularized models BHR and HGReg, however, improve their performance significantly, e.g., the NDCG score increases from about 0.13 to 0.41 (HGReg) and 0.49 (BHR). This illustrates that the regularized framework of the hypergraph ranking obtains higher accuracy in dealing with e-commerce recommendation problems compared to traditional random walk models.

For the CF approaches, including memory-based UCF and model-based SVD, we adopt the aforementioned "aggregating and filtering" method to handle multi-objective recommendations, and we find the results are even better than some graph-based models. This is because the models tested in this session of experiments are only required to rank certain items in the test set rather than the whole item set so that the sparseness problem does not influence the performance of rating prediction-based approaches. We conclude that UCF and SVD are able to predict accurate ratings for users for certain items, but cannot generate a proper ranking order for all items nor can they generate precise top-N recommendations if the rating data are too sparse.

For CF approaches including memory-based UCF and model-based SVD, we adopted the mentioned "aggregating and filtering" manner to handle multi-objective recommendations, and we find the results are even better than some graph-based models. That is because the tested models in this session of experiments are only required to rank the certain items in the test set rather than the whole item set, that the sparseness problem does not influence the performance of rating prediction-based approaches. We can conclude that UCF and SVD are indeed able to predict accurate ratings for users to certain items, but cannot generate proper ranking orders for all items and generate precise top-N recommendations if the rating data are too sparse.

To summarize all the aforementioned analyses, the proposed BHR model is able to improve recommendation quality compared to existing baseline and graph-based models for both traditional single-objective and the new multi-objective recommendations. The results support our expectation that the multi-objective recommendation framework performs well in complex e-commerce environments with various information resources.

7. Conclusions and future study

In this paper, we highlighted that e-commerce users often raise multi-objective requests, which cannot be tackled by traditional single-objective recommender systems. We summarized the diverse information resources, such as users, items, item attributes and context in e-commerce environments within a UIAC data model and discussed the construction of a multipartite hypergraph based on the pairwise or group relationships between different entities. In addition, multi-objective recommendation requests can be decomposed to special requirements to several query nodes in the multipartite hypergraph, and the multi-objective recommendation problem is transferred to a hypergraph ranking problem. We pointed out that traditional hypergraph ranking models might suffer from a ranking bias problem, so we developed an improved balanced hypergraph ranking to solve the ranking problem to determine the most appropriate items that best meet all requirements. A theoretical framework was then proposed as a guideline for e-commerce multi-objective recommender system implementations. We conducted empirical experiments on the data from Yelp to evaluate the performance of the proposed model. The results indicate that our model is able to handle and improve both single-objective and multi-objective recommendations, especially when the recommendation size is small. It can be concluded that the multi-objective recommendation framework performs well for practical applications. More importantly, it is also flexible when people change their requests at particular times.

A limitation of this study is the model scalability. The multipartite hypergraph model is easy to extend by adding new nodes or new edges when the system volume grows larger. However, we admit that the current solution may not be scalable enough because some large intermediate matrices need to be recalculated. To improve scalability, there is a need to develop a new approximation strategy to solve the optimization function of our model, and this is left as an important direction of our future study. Another limitation is that graph models are not the only ones suitable for multi-objective recommendations. It is also practicable to build a tensor model to handle heterogeneous information according to the UIAC data model, but it lacks sufficient study currently. Furthermore, exploring a new unified tensor model for multi-objective or context-aware recommendations could be a new challenge and a contribution of our future study. We also mentioned a limitation in the selected test set, that is, the explicit context-aware preference information is not available. In future studies, we will crawl context-aware data from real applications to test the multi-objective and context-aware recommendation models.

Acknowledgment

This work is supported by the Australian Research Council (ARC) (DP170101632), the National Science Foundation of China (61802156), and the Educational Commission of Jiangxi Province of China (GJJ170344, GJJ170346).

References

[1] G. Adomavicius, A. Tuzhilin, Context-aware recommender systems, in: F. Ricci, L. Rokach, B. Shapira, P.B. Kantor (Eds.), Recommender Systems Handbook, Springer, US, 2011, pp. 217–253.
[2] I. Ben Sassi, S. Mellouli, S. Ben Yahia, Context-aware recommender systems in mobile environment: on the road of future research, Inform. Syst. 72 (2017) 27–61.
[3] J. Bobadilla, F. Ortega, A. Hernando, A. Gutiérrez, Recommender systems survey, Knowl. Based Syst. 46 (2013) 109–132.
[4] T. Bogers, Movie recommendation using random walks over the contextual graph, in: Proceedings of the 2nd International Workshop on Context Aware Recommender Systems, 2010.
[5] R. Burke, The wasabi personal shopper: a case-based recommender system, in: Proceedings of the 11th National Conference on Innovative Applications of Artificial Intelligence, John Wiley & Sons, 1999, pp. 844–849.
[6] R. Burke, Hybrid recommender systems: survey and experiments, User Model. User Adap. Inter. 12 (2002) 331–370.
[7] J. Castro, J. Lu, G. Zhang, Y. Dong, L. Martínez, Opinion dynamics-based group recommender systems, IEEE Trans. Syst. Man Cybern. Syst. (2017) 1–13.
[8] H. Cheng, P.-N. Tan, J. Sticklen, W.F. Punch, Recommendation via query centered random walk on k-partite graph, in: Proceedings of the Seventh IEEE International Conference on Data Mining (ICDM), IEEE, 2007, pp. 457–462.
[9] A.K. Dey, G.D. Abowd, D. Salber, A conceptual framework and a toolkit for supporting the rapid prototyping of context-aware applications, Hum. Comput. Interact. 16 (2001) 97–166.
[10] F. Fouss, A. Pirotte, J.-M. Renders, M. Saerens, Random-walk computation of similarities between nodes of a graph with application to collaborative recommendation, IEEE Trans. Knowl. Data Eng. 19 (2007) 355–369.
[11] M. Gori, A. Pucci, V. Roma, I. Siena, ItemRank: a random-walk based scoring algorithm for recommender engines, in: Proceedings of the International Joint Conference on Artificial Intelligence, 2007, pp. 2766–2771.
[12] T. He, H. Yin, Z. Chen, X. Zhou, S. Sadiq, B. Luo, A spatial-temporal topic model for the semantic annotation of POIs in LBSNs, ACM Trans. Intelligent Syst. Technol. 8 (2016) 1–24.
[13] D. Horowitz, D. Contreras, M. Salamó, EventAware: A mobile recommender system for events, Pattern Recognition Letters 105 (2018) 121–134.

[14] C.-L. Huang, P.-H. Yeh, C.-W. Lin, D.-C. Wu, Utilizing user tag-based interests in recommender systems for social resource sharing websites, Knowl. Based Syst. 56 (2014) 86–96.
[15] Z. Huang, W. Chung, H. Chen, A graph model for e-commerce recommender systems, J. Am. Soc. Inform. Sci. Technol. 55 (2004) 259–274.
[16] M. Jamali, M. Ester, TrustWalker:a random walk model for combining trust-based and item-based recommendation, in: Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2009, pp. 397–406.
[17] D. Jannach, Z. Karakaya, F. Gedikli, Accuracy improvements for multi-criteria recommender systems, in: Proceedings of the 13th ACM conference on electronic commerce, ACM, 2012, pp. 674–689.
[18] D. Kim, C. Park, J. Oh, H. Yu, Deep hybrid recommender systems via exploiting document context and statistics of items, Inform. Sci. 417 (2017) 72–87.
[19] I. Konstas, V. Stathopoulos, J.M. Jose, On social networks and collaborative recommendation, in: Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval, ACM, 2009, pp. 195–202.
[20] M. Korakakis, P. Mylonas, E. Spyrou, Xenia: a context aware tour recommendation system based on social network metadata information, in: Proceedings of the 2016 11th International Workshop on Semantic and Social Media Adaptation and Personalization (SMAP), 2016, pp. 59–64.
[21] Y. Koren, R. Bell, C. Volinsky, Matrix factorization techniques for recommender systems, Computer 42 (2009) 30–37.
[22] N. Li, Y. Xiao-Wei, Z. Cheng-Qi, Z. Shi-Chao, Product hierarchy-based customer profiles for electronic commerce recommendation, in: Proceedings of the International Conference on Machine Learning and Cybernetics, Vol. 1072, 2002, pp. 1075–1080.
[23] J. Liu, C. Wu, W. Liu, Bayesian probabilistic matrix factorization with social relations and item contents for recommendation, Decis. Support Syst. 55 (2013) 838–850.
[24] L. Lovasz, L. Lov, O.P. Erdos, Random walks on graphs: a survey, Combinatorics 8 (4) (1993) 1–46.
[25] J. Lu, D. Wu, M. Mao, W. Wang, G. Zhang, Recommender system application developments: a survey, Decis. Support Syst. 74 (2015) 12–32.
[26] M. Mao, J. Lu, G. Zhang, J. Zhang, A fuzzy content matching-based e-commerce recommendation approach, in: Proceedings of the 2015 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE), 2015, pp. 1–8.
[27] M. Mao, J. Lu, G. Zhang, J. Zhang, Multirelational social recommendations via multigraph ranking, IEEE Trans. Cybern. 47 (12) (2017) 1–13.
[28] P. Massa, P. Avesani, Trust-aware recommender systems, in: Proceedings of the 2007 ACM Conference on Recommender Systems, ACM, Minneapolis, MN, USA, 2007, pp. 17–24.
[29] J. Masthoff, Group recommender systems: Combining individual models, Recommender systems handbook, Springer, Boston, MA, 2011, pp. 677–702.
[30] J. McAuley, J. Leskovec, Hidden factors and hidden topics: understanding rating dimensions with review text, in: Proceedings of the 7th ACM Conference on Recommender Systems, ACM, Hong Kong, China, 2013, pp. 165–172.
[31] J.M. Noguera, M.J. Barranco, R.J. Segura, L. Martínez, A mobile 3D-GIS hybrid recommender system for tourism, Inform. Sci. 215 (2012) 37–52.
[32] M.J. Pazzani, D. Billsus, Content-based recommendation systems, in: The Adaptive Web, Springer, 2007, pp. 325–341.
[33] F. Pourgholamali, M. Kahani, E. Bagheri, Z. Noorian, Embedding unstructured side information in product recommendation, Electron. Commer. Res. Appl. 25 (2017) 70–85.
[34] Y.S. Rawat, M.S. Kankanhalli, ClickSmart: a context-aware viewpoint recommendation system for mobile photography, IEEE Trans. Circuits Syst. Video Technol. 27 (2017) 149–158.
[35] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, J. Riedl, GroupLens: an open architecture for collaborative filtering of netnews, in: Proceedings of the 1994 ACM conference on Computer supported cooperative work, ACM, 1994, pp. 175–186.
[36] P.A. Riyaz, S.M. Varghese, A scalable product recommendations using collaborative filtering in hadoop for bigdata, Procedia Technol. 24 (2016) 1393–1399.
[37] J.B. Schafer, J. Konstan, J. Riedl, E-commerce recommendation applications, in: R. Kohavi, F. Provost (Eds.), Applications of Data Mining to Electronic Commerce, Springer, US, 2001, pp. 115–153.
[38] Y. Shi, M. Larson, A. Hanjalic, Collaborative filtering beyond the user–item matrix, ACM Comput. Surv. 47 (2014) 1–45.
[39] K. Shiratsuchi, S. Yoshii, M. Furukawa, Finding unknown interests utilizing the wisdom of crowds in a social bookmark service, in: Proceedings of the 2006 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology, IEEE Computer Society, 2006, pp. 421–424.
[40] S. Stabb, H. Werther, F. Ricci, A. Zipf, U. Gretzel, D.R. Fesenmaier, C. Paris, C. Knoblock, Intelligent systems for tourism, IEEE Intell. Syst. 17 (2002) 53–64.
[41] S. Tan, J. Bu, C. Chen, B. Xu, C. Wang, X. He, Using rich social media information for music recommendation via hypergraph model, ACM Trans. Multimed. Comput. Commun. Appl. 7S (2011) 1–22.
[42] R.Y. Toledo, L. Martinez, Fuzzy tools in recommender systems: a survey, Int. J. Comput. Intell. Syst. 10 (1) (2017) 776–803.
[43] M. Unger, A. Bar, B. Shapira, L. Rokach, Towards latent context-aware recommendation systems, Knowl. Based Syst. 104 (2016) 165–178.
[44] K. Verbert, N. Manouselis, X. Ochoa, M. Wolpers, H. Drachsler, I. Bosnic, E. Duval, Context-aware recommender systems for learning: a survey and future challenges, IEEE Trans. Learn. Technol. 5 (2012) 318–335.
[45] N.M. Villegas, C. Sánchez, J. Díaz-Cely, G. Tamura, Characterizing context-aware recommender systems: a systematic literature review, Knowl. Based Syst. 140 (2018) 173–200.
[46] S. Wang, C. Li, K. Zhao, H. Chen, Learning to context-aware recommend with hierarchical factorization machines, Inform. Sci. 409–410 (2017) 121–138.
[47] W. Wang, G. Zhang, J. Lu, Member contribution-based group recommender system, Decis. Support Syst. 87 (2016) 80–93.
[48] K. Wei, J. Huang, S. Fu, A survey of e-commerce recommender systems, in: Proceedings of the 2007 International Conference on Service Systems and Service Management, 2007, pp. 1–5.
[49] W. Wu, J. Zhao, C. Zhang, F. Meng, Z. Zhang, Y. Zhang, Q. Sun, Improving performance of tensor-based context-aware recommenders using bias tensor factorization with context feature auto-encoding, Knowl. Based Syst. 128 (2017) 71–77.
[50] B. Xiao, I. Benbasat, E-commerce product recommendation agents: use, characteristics, and impact, MIS Q. 31 (2007) 137–209.
[51] W.X. Zhao, S. Li, Y. He, E.Y. Chang, J.R. Wen, X. Li, Connecting social media to e-commerce: cold-start product recommendation using microblogging information, IEEE Trans. Knowl. Data Eng. 28 (2016) 1147–1159.
[52] Y. Zheng, B. Mobasher, R. Burke, Context recommendation using multi-label classification, in: Proceedings of the 2014 IEEE/WIC/ACM International Joint Conferences on Web Intelligence (WI) and Intelligent Agent Technologies (IAT), 2014, pp. 288–295.
[53] D. Zhou, J. Huang, B. Schölkopf, Learning with Hypergraphs: Clustering, Classification, and Embedding, Twentieth Annual Conference on Neural Information Processing Systems (NIPS 2006), MIT Press, 2007, pp. 1601–1608.
[54] D. Zhou, B. Schölkopf, A regularization framework for learning from graph data, in: Proceedings of the ICML workshop on statistical relational learning and Its connections to other fields, 2004, pp. 67–68.
[55] T. Zhu, P. Harrington, J. Li, L. Tang, Bundle recommendation in e-commerce, in: Proceedings of the 37th international ACM SIGIR conference on Research & development in information retrieval, ACM, Gold Coast, Queensland, Australia, 2014, pp. 657–666.