Gated Hypergraph Neural Network for Scene-aware Recommendation

Tianchi Yang[1], Luhao Zhang[2], Chuan Shi*[1], Cheng Yang[1], Siyong Xu[1], Ruiyu Fang[2], Maodi Hu[2], Huaijun Liu[2], Tao Li[2], Dong Wang[2]

[1]Beijing University of Posts and Telecommunications, China [2]Meituan, China
{yangtianchi, shichuan, xusiyong}@bupt.edu.cn albertyang33@gmail.com
{zhangluhao, fangruiyu, humaodi, liuhuaijun, litao19, wangdong07}@meituan.com,

Abstract. To improve e-commercial recommender systems, researchers have never stopped exploring the interactions between users and items. Unfortunately, most existing methods only explore one or some certain components of the entire interactions. In fact, the entire interaction process is much richer and more complex, including but not limited to "who purchases what items in which merchant under what interaction environments". Furthermore, many interactions have common features, thus forming a scene, a kind of prior knowledge for predicting user interactions. In this paper, we make the first attempt to study the scene-aware recommendation, which provides better recommendations with the entire interaction modeling and the scene prior knowledge. To this end, we propose a novel gated hypergraph neural network for Scene-aware Recommendation (SREC). Particularly, we first construct a heterogeneous scene hypergraph to model the entire interactions and scene prior knowledge. Then we propose a novel scene-aware gate mechanism-based hypergraph neural network to enrich their representations. Finally, we design a separable score function to predict the matching scores among user, scene, merchant and interaction environments for training and inference procedures. Extensive experiments demonstrate that our SREC can fully leverage the scene prior knowledge and outperforms state-of-the-art methods on real industrial datasets.

Keywords: Recommendation systems · Hypergraph neural network

1 Introduction

With the rapid development of online social media and e-commerce, the benefits of Recommendation Systems (RSs) are well recognized as a basic service of e-commerce platforms [13, 24], based on which users could filter out numerous uninformative messages and facilitate decision-making [27].

Since the birth of the recommendation system, researchers have never stopped exploring the interactions between users and items to accurately predict user

* Corresponding author.

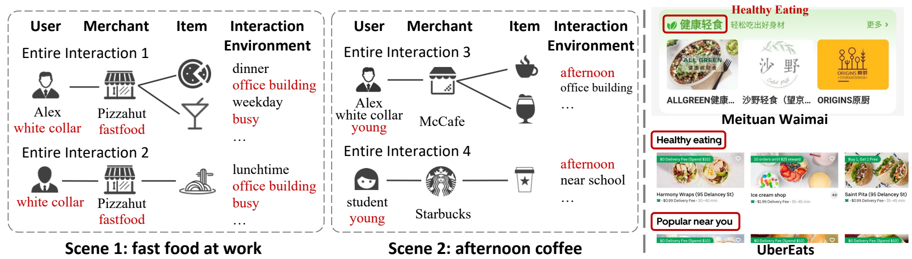


Fig. 1. (Left) Examples of entire interactions and scenes. The key features related to the scene are marked in red. (Right) Applications of scenes in e-commercial platforms.

preferences and suggest items. For example, classic recommendation methods, e.g., matrix factorization [18], mainly model users' preferences towards items using only simple historical user-item interaction records such as ratings [6]. Since these methods suffer from cold start and data sparsity problems [24], one or some kinds of interaction information are introduced into RSs. Social recommendation [17, 10] and location-aware recommendation [2, 22] respectively leverage social friendships and location relationships to enrich the interaction information. Similarly, context-aware recommendation [21, 4, 3, 5] incorporates contextual information, e.g., weather, location, etc., alongside the core data (users and items) for better recommendations. Moreover, heterogeneous information network [24, 12, 15] is employed to fuse more variety of rich interaction attributes to learn better user/item embeddings.

Despite the great success of these methods by exploring the interaction, in practical applications, the interaction process is much richer and more complex, and these methods only explore some certain components of the entire interactions. For instance, in e-commercial applications, especially food delivery industry such as Meituan Waimai, UberEats, GrubHub, etc., an entire interaction includes but is not limited to "who purchases what items in which merchant under what interaction environments[1]", which involves multiple entities (e.g., users, merchants, items) and interaction environments (e.g., time period, location, season). As Figure 1 illustrates, a white-collar, Alex, bought fast food, pizza and juice, for dinner at PizzaHut near his office building on a weekday to save time, etc. Most previous studies only consider the user and item, e.g., "Alex, pizza and juice", while some further introduce one or some more interaction information, e.g., "office building" in location-aware recommendation, "dinner, weekday" in context-aware recommendation. They still fail to formally and explicitly model the entire interactions.

Furthermore, we can find that many interactions have common features, which form a *scene*. As Figure 1 illustrates, there are a series of interactions (e.g., entire interactions 1 and 2) containing the same features "white-collar, fast-food,

[1] The term interaction environment refers to the properties specific to the interaction itself, e.g., spatial and temporal properties.

office building, weekday" respectively from the user profiles, merchant/item attributes and interaction environments. Then we can summarize a scene named "fast food at work", representing a common prior knowledge that white-collars likely prefer to buy fast food when being busy at work. Similarly, we can conclude another scene named "afternoon coffee", representing that young people tend to buy coffee in the afternoon to keep awake. Scenes have begun to be employed to e-commercial applications, especially food delivery applications, such as Meituan Waimai as shown in Figure 1, and these scene information have been summarized based on the entire interaction records through manual rules or statistical methods. Scene is totally different from context in context-aware recommendation. Although the meaning of the context is extended from temporal-spatial properties in early definitions [21] to click sequence [7], social relationships [17], etc., it is still a supporting component of interaction. Differently, a scene is the abstraction of a group of entire interactions involving users, merchants, items, and interaction environments. Note that, in this work, we focus on the usage of the scene prior knowledge tagged beforehand, and leave the automatically extracting scene from interactions for future work.

With the help of scenes, the recommendation system will benefit from the following advantages: (1) the entire interaction between user and items can be effectively characterized by scenes, since scene is a high-level abstraction of not only user and item, but also merchant and interaction environments; (2) the scenes bring more comprehensive prior knowledge about the entire interactions, so that avoid improper recommendation due to incomplete priors (e.g., recommend coffee to a coffee lover even late at night); (3) the scenes can make the recommendation list more interpretable and well-organized as shown in Figure 1. In a word, we can provide better recommendations with the entire interaction modeling and the scene prior knowledge. We name this recommendation setting that introduces scene prior knowledge as scene-aware recommendation. Different from existing studies, which are limited on modeling interactions and hence failing to leverage the scene prior knowledge, we model the entire interactions from a more comprehensive perspective and successfully make full use of the scenes.

In this paper, we make the first attempt to study the scene-aware recommendation. However, this is challenging due to the following reasons: Firstly, each entire interaction involves multiple entities (user, merchant, item) and interaction environments (time period, location, season, etc.) as well as complex relations among them. How to model such complex entire interactions? Secondly, each specific entity is involved in multiple scenes since user preferences have always been changing (e.g., "Alex" is involved in two interactions of different scenes in Figure 1). How to correctly extract the scene-specific information from each entity and make full use of it? Thirdly, the scene is already known in training samples, but in real applications, it is unknown to which scene the user's potential behavior will belong. How to bridge this gap and correctly infer the possible scenes for users while predicting user preference?

To tackle the aforementioned challenges, we propose a novel gated hypergraph neural network for **S**cene-aware **REC**ommendation, named SREC. Particularly, we first construct a heterogeneous scene hypergraph to model the entire interactions among users, merchants, items and interaction environments together with the scenes, which both comprehensively models the complex relationships among them all and appropriately incorporates the scene prior knowledge. Then we propose a novel gated hypergraph neural network to learn the representations of user, merchant and scene with a scene-aware gate mechanism designed to effectively discern different scene-specific information. Finally, we propose an effective separable score function to support a two-stage inference in line with practical requirements that firstly infers the proper scene before the final recommendation. The main contributions are summarized as follows:

(1) To the best of our knowledge, *this is the first attempt* to study scene-aware recommendation which provides better recommendation with the entire interaction modeling and the scene prior knowledge.
(2) We propose a novel gated hypergraph neural network for scene-aware recommendation. It first constructs a heterogeneous scene hypergraph to model the entire interactions and the scenes, then designs a gated hypergraph neural network followed by a separable score function to predict user preference with the help of scenes.
(3) Extensive experiments verify that our SREC can make full use of the entire interaction modeling and scene prior knowledge, thus greatly outperforming state-of-the-art (SOTA) methods for two settings on real industrial datasets.

2 Preliminary

As mentioned above, in e-commercial applications, an interaction usually forms "who purchases what items in which merchant under what interaction environments", commonly involving a user, a merchant, several items and the interaction environments. In order to model the interaction comprehensively, we formalize it as an *entire interaction*.

Definition 1. ***Entire Interaction.*** *Given a 4-tuple* $<\mathcal{U},\mathcal{M},\mathcal{I},\mathcal{C}>$ *(denoted as* Γ *for short,* $\mathcal{U}$*,* $\mathcal{M}$*,* $\mathcal{I}$ *and* $\mathcal{C}$ *denoting the set of users, merchants, items and interaction environments, respectively), an entire interaction* $\tau \in \Gamma$ *is formulated as* $\tau =< u, m, \boldsymbol{i}, \boldsymbol{c} >$*, representing a user* $u \in \mathcal{U}$ *purchased some items* $\boldsymbol{i} \subset \mathcal{I}$ *in the merchant* $m \in \mathcal{M}$ *under the interaction environments* $\boldsymbol{c} \subset \mathcal{C}$*.*

Note that U, M, I may contain attribute information, which is default to make the definition clearer. Different from the traditional recommendation methods that only users and items are explicitly modeled, we also explicitly model merchants and interaction environments. They are somehow underestimated as implicit auxiliary attributes in traditional e-commercial apps, but have essential influence for location-based services, e.g., Meituan Waimai. Considering the interaction environment "afternoon", for instance, people are likely to drink coffee in the afternoon but hardly at night. Furthermore, we find that many interactions have common features, thus forming a *scene*, which is formalized as follows.

Definition 2. ***Scene.*** *A scene $s \in S$ is defined as a set of entire interactions that have common features. Therefore, each entire interaction can be labeled with a scene by the scene function $\psi : \Gamma \rightarrow S$.*

As Figure 1 shows, each scene indicates a kind of common purchase patterns, e.g., "scene 1" represents "young white-collars often choose fast food at work on weekdays". The scenes are usually obtained by summarizing the entire interaction records through manual rules or statistical methods in industrial applications. Therefore, each entire interaction τ can be tagged with one of the predefined scenes by the manual-defined scene function $\psi(\tau)$. Therefore, we can provide better recommendations with the entire interaction modeling and scene prior knowledge. We name this recommendation setting *scene-aware recommendation*. In this work, we only focus on recommending merchants, an urgent practical task for industrial applications like food delivery, while the item recommendation is left as future work.

Definition 3. ***Scene-aware Recommendation.*** *Given the entire interactions labeled with $|S|$ scenes, i.e., $< \Gamma; S >$, for a user u under interaction environments $\boldsymbol{c}$, scene-aware recommendation aims to predict a merchant m with the help of scenes S, i.e., $P(m|\boldsymbol{c}, u; S)$.*

Note that scene-aware recommendation is different from existing recommendation. Previous recommendation settings, e.g., session-based recommendation, are mostly meant to predict m based on a given u, i.e., $P(m|u)$, while some others, e.g., context-aware recommendation, only further consider the interaction environments, i.e., $P(m|\boldsymbol{c}, u)$. Our scene-aware recommendation will leverage the scene prior knowledge, thus forming $P(m|\boldsymbol{c}, u; S)$.

3 Methodology

In this section, we first present an overview for the proposed SREC. The basic idea is that, as illustrated in Figure 2, with comprehensively modeling entire interactions together with the scenes, we design a novel gated hypergraph neural network to enrich the representations of users, merchants and scenes, followed by a score function to make the scene bridge the given user and corresponding interaction environments to the recommended merchants.

In detail, to model the entire interactions among users, merchants, items, interaction environments and scenes, we construct a heterogeneous scene hypergraph based on the entire interaction records, where each hyperedge and its type represent an entire interaction and the corresponding scene. Next, a novel gated hypergraph neural network enriches the representations of users, merchants and scenes by aggregating the hypergraph-based neighboring information. During aggregation, a scene-aware gate mechanism is designed to effectively discern different scene-specific information and make full use of them correctly. Noting that we only explicitly enrich the embeddings of users, merchants and scenes in this work to reduce computing costs, since our task focuses on recommending

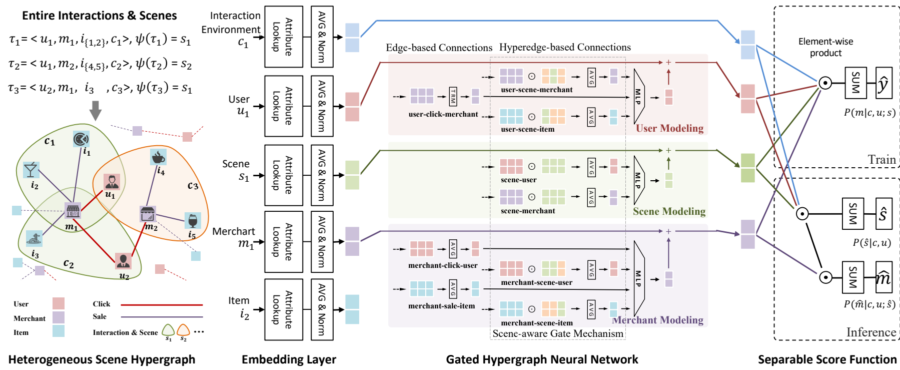


Fig. 2. Illustration of the proposed model SREC.

merchant to user. Finally, only after an interaction occurs can it be confirmed which scene this interaction will belong to, i.e., the scene is not available that the user's following purchase will belong to. Therefore, we design a separable score function to firstly infer proper scenes and then recommend merchants.

3.1 Heterogeneous Scene Hypergraph

Since each entire interaction involves multiple entities and interaction environments, we model them and scenes via a heterogeneous scene hypergraph.

Specifically, we first build a graph with three types of nodes, i.e., user, merchant and item nodes. To model the historical interaction records, we add a hyperedge for each entire interaction, e.g., a hyperedge $e = \{u_1, m_1, i_1, i_2\}$ is built for entire interaction τ_1 in Figure 2. Naturally, we attach the entity attributes to the node features, and can attach the interaction environments to hyperedge features. Then, the scene tagged on each entire interaction $\psi(\tau_1)$ is attached to the hyperedge type (refer to the hyperedge color in Figure 2). Therefore, each hyperedge and its connecting nodes together with their features can represent an entire interaction instance. Furthermore, two types of edges are added to enrich the information of the heterogeneous scene hypergraph. Particularly, we establish edges between a certain user and his/her clicking merchants to model the user's short-term historical behaviors within a session. Edges are built between a certain merchant and some items, representing sales relationship. In summary, the heterogeneous scene hypergraph has three types of nodes (users $\mathcal{U}$, merchants $\mathcal{M}$, items $\mathcal{I}$), two types of undirected edges for click and sale, and $|\mathcal{S}|$ types of attributed hyperedges.

3.2 Embedding Layer

We propose to initialize the representations of users, merchants, items, interaction environments and scenes with their attributes, since the attribute-based representations can alleviate the cold start problem for both entities and scenes.

Taking a user $u \in \mathcal{U}$ as an example, we first embed each feature field into a low-dimensional space and then fuse them into node embedding. Formally,

$$\boldsymbol{e}_i = 1/q \cdot \boldsymbol{x}_i \cdot \boldsymbol{F}_i, \quad i = 1, 2, \cdots, F, \tag{1}$$

$$\boldsymbol{u} = h(\boldsymbol{e}_1, \boldsymbol{e}_2, \cdots, \boldsymbol{e}_F), \tag{2}$$

where F is number of feature fields, $\boldsymbol{x}_i$ is the one-/multi-hot encoding of the i-th field, $\boldsymbol{F}_i$ is attribute embedding matrix, q is the number of non-zero elements of $\boldsymbol{x}_i$. Following traditional settings [7], average function is adopted as the field aggregation function. Besides, we further add a layer normalization for fast convergence, formally, $h = \text{LayerNorm} \circ \text{AVG}$.

Similarly, we can embed merchant, item, interaction environment and scene as dense embeddings $\boldsymbol{m}$, $\boldsymbol{i}$, $\boldsymbol{c}$ and $\boldsymbol{s}$ based on their attributes[2], respectively.

3.3 Gated Hypergraph Neural Network

To show the intuition of the gated hypergraph neural network, we illustrate three observations based on the e-commercial data, being described in accordance with the left part in Figure 2: (1) Apparently, the entities belonging to a certain interaction are related to each other, e.g., a node u_1 and its neighbors m_1, i_1, i_2 under the hyperedge τ_1 are related. (2) Each user (merchant, or item) will have interactions of different scenes, but the reasons why it belongs to one specific scene may be completely different from another. For instance, u_1 is included in both τ_1 and τ_2), but the reason why it belongs to the scene s_1 could be completely different from s_2. (3) Different users (merchants, or items) who have the interactions of the same scene are also related, i.e., two unconnected nodes are related by the same hyperedge type, e.g., u_1 and u_2 are related under scene s_1. Therefore, the scenes are exactly required by the gate mechanism, namely scene-aware gate mechanism, which could filter out irrelevant information to a specific scene while strengthen the relevant information during neighboring information propagation. In the following, we will introduce the propagation rule for user, merchant and scene, respectively.

User Modeling. In our hypergraph, if the central node u is a user node, it has three types of connections: user-click-merchant, user-hyperedge-merchant[3] and user-hyperedge-item. For user-click-merchant connections, following previous work [16], we apply Transformer to capture the user behavior information:

$$\boldsymbol{u}_b = \text{Transformer}\left(\boldsymbol{m}_1, \boldsymbol{m}_2, \cdots, \boldsymbol{m}_n\right), \tag{3}$$

where n is the number of sampled neighbors. For the other two types of hyperedge-based connections, we design a scene-aware gate mechanism to filter out irrelevant information to a specific scene and allow the propagation of scene-aware

[2] We take the scene attributes from the common features of its containing entire interactions.

[3] User-hyperedge-merchant represents a series of connection forms: user-s_1-merchant, user-s_2-merchant, $\cdots$, and similarly for user-hyperedge-item.

information. Next, following [7], we use the average function to aggregate for reducing the computational complexity. Formally,

$$\boldsymbol{u}_{sm} = \mathrm{AVG}(\{\boldsymbol{m}_j \odot \boldsymbol{s}_j | j = 1, 2, \cdots, n\}), \tag{4}$$
$$\boldsymbol{u}_{si} = \mathrm{AVG}(\{\boldsymbol{i}_j \odot \boldsymbol{s}_j | j = 1, 2, \cdots, n\}), \tag{5}$$

where $\odot$ denotes element-wise product, m_j and i_j represent the neighbors based on user-hyperedge-merchant and user-hyperedge-item connections, respectively, and $\boldsymbol{s}_j$ is the representation of corresponding hyperedge type, i.e., scene. Hence, the scene-specific information will be amplified by the gating $\boldsymbol{s}_j$, while irrelevant information will be blocked. Now, we have three type-level embeddings $\boldsymbol{u}_b, \boldsymbol{u}_{sm}$ and $\boldsymbol{u}_{si}$ for user node u. Then we utilize a Multi-Layer Perception (MLP) to fuse them together, and update it to the raw user embedding $\boldsymbol{u}$, formally,

$$\boldsymbol{u}' = \boldsymbol{u} + MLP(\boldsymbol{u}_b \oplus \boldsymbol{u}_{sm} \oplus \boldsymbol{u}_{si}), \tag{6}$$

where $\oplus$ denotes the operation of concatenation.

Merchant Modeling. If the central node is a merchant node, it has four types of connections: merchant-clicked-user, merchant-sell-item, merchant-hyperedge-user and merchant-hyperedge-item. For merchant-clicked-user and merchant-sell-item connections, we also use average function to obtain the type-level embeddings $\boldsymbol{m}_u$ and $\boldsymbol{m}_i$. For merchant-hyperedge-user and merchant-hyperedge-item connections, we similarly apply the scene-aware gate mechanism followed by average function to obtain the hyperedge-based type-level embeddings $\boldsymbol{m}_{su}$ and $\boldsymbol{m}_{si}$. Finally, we apply another MLP to the four type embeddings and update its output to the raw merchant embedding, formally,

$$\boldsymbol{m}' = \boldsymbol{m} + MLP(\boldsymbol{m}_u \oplus \boldsymbol{m}_i \oplus \boldsymbol{m}_{su} \oplus \boldsymbol{m}_{si}). \tag{7}$$

Scene Modeling. According to observation (3), even two unconnected users (or merchants) will have some common characteristics because some of their belonging interactions are of the same scene. A straightforward solution is to design a cross-hyperedge propagation, which allows information flowing along with the type of hyperedges. However, this will lead to an inefficient computing process. Since each user/merchant is not bounded to a certain scene, we need to calculate the user/merchant representations based on all possible scenes. If this process is computed dynamically, it will extremely reduce the online efficiency. If we pre-calculate the user/merchant embeddings of all candidate scenes, it will require large storage space, i.e., each user/merchant needs to store $|S|$ embeddings.

Therefore, we explore another solution instead: since the common characteristics among the user/merchants are scene-specific, we could just incorporate them into the scene representation. Similarly, denoting $\boldsymbol{u}_j$ and $\boldsymbol{m}_j$ as the users/merchants included into the hyperedge assigned with scene $s \in S$, we have

$$\boldsymbol{s}_u = \mathrm{AVG}(\{\boldsymbol{u}_j \odot \boldsymbol{s} | j = 1, 2, \cdots, n\}), \tag{8}$$
$$\boldsymbol{s}_m = \mathrm{AVG}(\{\boldsymbol{m}_j \odot \boldsymbol{s} | j = 1, 2, \cdots, n\}), \tag{9}$$
$$\boldsymbol{s}' = \boldsymbol{s} + MLP(\boldsymbol{s}_u \oplus \boldsymbol{s}_m). \tag{10}$$

3.4 Separable Score Function

After obtaining the enriched embeddings of users, merchants and scenes, here we focus on how to evaluate the match score among the triple <user, scene, merchant> and the corresponding interaction environments. To avoid searching for the best from all the possible triples, we need to design a score function $f(m|c,u;s) = P(m|c,u;s)$ which meets the following requirements:

Separability. f should be separable: it can be split into a two-way subfunction $g(s|c,u)$ to predict match score between the user and candidate scenes, while f further predicts match score among user, scene and merchant.

Reusability. f and g should be reusable: when calculating f, some calculation results of g should be available and useful to improve online efficiency.

Consistency. f and g should be consistent: if f returns a large score, g should also return a relatively large score. This ensures that the "correct" merchant will not be directly filtered out when g selects the related scenes.

Here we discuss the two most widely used score functions: MLP and inner product. The former usually obtains a satisfactory performance due to strong fitting capability, but it is not separable due to its deep structure. Noticing the latter is actually the sum of element-wise multiplication of two vectors, it is easy to expand for multiple inputs and meanwhile keep reusable. Formally, we have

$$g(s|c,u) = \mathrm{sum}(\boldsymbol{c} \odot \boldsymbol{u} \odot \boldsymbol{s}), \qquad f(m|c,u;s) \quad = \mathrm{sum}(\boldsymbol{c} \odot \boldsymbol{u} \odot \boldsymbol{s} \odot \boldsymbol{m}), \quad (11)$$

where each element of $\boldsymbol{c}, \boldsymbol{u}, \boldsymbol{m}$ and $\boldsymbol{s}$ is constrained to be non-negative real numbers to meet consistency requirement. In our work, we apply hard sigmoid function before the element-wise product to satisfy the non-negative condition. In the following, we present a brief proof for the above claim.

Theorem 1. *If there exists δ subject to $f \geq \delta$, g will have a low bound with the non-negative constraint.*

Proof. Considering the j-th dimension of the d-dimensional vectors $\boldsymbol{c}$, $\boldsymbol{u}$, $\boldsymbol{s}$ and $\boldsymbol{m}$, suppose the upper bound of m_j is σ_j. Then, we have $c_j, u_j, t_j, m_j \geq 0$ and $m_j \leq \sigma_i$. Suppose there exists a positive real number δ_j subject to $c_j \cdot u_j \cdot s_j \cdot m_j \geq \delta_j > 0$. Then we get $c_j \cdot u_j \cdot s_j \geq \frac{\delta_j}{m_j} \geq \frac{\delta_j}{\sigma_j}$. Therefore, for the entire vector, we have $g = \sum_{j=1}^{d} c_j \cdot u_j \cdot t_j \geq \sum_{j=1}^{d} \frac{\delta_j}{\sigma_j} \geq \frac{\sum_{j=1}^{d} \delta_j}{\max_j(\sigma_j)} = \frac{\delta}{\sigma}$. Here, $\delta = \sum_{j=1}^{d} \delta_j$ and $\sigma = \max_j(\sigma_j)$. That is, if we have $f \geq \delta$, g will have a low bound δ/σ.

Classification Setting & Model Training. Through the above modules, we can obtain the match score among user, scene and merchant under interaction environments by function f. In other words, in this case, the entire interaction and the scene are already given, hence we can directly predict their matching scores. We name this setting "classification", which is, following existing methods [7], also applied for model training. Formally, given a sample $< m, c, u; s >$:

$$\hat{y} = \mathrm{sigmoid}(f(m|c,u;s)), \quad (12)$$

$$\mathcal{L} = \sum_{j \in \mathcal{Y}^+ \cup \mathcal{Y}^-} (y_j \log \hat{y}_j + (1 - y_j) \log(1 - \hat{y}_j)), \quad (13)$$

Table 1. Statistics of Datasets

Dataset	#Train	#Test	Train User/Merchant/Item/Scene	Test User/Merchant/Item/Scene
1-day	1.13M	1.19M	153K / 32K / 409K / 263	160K / 33K / 427K / 263
3-day	3.51M	1.25M	381K / 41K / 892K / 283	168K / 33K / 453K / 265
5-day	6.13M	1.32M	589K / 45K / 1.27M / 288	172K / 33K / 479K / 257
7-day	8.55M	1.15M	746K / 47K / 1.54M / 294	155K / 33K / 419K / 258

where y_j and $\hat{y}_j$ are the true label and prediction of the sample j. $\mathcal{Y}^+$ and $\mathcal{Y}^-$ are the positive and negative instance sets, respectively. The set of negative instances is composed of training triples with either the user, scene or merchant replaced. We will discuss in detail our negative sampling strategy in Section 4.2.

Inference Setting. As mentioned before, in practical applications, we will first recall highly related k_s scenes based on a given user and interaction environments with g. Then the most related k_m merchants based on each selected scene above can be recommended with function f. Finally, we will recommend $k_s \cdot k_m$ merchants in all. We name this procedure "inference", formally,

$$\{\hat{s}_j | j = 1, 2, \cdots, k_s\} = \mathrm{Top}_{k_s}\ g(s|c, u), \tag{14}$$

$$\{\hat{m}_j | j = 1, 2, \cdots, k_m\} = \cup_{\hat{s} \in \{\hat{s}_j\}} \mathrm{Top}_{k_m}\ f(m|c, u; \hat{s}). \tag{15}$$

4 Experiments

4.1 Experimental Setup

Datasets. A real-world large-scale dataset is built from the food delivery industry, i.e., Meituan Waimai platform. We collect 8-day orders of user purchases of foods and the corresponding click records before the purchase in Beijing District. Each order is an interaction instance, mostly containing a user, a merchant, several items and corresponding interaction environments, and has been already tagged with a scene based on some hand-craft rules and manual efforts. Then we use these orders to build a heterogeneous scene hypergraph as described in Section 3.1. For better validation, we split the whole data into several different scales of data: we use different periods (from 1 to 7 days) as the training data (about 10% data from the training set is extracted for validation) and predict the next one day. Therefore, we have four datasets marked as **1-day**, **3-day**, **5-day** and **7-day**. To get robust results, we vary the size of each training set from 50% to 100%. The detailed statistics of the data are reported in Table 1.

Baselines. We compare SREC with the following four groups of methods and their variants: recommendation methods without scenes: AutoInt [19], NIRec [12]; variant recommendation methods with scenes: AutoInt$_S$, MEIRec [7], NIRec$_S$; graph embedding methods: HAN [25], HGAT [14]; hypergraph-based methods: Hyper-SAGNN [31], Hyper-SAGNN$_S$. They are detailed as follows: AutoInt is SOTA feature-based Click-Through-Rate model. MEIRec and NIRec are meta-path-guided heterogeneous graph neural network based approaches for context-aware recommendation, while HAN and HGAT are both SOTA heterogeneous

Table 2. AUC comparisons of different methods. The last row indicates the improvements (%) compared to the best baseline (underlined).

Method	1-day			3-day			5-day			7-day		
	50%	75%	100%	50%	75%	100%	50%	75%	100%	50%	75%	100%
AutoInt	.5785	.5868	.5891	.5841	.5838	.5846	.5884	.5904	.5895	.5839	.5834	.5839
NIRec	.6589	.6712	.6877	.7075	.7295	.7433	.7319	.7534	.7719	.7502	.7664	.7742
AutoInt$_S$	.6781	.6803	.6892	.6842	.6913	.6987	.7135	.7198	.7203	.6906	.6939	.7001
MEIRec	.7497	.7565	.7634	.7393	.7567	.7658	.7814	.7761	.7956	.7956	.7851	.7891
NIRec$_S$	.7118	.7597	.7660	.7880	.7975	.8072	.8138	.8151	.8213	.8342	.8398	.8390
HAN	.7459	.7569	.7652	.7138	.7230	.7310	.8103	.8193	.8258	.8132	.8137	.8144
HGAT	.7471	.7584	.7675	.7627	.7667	.7697	.7846	.7839	.7923	.8255	.8336	.8390
Hyper-SAGNN	.7061	.7130	.7133	.7270	.7327	.7378	.7318	.7350	.7360	.7560	.7591	.7583
Hyper-SAGNN$_S$	.7579	.7660	.7649	.7882	.7944	.7980	.8147	.8186	.8176	.8350	.8367	.8340
SREC	**.8083**	**.8087**	**.8097**	**.8695**	**.8702**	**.8787**	**.8944**	**.8915**	**.8970**	**.8992**	**.9032**	**.8995**
Improvement	6.64	5.57	5.50	10.32	9.11	8.85	9.79	8.80	8.62	7.68	7.55	7.21

graph embedding models. Hyper-SAGNN is a SOTA hypergraph-based model for link prediction. For fair comparisons, we further modify the above methods to be accessible to the same information as ours (such as interaction environments, scenes, etc.) since the absence of any information harms the performance. Particularly, we add a channel of scene information to AutoInt and Hyper-SAGNN, denoting as AutoInt$_{\boldsymbol{S}}$ and Hyper-SAGNN$_{\boldsymbol{S}}$. For graph-based models, we transform the hypergraph into a heterogeneous graph: a summary node is introduced for each interaction instance, whose type and attributes depend on the corresponding scene and interaction environments, i.e., we transform the hyperedges into summary nodes, and link the summary nodes with the nodes related the corresponding interaction instance. For meta-path based methods, we select the following meta-paths based on experiments: UMU, USU, UMIMU, MUM, MIM, MSM, SUS, SMS, SIS.

Settings and Metrics. We evaluate model performance for both settings. For classification setting, following previous work [7], we use Area Under receiver operator characteristic Curve (AUC) for evaluation. We use HR@K (Hit Ratio) to evaluate under the inference setting. Without loss of generality, we set K as 10 (k_s) for scenes and 100 ($k_s \cdot k_m$) for merchants in this work, denoting HR@10-S and HR@100-M, respectively. This is computed only on the positive sample set.

Detailed Implementation. For our method, we set the dimension of attribute embeddings, linear transformations and 2-layer MLPs all as 64. We use 8 hidden neurons and 8 heads in the transformer. For all baselines, we also set the hidden dimensions as 64 for fair comparison. We set the batch size as 2048, and set the learning rate as 0.0001 with Adam optimizer.

4.2 Experimental Results

Main Results We evaluate the AUC performance based on the constructed negative samples in Table 2. We can conclude as follows: (1) SREC significantly outperforms all the competitive baselines. Compared to the best performance of baselines, SREC gains 5.50% - 10.32% improvement in the four datasets.

Table 3. HR@K comparisons for the recall of scenes and merchants.

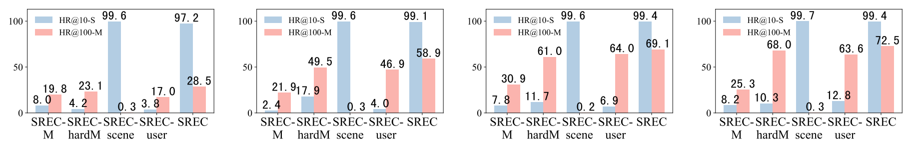
Method		1-day			3-day			5-day			7-day		
		50%	75%	100%	50%	75%	100%	50%	75%	100%	50%	75%	100%
HAN	HR@10-S	.3910	.3678	.4148	.3652	.3608	.3809	.1455	.1904	.2655	.2604	.2858	.2493
	HR@100-M	.0123	.0505	.1048	.0392	.0728	.0949	.0107	.0307	.1678	.0614	.0881	.1079
HGAT	HR@10-S	.3696	.3851	.4284	.3598	.3324	.3910	.1123	.1398	.1609	.2690	.2542	.2821
	HR@100-M	.0796	.0549	.0950	.0336	.0325	.0581	.0235	.0613	.0411	.0643	.1274	.0919
SREC	HR@10-S	**.9588**	**.9756**	**.9720**	**.9884**	**.9922**	**.9913**	**.9927**	**.9920**	**.9936**	**.9926**	**.9913**	**.9936**
	HR@100-M	**.2576**	**.2615**	**.2849**	**.5234**	**.5439**	**.5886**	**.6560**	**.6916**	**.6912**	**.7113**	**.7503**	**.7249**

Moreover, when fewer training instances (50%, 75%) are provided, our SREC achieves a higher improvement. It indicates the effectiveness of our completely modeling of interactions and integrating scene prior knowledge, which makes our model more robust. (2) The scene prior knowledge is greatly useful. Compared with the methods access to scene prior knowledge, those without scene prior knowledge exhibit an obvious performance drop. It demonstrates *the superiority to considerate scenes than to direct recommend merchants and the significance of this scene-aware recommendation problem.* (3) The more accurate and comprehensive the interaction modeling is, the better the performance is. In detail, the graph-based models are better than the traditional feature-based model due to the consideration of the interactive relations between entities. Next, the hypergraph-based model HYPER-SAGNN$_S$ further performs better in most cases, which verifies the better modeling of entire interactions within scenes as hyperedges, since introducing a summary node cannot correctly model the entire interaction. Moreover, the huge performance gap between HYPER-SAGNN$_S$ and SREC verifies the necessity of our scene-aware gate mechanism to distinguish the mixed scene-related information.

Evaluation of Inference In practice, we cannot search for the best from all the possible <user, scene, merchant> triples in terms of efficiency, thus requiring the inference setting. Here we measure the inference performances for scenes and merchants, i.e., HR@10-S and HR@100-M. Since only HAN and HGAT support the inference procedure, which can export relatively fixed embeddings for users, merchants and scenes, we compare our SREC with these two baselines on the four datasets. As reported in Table 3, our method consistently outperforms greatly in terms of both scenes and merchants, while HAN and HGAT only achieve limited and unrobust performance for inference. We believe this huge performance gap is caused by the following reasons: (1) The three requirements of our separable score function play a vital role to ensure the effective inference procedure. (2) It is essential to comprehensively model the entire interactions by hyperedges. Introducing summary node for each interaction will forcibly divide the indivisible whole interaction relation into several pair-wise sub-relations, resulting in information loss. (3) The information relevant to different scenes is mixed in every single entity. For a particular scene, the information relevant to other scenes becomes noise on the contrary.

Table 4. AUC and HR@100-M comparisons of our variants.

Variant	1-day		3-day		5-day		7-day	
	AUC	HR@100	AUC	HR@100	AUC	HR@100	AUC	HR@100
SREC	.8097	**.2849**	.8787	**.5886**	.8970	**.6912**	.8995	**.7249**
SREC \ GNN	.7649	.1981	.7691	.2266	.7844	.3171	.7726	.3929
SREC \ Scene	.7493	.2071	.8072	.2757	.8656	.4375	.8811	.4984
SREC \ Gate	.8002	.2113	.8122	.3388	.8664	.6164	.8888	.6062
SREC-ReLU	.5107	.0019	.5010	.0016	.5134	.0018	.5029	.0010
SREC-None	**.8960**	.0034	**.9213**	.0057	**.9219**	.0112	**.9246**	.0054



(a) 1-day (b) 3-day (c) 5-day (d) 7-day

Fig. 3. HR@K (%) of SREC with different negative sampling strategies.

Comparison of Variants We compare SREC with 2 groups of variants to validate the design of its modules: One group is used to verify the effectiveness of our propagation rule. The other group aims to verify the design of constraints in our separable score function. As reported in Table 4, we can draw the following conclusions. Firstly, the performances of SREC\GNN, SREC\Scene and SREC\Gate, which are removed any neighboring information, scene type-specific information or scene-aware gate mechanism respectively, are limited in terms of both AUC and HR metrics, thus verifying the design of our propagation rule in gated hypergraph neural network. Secondly, ReLU, as a straightforward solution to for non-negative constraint, cannot improve the performance if replace hardsigmoid activation in the separable score function with it, i.e., SREC-ReLU. Even worse, ReLU will cause large-scale death of neurons and thus the model cannot be trained, because its gradient in the negative range is 0 and the operation of element-wise product among multiple vectors will worsen this phenomenon. Moreover, the AUC metric gets better if the activation function hardsigmoid is directly removed, i.e., SREC-None, but the HR@K performance obtains a severe drop. Because the embedding space is limited by the hardsigmoid function, thereby removing it will strengthen the ability of fitting data (refer to the improvement on AUC). However, this constraint is indispensable to ensure the successful recall (refer to the performance drop on HR@K).

Impact of Negative Sampling Strategy In our task, the model need learn based on triples <user, scene, merchant>, thus requiring a best negative sampling strategy. We have explored four negative sampling strategies: For each triple, SREC-M randomly replaces the merchant without any constraint; SREC-hardM replaces the merchant under the constraints of the same interaction environments; SREC-scene randomly replaces the scene; SREC-user replaces the user also under interaction environment constraints. The AUC metric is meaningless since the negative instances are different. Therefore, we choose HR@K metric on the positive instances in the test set for evaluation. As depicted in Figure 3, the poor performance of SREC-M indicates that without any constraint is too simple to generate negative samples, while the performance of SREC-hardM is much better. Moreover, the result of SREC-scene shows that it is helpful for improving the HR@10-S to directly replace the scene, but it causes a lot of outrageous scene-merchant combinations, such as nutritious breakfast is paired with a barbecue restaurant. In our testing, we find SREC-user can effectively help correct scene-merchant pairing, although its effect is not ideal. Therefore, we combine the above three strategies to construct negative samples, to improve both the scene recalling and merchant recommendation. As shown in Figure 3, this strategy combination at the expense of a little loss of HR@10-S, in exchange for a great improvement in HR@100-M.

5 Related Work

We first introduce some related recommendation methods, and then discuss the recent hypergraph-based methods.

Researchers have always been exploring the interactions between users and items for better recommendation. Since the classic methods such as matrix factorization [18] suffer from cold start and data sparsity problems [24], many kinds of interaction information are studied, e.g., social friendships [17, 10], location relationships [2, 26, 22], contextual information [21, 4, 3, 5], etc. HIN-based recommendation is then proposed to integrate any type of interaction information [24, 12], attracting more research interests recently. However, the interaction process in real applications is still much richer and more complex, and these methods only explore some certain components of the entire interactions.

Hypergraph expands the concept of edge in graph into hyperedge that connects multiple nodes, thus having a strong ability to model a complex interaction among multiple entities [9]. HGNN [8] and HyperGCN [29] were the first to expand graph convolution to hypergraph, thereby inspiring researchers' enthusiasm for hypergraph neural network [1, 20, 31]. Recently, some researchers have begun to explore hypergraph-based methods for recommendation. For example, DHCF [11] developed a dual-channel learning strategy based on hypergraph that explicitly defined the hybrid high-order correlations, while [30] further integrates self-supervised learning into the training of the hypergraph convolutional network. There are also studies focusing on finer-grained recommendations, such as HyperRec for next-item recommendation [23], DHCN for session-based recommendation [28], etc. However, these methods still fail to comprehensively model the user's behaviors, thereby they can neither leverage the scene prior knowledge, limiting the performance of recommendation. Consequently, we are the first to fill this gap: A novel gated hypergraph neural network is proposed in this paper for scene-aware recommendation.

6 Conclusion

In this paper, we make the first attempt to study the scene-aware recommendation, where the entire interactions are modeled comprehensively and the scenes are introduced to guide the recommendation. Specifically, we propose a novel gated hypergraph neural network for scene-aware recommendation (SREC). It first constructs a heterogeneous scene hypergraph to comprehensively model the entire interactions and incorporate the scene prior knowledge, followed by a novel gated hypergraph neural network to learn the representations of users, merchants and scenes. Finally, it designs a separable score function to predict the match score, thus recommending the merchants. Extensive experiments demonstrate our SREC outperforms SOTA methods for both classification and inference on the real industrial datasets. In the future, we will explore clustering approaches for automatically discovering and tagging scenes in an unsupervised manner to reduce the costs of expert annotations.

Acknowledgments

This work is supported in part by the National Natural Science Foundation of China (No. U20B2045, 62192784, 62172052, 61772082, 62002029) and also supported by Meituan.

References

1. Bandyopadhyay, S., Das, K., Murty, M.N.: Line Hypergraph Convolution Network: Applying Graph Convolution for Hypergraphs. arXiv (2020)
2. Chang, B., Jang, G., Kim, S., Kang, J.: Learning Graph-Based Geographical Latent Representation for Point-of-Interest Recommendation. In: CIKM (2020)
3. Chen, C., Zhang, M., Ma, W., Liu, Y., Ma, S.: Efficient non-sampling factorization machines for optimal context-aware recommendation. In: WWW (2020)
4. Chen, H., Li, J.: Adversarial tensor factorization for context-aware recommendation. In: RecSys. pp. 363–367 (2019)
5. Chen, L., Xia, M.: A context-aware recommendation approach based on feature selection. APIN **51**(2), 865–875 (2021)
6. Da'u, A., Salim, N., Idris, R.: Multi-level attentive deep user-item representation learning for recommendation system. Neurocomputing **433**, 119–130 (2021)
7. Fan, S., Zhu, J., Han, X., Shi, C., Hu, L., Ma, B., Li, Y.: Metapath-guided Heterogeneous Graph Neural Network for Intent Recommendation. In: KDD (2019)
8. Feng, Y., You, H., Zhang, Z., Ji, R., Gao, Y.: Hypergraph neural networks. In: AAAI. pp. 3558–3565 (2019)
9. Gao, Y., Zhang, Z., Lin, H., Zhao, X., Du, S., Zou, C.: Hypergraph Learning: Methods and Practices. TPAMI (2020)
10. Huang, C., Xu, H., Xu, Y., Dai, P., Xia, L., Lu, M., Bo, L., Xing, H., Lai, X., Ye, Y.: Knowledge-Aware Coupled Graph Neural Network for Social Recommendation. In: AAAI (2021)
11. Ji, S., Feng, Y., Ji, R., Zhao, X., Tang, W., Gao, Y.: Dual Channel Hypergraph Collaborative Filtering. In: KDD. pp. 2020–2029 (2020)

12. Jin, J., Qin, J., Fang, Y., Du, K., Zhang, W., Yu, Y., Zhang, Z., Smola, A.J.: An Efficient Neighborhood-based Interaction Model for Recommendation on Heterogeneous Graph. In: KDD. pp. 75–84 (2020)
13. Kou, F., Du, J., He, Y., Ye, L.: Social network search based on semantic analysis and learning. CAAI transactions on intelligence technology **1**(4), 293–302 (2016)
14. Linmei, H., Yang, T., Shi, C., Ji, H., Li, X.: Heterogeneous Graph Attention Networks for Semi-supervised Short Text Classification. In: EMNLP (2019)
15. Luo, X., Liu, L., Yang, Y., Bo, L., Cao, Y., Wu, J., Li, Q., Yang, K., Zhu, K.Q.: AliCoCo: Alibaba E-commerce Cognitive Concept Net. In: SIGMOD (2020)
16. Luo, X., Yang, Y., Zhu, K.Q., Gong, Y., Yang, K.: Conceptualize and Infer User Needs in E-commerce. In: CIKM. pp. 2517–2525 (2019)
17. Ma, H., Zhou, T.C., Lyu, M.R., King, I.: Improving Recommender Systems by Incorporating Social Contextual Information. TOIS **29**(2), 1–23 (2011)
18. Rendle, S., Freudenthaler, C., Gantner, Z., Schmidt-Thieme, L.: BPR: Bayesian Personalized Ranking from Implicit Feedback. In: UAI. pp. 452–461 (2009)
19. Song, W., Shi, C., Xiao, Z., Duan, Z., Xu, Y., Zhang, M., Tang, J.: AutoInt: Automatic Feature Interaction Learning via Self-Attentive Neural Networks. In: CIKM. pp. 1161–1170 (2019)
20. Tu, K., Cui, P., Wang, X., Wang, F., Zhu, W.: Structural deep embedding for hyper-networks. In: AAAI. pp. 426–433 (2018)
21. University of Ottawa, Ottawa, K1N 6N5, Canada, Agagu, T., Tran, T.: Context-Aware Recommendation Methods. IJISA **10**(9), 1–12 (2018)
22. Wang, H., Li, P., Liu, Y., Shao, J.: Towards real-time demand-aware sequential POI recommendation. Information Sciences **547**, 482–497 (2021)
23. Wang, J., Ding, K., Hong, L., Liu, H., Caverlee, J.: Next-item Recommendation with Sequential Hypergraphs. In: SIGIR. pp. 1101–1110 (2020)
24. Wang, X., Bo, D., Shi, C., Fan, S., Ye, Y., Yu, P.S.: A Survey on Heterogeneous Graph Embedding: Methods, Techniques, Applications and Sources. arXiv (2020)
25. Wang, X., Ji, H., Shi, C., Wang, B., Ye, Y., Cui, P., Yu, P.S.: Heterogeneous Graph Attention Network. In: WWW (2019)
26. Werneck, H., Silva, N., Viana, M.C., Mourão, F., Pereira, A.C.M., Rocha, L.: A Survey on Point-of-Interest Recommendation in Location-based Social Networks. In: WebMedia. pp. 185–192 (2020)
27. Wu, S., Zhang, W., Sun, F., Cui, B.: Graph Neural Networks in Recommender Systems: A Survey. arXiv (2020)
28. Xia, X., Yin, H., Yu, J., Wang, Q., Cui, L., Zhang, X.: Self-Supervised Hypergraph Convolutional Networks for Session-based Recommendation. In: AAAI (2021)
29. Yadati, N., Nimishakavi, M., Yadav, P., Nitin, V., Louis, A., Talukdar, P.: HyperGCN: A New Method For Training Graph Convolutional Networks on Hypergraphs. In: NeurIPS, pp. 1511–1522 (2019)
30. Yu, J., Yin, H., Li, J., Wang, Q., Hung, N.Q.V., Zhang, X.: Self-supervised multi-channel hypergraph convolutional network for social recommendation. In: WWW. p. 413–424 (2021)
31. Zhang, R., Zou, Y., Ma, J.: Hyper-SAGNN: a self-attention based graph neural network for hypergraphs. In: ICLR (2020)