Automatically Discovering User Consumption Intents in Meituan

Yinfeng Li*
Department of Electronic Engineering, Tsinghua University, Beijing, China

Chen Gao†
Department of Electronic Engineering, Tsinghua University, Beijing, China

Xiaoyi Du
Meituan Inc., Beijing, China

Huazhou Wei
Meituan Inc., Beijing, China

Hengliang Luo
Meituan Inc., Beijing, China

Depeng Jin
Department of Electronic Engineering, Tsinghua University, Beijing, China

Yong Li
Department of Electronic Engineering, Tsinghua University, Beijing, China

ABSTRACT

Consumption intent, defined as the decision-driven force of consumption behaviors, is crucial for improving the explainability and performance of user-modeling systems, with various downstream applications like recommendation and targeted marketing. However, consumption intent is implicit, and only a few known intents have been explored from the user consumption data in Meituan. Hence, discovering new consumption intents is a crucial but challenging task, which suffers from two critical challenges: 1) how to encode the consumption intent related to multiple aspects of preferences, and 2) how to discover the new intents with only a few known ones. In Meituan, we designed the AutoIntent system, consisting of the disentangled intent encoders and intent discovery decoders, to address the above challenges. Specifically, for the disentangled intent encoders, we construct three groups of dual hypergraphs to capture the high-order relations under the three aspects of preferences and then utilize the designed hypergraph neural networks to extract disentangled intent features. For the intent discovery decoders, we propose to build intent-pair pseudo labels based on the denoised feature similarities to transfer knowledge from known intents to new ones. Extensive evaluations verify that AutoIntent can effectively discover unknown consumption intents. Moreover, experiments also demonstrate that AutoIntent can effectively enhance the downstream recommendation.

CCS CONCEPTS

• **Information systems** → **Information systems applications**;

KEYWORDS

Consumption Intents Discovery; Graph Neural Networks; Self-supervised Learning; Disentangled Representation Learning

ACM Reference Format:
Yinfeng Li*, Chen Gao†, Xiaoyi Du, Huazhou Wei, Hengliang Luo, Depeng Jin, and Yong Li. 2022. Automatically Discovering User Consumption Intents in Meituan. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '22), August 14–18, 2022, Washington, DC, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3534678.3539122

1 INTRODUCTION

Most existing models in industrial recommendation engines work as a black-box, without explicit modeling of why a user behavior occurs. To address it, a possible solution is to detect why a user makes that decision, which can be defined as *intent*. For example, in local life service platforms such as Meituan[1], an intent of *family gathering* may leads to a consumption behavior of *movie ticket*. In other words, a consumption intent can be understood as a group or a typical pattern of user consumption behaviors. Hence, consumption intents can support many downstream applications, such as item recommendation, target-user marketing, supply chain optimization, etc. In the real-world scenarios of Meituan, practitioners can obtain a small fraction of intents based on expert knowledge and user reviews [28]. A user may have written a short review including "*family gathering*" after a consumption behavior of *movie ticket*, and then practitioners can obtain an intent label for that behavior. However, the reviewing data is always sparse, and the behaviors along with reviews only take a tiny percentage of all behaviors, as revealed by industrial practice and academic datasets [25].

*Work done when interning at Meituan.
†Chen Gao is the corresponding author (chgao96@gmail.com).

KDD '22, August 14–18, 2022, Washington, DC, USA

ACM ISBN 978-1-4503-9385-0/22/08…$15.00
https://doi.org/10.1145/3534678.3539122

[1]https://about.meituan.com/en

In this work, we approach the problem of intent discovery, which aims to assign intent labels (known + new) for the unlabeled consumption behavior data with a tiny fraction of labeled data. There are two critical challenges when solving this problem.

- **Consumption intent is related to multiple aspects of user preferences (how to encode).** Consumption is not only determined by users' intrinsic preferences like price and brand but also largely affected by spatial and temporal factors, especially for local life service platforms like Meituan. For example, the intent of *family gathering* tends to occur at night and not to occur around the office area.
- **Learning from the unlabeled data along with only a few labeled data (how to discover).** Since there is only a few labeled data, the intent discovery problem is faced with the challenge of transferring the knowledge of labeled data into unlabeled data and extracting self-supervision signal from unlabeled data.

To address the above challenges, we propose a system named AutoIntent (short for **Auto**matically Consumption **Intent** Discovery), consisting of two main parts, 1) disentangled intent encoders and 2) intent discovery decoders. Specifically, we first construct three groups of dual hypergraphs to represent the relations among user intrinsic preferences, spatial factor, and temporal factor, respectively. We then deploy a dual-hypergraph neural networks model to extract high-order relations and obtain disentangled intent representations. With the disentangled intent encoders, we address the first challenge. As for the second challenge, we propose to first warm up the intent discovery decoders with intent number estimation and feature fine-tuning. We then propose to build intent-pair pseudo labels based on the denoised feature similarities, which can be regarded as a self-supervision signal. Finally, We design a joint-learning framework to discover new intents, which can well transfer knowledge from known intents to unknown ones. With the discovered intents, we further explored the possible applications in Meituan. AutoIntent can serve as an essential component in Meituan's user-modeling system, with various downstream applications like recommendation and targeted marketing. Hence, we deploy AutoIntent in the recommendation engine of the Meituan APP to further verify the effectiveness of the AutoIntent system. The contributions of this work can be summarized as follows.

- To the best of our knowledge, we take the pioneering step to approach the problem of consumption intent discovery, which is critical for various industrial user personalized services, such as recommendation, targeting marketing, etc.
- We develop an AutoIntent system, which consists of two parts: **1) disentangled intent encoders** to learn the disentangled intent representations with and **2) intent discovery decoders** to discover the new intents with knowledge transfer.
- We evaluate our system on both intent discovery and downstream recommendation tasks. Experimental results on two real-world datasets verify that AutoIntent can effectively discover unknown intents. The downstream evaluations further confirm that AutoIntent can enhance the recommendation in Meituan APP.

2 PROBLEM STATEMENT

The i-th record of user consumption behavior can be denoted as $x_i = (u_i, l_i, t_i, c_i)$, which means user u_i bought the item with category c_i on location l_i at time-slot t_i, where u_i, l_i, t_i, c_i denote the user ID, location ID, time-slot ID and item category ID, respectively. Let $\mathcal{U}$, $\mathcal{L}, \mathcal{T}, C$ denote the sets of users, locations, time-slots and categories, of which the sizes are denoted as N_U, N_L, N_T, and N_C, respectively. As mentioned in the introduction, the user makes a consumption behavior due to a specific intent. For each consumption behavior x_i, we use y_i to denote the associated intent. In real world, we can only manually define a limited set of intents, with a small amount of *labeled* consumption behaviors $\mathcal{D}^l = \{(x_i^l, y_i^l)|_{i=1}^M\}$, where $y_i^l \in \mathcal{I}^k$ ($\mathcal{I}^k$ denotes the known intents with K^k classes). For the *unlabeled* data $\mathcal{D}^u = \{x_i^u|_{i=1}^N\}$, the intents of $\mathcal{D}^u$ may belong to the unknown intents $\mathcal{I}^u$ or the known ones $\mathcal{I}^k$.

Given the user consumption data $\mathcal{D} = \mathcal{D}^l \cup \mathcal{D}^u$, intent discovery aims to automatically cluster the unlabeled data $\mathcal{D}^u$ into a number of intents classes ($\mathcal{I}^k \cup \mathcal{I}^u$) by transferring knowledge from the labeled data $\mathcal{D}^l$. In other words, we assign each consumption behavior a label from known intents $\mathcal{I}^k$ or unknown ones $\mathcal{I}^u$.

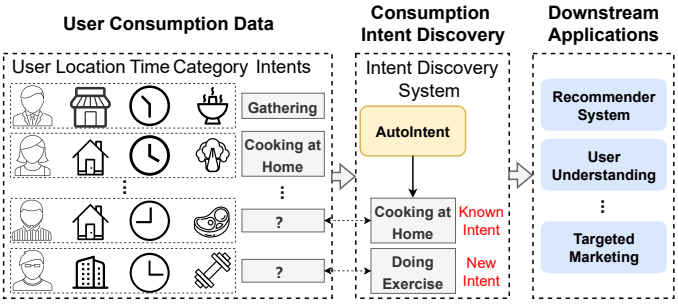


Figure 1: Illustration of the intent discovery and its applications in Meituan.

3 THE AUTOINTENT SYSTEM

Figure 1 illustrates the intent discovery and its applications in Meituan. In this section, we first introduce our proposed AutoIntent model, and then introduce how to deploy it in the recommendation engine of the Meituan APP. To address the challenges in the introduction, we propose AutoIntent, illustrated in Figure 3 (a), which consists of the following parts: **1) Disentangled Intent Encoders** to sufficiently model consumption intent in multiple aspects and **2) Intent Discovery Decoders** to discover the new intents by transferring knowledge from known intents to unknown ones.

3.1 Disentangled Intent Encoders

The user consumption behaviors in life-service platforms, such as Meituan and Yelp, are driven by complex heterogeneous factors. Specifically, user consumption intents are determined by the following three key factors, 1) *intrinsic preference*[2]: the users preference towards the attributes of the item such as taste, price, brand, etc.; 2) *location-aware preference*: the user may have location-related consumption behaviors such as consuming quick food when the user is around office building; 3) *time-aware preference*: the user's consumption behaviors are relevant to the time, such as consuming at Bar at night.

[2]Note that we capture the *intrinsic preference* with the *user-category* (UC) relation in the proposed AutoIntent system.

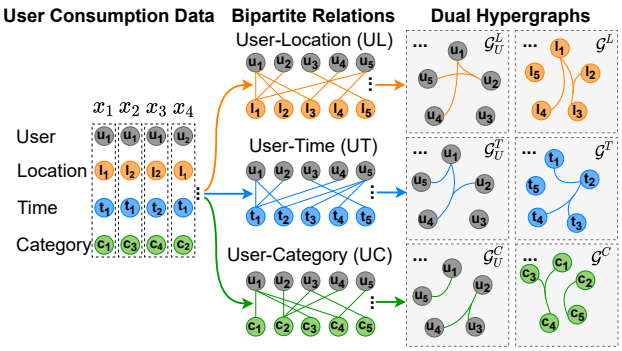


Figure 2: The Construction of Dual Hypergraphs.

To sufficiently utilize the above three preferences, from the *disentangled view* (ensuring we can learn three-aspect disentangled representations of users), we decompose the quadruple user consumption data into three types of bipartite relations, *i.e. user-location* (UL), *user-time* (UT) and *user-category* (UC). For example, a user consumption data $x_i = (u_i, l_i, t_i, c_i)$ contains three types of bipartite relations, (u_i, l_i) for location preference, (u_i, t_i) for time preference, and (u_i, c_i) for category preference. Given that the dual hypergraphs can naturally match the bipartite relations and have the better capability on high-order relations than a normal graph, inspired by the recent advances in dual-hypergraph based bipartite-relation learning [36], we construct three groups of dual hypergraphs to model the above bipartite relations.

3.1.1 Dual Hypergraphs Construction. Let $\{\mathcal{G}_U^L = (\mathcal{U}, \mathcal{E}_U^L), \mathcal{G}_L = (\mathcal{L}, \mathcal{E}_L)\}$, $\{\mathcal{G}_U^T = (\mathcal{U}, \mathcal{E}_U^T), \mathcal{G}_T = (\mathcal{T}, \mathcal{E}_T)\}$, $\{\mathcal{G}_U^C = (\mathcal{U}, \mathcal{E}_U^C), \mathcal{G}_C = (C, \mathcal{E}_C)\}$ denote the dual hypergraphs groups to capture the bipartite relations of *user-location* (UL), *user-time* (UT) and *user-category* (UC), respectively. Note that the hypergraphs $\mathcal{G}_U^L, \mathcal{G}_U^T, \mathcal{G}_U^C$ share the same node set $\mathcal{U}$ but have distinct hyperedges $(\mathcal{E}_U^L, \mathcal{E}_U^T, \mathcal{E}_U^C)$.

Then we elaborate on how to represent the bipartite relations with dual homogeneous hypergraphs. Take *user-category* bipartite relations in Figure 2 as an example, the user u_1 purchases items with categories c_1, c_3, c_4, which corresponds to a hyperedge $\{c_1, c_3, c_4\} \in \mathcal{E}_C$ in $\mathcal{G}_C$. From the perspective of items, the item with categoryc_2 is bought by u_2, u_3, u_4, which forms a hyperedge $\{u_2, u_3, u_4\} \in \mathcal{E}_U^C$ in $\mathcal{G}_U$. In this way, we construct the dual hypergraphs $\{\mathcal{G}_U^C, \mathcal{G}_C\}$ to capture the UC bipartite relation. Similarly, we construct other two groups of dual hypergraphs $\{\mathcal{G}_U^L, \mathcal{G}_L\}$ and $\{\mathcal{G}_U^T, \mathcal{G}_T\}$ to model the bipartite relations of UL and UT, respectively.

Hypergraph extends the concept of *adjacency matrix* in normal graph to *incidence matrix*, $\mathbf{H}$, to represent the connections among more than two nodes. For the constructed homogeneous hypergraph $\mathcal{G}_U^C$, each entry of the incidence matrix $\mathbf{H}_U^C \in \mathbb{R}^{|\mathcal{U}|\times|\mathcal{E}_U^C|}$ can be defined as follows,

$$\mathbf{H}_U^C(u, e) = \begin{cases} 1 & \text{if } u \text{ is connected by } e, e \in \mathcal{E}_U^C, \\ 0 & \text{otherwise.} \end{cases} \tag{1}$$

Further, we use diagonal matrices $\mathbf{D}_U^C \in \mathbb{R}^{|\mathcal{U}|\times|\mathcal{U}|}$ and $\mathbf{B}_U^C \in \mathbb{R}^{|\mathcal{E}_U^C|\times|\mathcal{E}_U^C|}$ to represent the node degrees and hyperedge degrees, where $\mathbf{D}_U^C(u,u) = \sum_{e\in\mathcal{E}_U^C} \mathbf{H}_U^C(u,e)$ and $\mathbf{B}_U^C(e,e) = \sum_{u\in\mathcal{U}} \mathbf{H}_U^C(u,e)$.

Obviously, we can easily generate the incidence matrix $(\mathbf{H}_U^L, \mathbf{H}_U^T, \mathbf{H}_L, \mathbf{H}_T, \mathbf{H}_C)$ for other homogeneous hypergraphs $(\mathcal{G}_U^L, \mathcal{G}_U^T, \mathcal{G}^L, \mathcal{G}^T, \mathcal{G}^C)$ in a similar way.

3.1.2 Embedding Propagation on Dual Hypergraphs. With the constructed dual hypergraphs, to learn representations that capture the high-order relations, we propose hypergraph convolutional layers based on the embedding propagation. Specifically, we first introduce the embedding layer and then conduct the proposed joint hypergraph convolution (Joint-HGC) for aggregation.

Embedding layer. We create four learnable embedding matrices $\mathbf{E}_U \in \mathbb{R}^{|\mathcal{U}|\times d}, \mathbf{E}_L \in \mathbb{R}^{|\mathcal{L}|\times d}, \mathbf{E}_T \in \mathbb{R}^{|\mathcal{T}|\times d}, \mathbf{E}_C \in \mathbb{R}^{|C|\times d}$ for all the users, locations, time-slots and item categories, where d denotes the embedding size. Note that $\mathcal{G}_U^L, \mathcal{G}_U^T, \mathcal{G}_U^C$ share the same nodes but reveals completely different information and semantics (*e.g.* node embeddings in $\mathcal{G}_U^L$ represent users' location preference). Thus, we further transform the original user embedding matrix $\mathbf{E}_u$ into three disentangled sub-spaces to represent users' preferences on location, time and item category, respectively. The above transformation operation on user embedding matrix can be formulated as $\mathbf{E}_{U,s} = \mathbf{E}_U\mathbf{W}^s \in \mathbb{R}^{d\times d}$, where $s \in \{L, T, C\}$ and $\mathbf{W}^s$ denote the sub-spaces and transformation matrix in sub-space s, respectively.

Joint HyperGraph Convolution (Joint-HGC). As for the dual homogeneous hypergraphs constructed from bipartite relations, an intuitive approach of representation learning is the traditional hypergraph convolution networks [8]. However, although it can capture the high-order relations among nodes in each hypergraph (*intra-graph view*), it neglects the relations between dual hypergraphs (*inter-graph view*), which reveals the important interaction information. For example, in the *user-category* (UC) relation, the inter-graph propagation can directly fuse the user embedding and category embedding from distinct hypergraphs $(\mathcal{G}_U^C, \mathcal{G}^C)$ to naturally capture the interaction relations. To address it, we combine both the intra- and inter-graph propagation by the proposed Joint-HGC. Given the dual hypergraphs $\{\mathcal{G}_U, \mathcal{G}_V\}$, the aggregation of Joint-HGC in the $(\ell+1)$-th layer can be formulated as follows,

$$\begin{aligned} \mathbf{X}_U^{(\ell+1)} &= \mathbf{D}_U^{-\frac{1}{2}}\mathbf{H}_U\mathbf{B}_U^{-1}\mathbf{H}_U^\top\mathbf{D}_U^{-\frac{1}{2}}\mathbf{X}_U^{(\ell)} + \mathbf{B}_V^{-1}\mathbf{H}_V^\top\mathbf{X}_V^{(\ell)}, \\ \mathbf{X}_V^{(\ell+1)} &= \underbrace{\mathbf{D}_V^{-\frac{1}{2}}\mathbf{H}_V\mathbf{B}_V^{-1}\mathbf{H}_V^\top\mathbf{D}_V^{-\frac{1}{2}}\mathbf{X}_V^{(\ell)}}_{\text{intra-graph}} + \underbrace{\mathbf{B}_U^{-1}\mathbf{H}_U^\top\mathbf{X}_U^{(\ell)}}_{\text{inter-graph}}, \end{aligned} \tag{2}$$

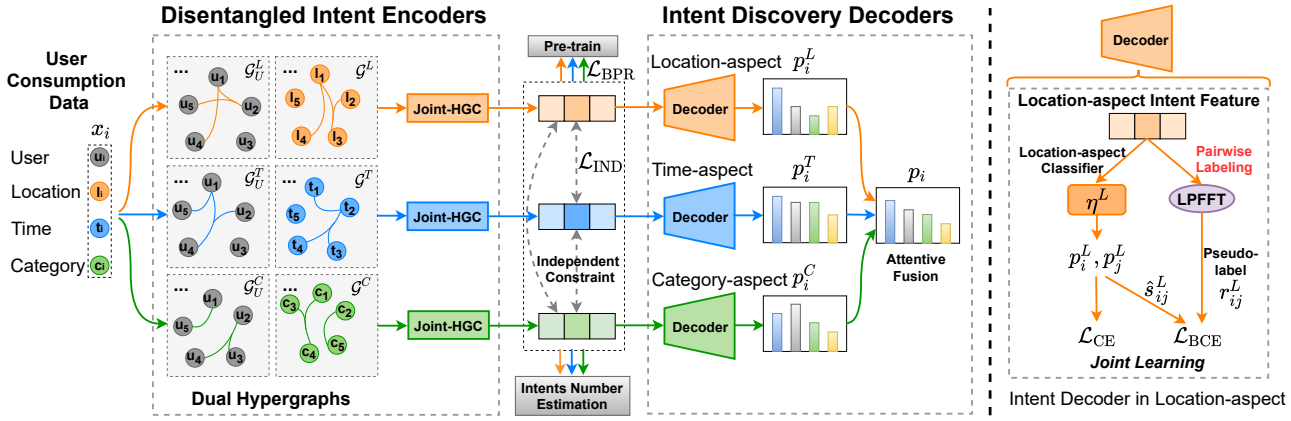
where $\mathbf{H}_U, \mathbf{D}_U, \mathbf{B}_U, \mathbf{X}_U^{(\ell)}$ and $\mathbf{H}_V, \mathbf{D}_V, \mathbf{B}_V, \mathbf{X}_V^{(\ell)}$ denote the incidence matrix, node degree matrix, hyperedge degree matrix, and node features of $\mathcal{G}_U, \mathcal{G}_V$, respectively. Here we remove the nonlinear feature transformations by following [14, 36].

Since $\mathbf{D}, \mathbf{B}$ can be calculated with $\mathbf{H}$, we simplify the formulation Equation (2) by defined a function of Joint-HGC$(\cdot)$ as follows,

$$\mathbf{X}_U^{(\ell+1)}, \mathbf{X}_V^{(\ell+1)} = \text{Joint-HGC}(\mathbf{H}_U, \mathbf{H}_V, \mathbf{X}_U^{(\ell)}, \mathbf{X}_V^{(\ell)}). \tag{3}$$

Then the information aggregation on the three groups of dual hypergraphs can be further formulated as follows,

$$\begin{aligned} \mathbf{E}_{U,L}^{(\ell+1)}, \mathbf{E}_L^{(\ell+1)} &= \text{Joint-HGC}(\mathbf{H}_U^L, \mathbf{H}_L, \mathbf{E}_{U,L}^{(\ell)}, \mathbf{E}_L^{(\ell)}), \\ \mathbf{E}_{U,T}^{(\ell+1)}, \mathbf{E}_T^{(\ell+1)} &= \text{Joint-HGC}(\mathbf{H}_U^T, \mathbf{H}_T, \mathbf{E}_{U,T}^{(\ell)}, \mathbf{E}_T^{(\ell)}), \\ \mathbf{E}_{U,C}^{(\ell+1)}, \mathbf{E}_C^{(\ell+1)} &= \text{Joint-HGC}(\mathbf{H}_U^C, \mathbf{H}_C, \mathbf{E}_{U,C}^{(\ell)}, \mathbf{E}_C^{(\ell)}), \end{aligned} \tag{4}$$



(a) Architecture of AutoIntent Model (b) Decoder Details

Figure 3: The General architecture of the proposed AutoIntent model (a) and the details of intent decoder (b).

where $\{\mathbf{H}_*\}, \{\mathbf{E}_*^{(\ell)}\}$ ($*$ represents subscript) denote the incidence matrices and embedding features at ℓ-th layer. Note that we initialize the node features at 0-th layer as $\mathbf{E}_*^{(0)} = \mathbf{E}_*$. After propagating through L layers, we combine the embeddings learned from each layer with average pooling to obtain final embeddings $\overline{\mathbf{E}}_*$.

3.1.3 Intent Feature Generation. With the learned embeddings, for each user consumption behavior $x_i = (u_i, l_i, t_i, c_i)$, we can generate the disentangled intent features $\mathbf{z}_i^L, \mathbf{z}_i^T, \mathbf{z}_i^C$ in the location-aware, time-aware and category-aware spaces as follows,

$$\begin{aligned} \mathbf{z}_i^L &= \text{MLP}^L([\overline{\mathbf{e}}_{u_i}^L, \overline{\mathbf{e}}_{l_i}]), \quad \mathbf{z}_i^T = \text{MLP}^T([\overline{\mathbf{e}}_{u_i}^T, \overline{\mathbf{e}}_{t_i}]), \\ \mathbf{z}_i^C &= \text{MLP}^C([\overline{\mathbf{e}}_{u_i}^C, \overline{\mathbf{e}}_{c_i}]), \end{aligned} \tag{5}$$

where [] denote *concatenation* operation. Here MLP^s and $\overline{\mathbf{e}}_{u_i}^s$ denote the multilayer perceptron and user embedding of u_i in each subspace $s \in \{L, T, C\}$, respectively, and $\overline{\mathbf{e}}_{l_i}, \overline{\mathbf{e}}_{t_i}, \overline{\mathbf{e}}_{c_i}$ denote the final embedding of l_i, t_i, c_i, respectively. In this way, we obtain the disentangled intent encoders $\Phi^s : x_i \mapsto (x_i) \in \mathbb{R}^d$ to generate intent features $\mathbf{z}_i^s$ in each disentangled subspace s.

3.1.4 Independence-constraint Loss. Since the disentangled intent features should reflect different aspects of user preferences, we add the independence constrain on them. Specifically, following the recent advances of disentangled representation learning [34], we regard the distance correlation of any two intent features among three subspaces $\mathcal{S} = \{L, T, C\}$ as an independence loss to ensure independence, formulated as follows,

$$\mathcal{L}_{\text{IND}} = \frac{1}{M+N} \sum_{i=1}^{M+N} \sum_{s,s' \in \mathcal{S}, s \neq s'} \frac{\text{dCov}(\mathbf{z}_i^s, \mathbf{z}_i^{s'})}{\sqrt{\text{dVar}(\mathbf{z}_i^s) \cdot \text{dVar}(\mathbf{z}_i^{s'})}}, \tag{6}$$

where $\text{dCov}(\cdot)$ and $\text{dVar}(\cdot)$ denote the distance covariance and the distance variance, respectively.

3.1.5 Encoder Pre-training. To enhance the supervision signal in the feature learning, we conduct a pre-training based on the fact that the observed behavior $x_i = (u_i, l_i, t_i, c_i)$ reflect the similarity of embeddings of u_i, l_i, t_i, and c_i. Furthermore, we can use both the labeled and unlabeled data for pre-training, and thus the representation learning can aid the knowledge transfer process from known intents to unknown intents. Specifically, for each consumption behavior $x_i = (u_i, l_i, t_i, c_i)$, we first calculate the prediction score with its disentangled intent features $\mathbf{z}_i^s, s \in \mathcal{S} = \{L, T, C\}$, denoted as,

$$\hat{y}(x_i) = \sum_{s \in \mathcal{S}} f^s(\mathbf{z}_i^s), \tag{7}$$

where $f^s : \mathbb{R}^d \mapsto \mathbb{R}$ denotes the score function in the disentangled subspace s. Then, we adopts BPR loss [29] to ensure that the the observed behaviors can be assigned a higher score than the unobserved ones when pre-train the encoders, formulated as follows,

$$\mathcal{L}_{\text{BPR}} = \frac{1}{|O|} \sum\nolimits_{(x_i, x_j^*) \in O} -\ln \sigma(\hat{y}_{(x_i)} - \hat{y}_{(x_j^*)}), \tag{8}$$

where $O = \{(x_i, x_j^*) | x_i \in \mathcal{D}, x_j^* \notin \mathcal{D}\}$ denotes the pairwise training set built with negative sampling, $\sigma(\cdot)$ is the sigmoid function. Combining the BPR loss and independence loss, the loss function in the pre-training stage can be formulated as follows,

$$\mathcal{L}_{\text{PRE}} = \mathcal{L}_{\text{BPR}} + \lambda \mathcal{L}_{\text{IND}}, \tag{9}$$

where λ denotes the hyperparameter to control the influence of independence constraints among the disentangled intent features.

To sum up, we obtain the disentangled intent feature in each aspect with the disentangled intent encoders to capture the user preferences in distinct aspects.

3.2 Intent Discovery Decoders

To discover the new intents with only a small amount of labeled data, we propose to transfer knowledge from known intents to unknown ones with intent discovery decoders, which consist of three stages, *i.e.*, warm-up stage, main stage, and output stage.

3.2.1 Warm-up Stage. The warm-up stage includes a) intents number estimation and b) feature fine-tuning on the labeled data.

a) Intents Number Estimation. In real-world business scenarios, we may not know the number of unknown intents in the unlabeled user consumption data. Hence, we first propose a simple yet effective method to estimate the number of intents. Given that the designed intent encoders in section 3.1 have sufficiently capture users' core preferences in the disentangled aspects, we first generate intent features with the pre-trained intent encoders, denoted

as $\mathbf{z}_i = \sum_{s\in\mathcal{S}} \mathbf{z}_i^s$, where $\mathbf{z}_i^s$ is the intent feature of record x_i in the subspace s. Then, following [39], we conduct k-means [24] on the extracted intent features to estimate the intent number. Specifically, we assign K' (*e.g.*, three times of the known intent classes K^k) as the number of all intents (known and unknown) and cluster all samples into K' clusters with k-means. Similar to [39], we drop the low confidence clusters (the size smaller than the expected cluster mean size $\frac{M+N}{K'}$) and obtain the total intent number K as follows,

$$K = \sum_{i=1}^{K'} \delta(|S_i| \geq \frac{M+N}{K'}), \tag{10}$$

where $\delta(\cdot)$ is the indicator function (1 if condition satisfied else 0). M and N denote the number of the labeled and unlabeled data, respectively. In this way, we estimate the total intent number K and the number of unknown intents $K^u = K - K^k$.

After estimating the intent number in user consumption behavior data, we will introduce how to discover the new consumption intents. Since we do not reveal intent labels in the pre-training stage (section 3.1.5), we first fine-tune the pre-trained encoders with the labeled (known intents) data.

b) Feature fine-tuning. With the pre-trained intent encoder Φ^s in the disentangled subspace s, following [12], we further extent it with a classification head $\eta_k^s : \mathbb{R}^d \mapsto \mathbb{R}^{K^k}$ (a linear layer with softmax function) to learn a classifier for the K^k known intents in the subspace s. Specifically, we first generate the disentangled intent features $z_i^s = \Phi^s(x_i), s \in \mathcal{S} = \{L, T, C\}$ for each consumption behavior x_i to capture the intent features that reveal distinct aspects of user preferences. Then, we calculate the classification probabilities in each disentangled subspaces as $\eta_k^s(z_i^s)$. Given that user consumption intents may be more relevant to one or more aspects among location-, time- and category-aware preferences, we attentively fuse the classification probabilities in distinct disentangled subspaces with typical attention modules [6, 42], and further optimize the model with the cross-entropy (CE) loss as follows,

$$\begin{aligned} \mathcal{L}_{\text{FT}} &= -\frac{1}{M}\sum_{i=1}^{M} y_i^l \log[\sum_{s\in\mathcal{S}} \alpha^s \cdot \eta_k^s(\mathbf{z}_i^s)], \\ \alpha^s &= \text{softmax}(\mathbf{q}^\top \mathbf{z}_i^s), \end{aligned} \tag{11}$$

where M and y_i^l denotes the number and the intent label of the labeled data $\mathcal{D}^l$, respectively. $\mathbf{q} \in \mathbb{R}^d$ is the learnable attention vector. Note that we froze the weights of encoders Φ^s and only update the parameters of classifiers (*i.e.* η_k^s and $\mathbf{q}$) to avoid overfitting when fine-tuning on the labeled data $\mathcal{D}^l$.

After the feature fine-tuning, we next introduce how to transfer knowledge from known intents to unknown ones.

3.2.2 Main Stage: Knowledge Transferring from Labeled Intents. Once the intent encoder Φ^s and the classifier for known intents η_k^s in each subspace s has been well trained, we next introduce how to discover the new intents by transferring knowledge from known intents to unknown ones. Given that we have estimated the class number of unknown intents K^u in section 3.2.1 a), similar to known intents, we also extend the encoder Φ^s with classification head $\eta_u^s : \mathbb{R}^d \mapsto \mathbb{R}^{K^u}$ for K^u unknown intents. With the classifiers η_k^s (for known intents) and η_u^s (for unknown intents), we can obtain the final classifier $\eta^s = [\eta_k^s, \eta_u^s] : \mathbb{R}^d \mapsto \mathbb{R}^K$ to classify any unlabeled sample into K intents (K_k known and K_u unknown). Hence, the problem turns to how to train the final classifiers η^s with both the labeled data $\mathcal{D}^l$ and unlabeled data $\mathcal{D}^u$.

To address the above problem, we propose the **intent-pair pseudo-label learning for intent discovery**, which consists of two steps, a) label construction based on denoised similarity and b) pseudo-label enhanced joint learning.

a) Label construction based on denoised similarity. The key assumption of intent discovery is that the similar user consumption behaviors on the Meituan platform should belong to the same intent classes. Hence, similar to [12], we first define a relation among pairs of unlabeled samples (x_i^u, x_j^u). Since the well-trained intent encoders can obtain the transferable intent features for both known intents and unknown ones, we can generate the intent features $\mathbf{z}_{i,u}^s, \mathbf{z}_{j,u}^s$ for the above pairwise data in subspace s. Given that consumption intents may be more relevant in one aspect but less in other ones (*e.g. have lunch* and *have dinner* are more relevant in category-aspect but less in time-aspect), for the pair (x_i^u, x_j^u), we generate the pseudo-label r_{ij}^s in each disentangled subspace s based on the similarity of the corresponding intent features $\mathbf{z}_{i,u}^s, \mathbf{z}_{j,u}^s$.

Calculating denoised similarity: Since the robust pseudo-labels are more suitable for training the classifiers of unknown intents, instead of directly calculating the similarity of $\mathbf{z}_{i,u}^s$ and $\mathbf{z}_{j,u}^s$, we propose a more robust method by denoising the intent features with *Low-Pass Fast Fourier Transform* (LPFFT). Since *Fast Fourier Transform* (FFT) can convert signals into the frequency domain, it is widely used for denoising in signal processing area [2]. In this paper, we first conduct FFT for each dimension of intent features, and then we remove the higher-frequency half of signals via *Low-Pass Filter* (LPF). Finally, we perform inverse FFT (IFFT) to generate the robust features. The above operations can be denoted as LPFFT : IFFT(LPF(FFT($\cdot$))). With the denoised features, we further calculate the cosine similarity r_{ij}^s for the pairwise samples (x_i^u, x_j^u) in subspace s, formulated as,

$$r_{ij}^s = \text{COSINE}(\text{LPFFT}(\mathbf{z}_{i,u}^s), \text{LPFFT}(\mathbf{z}_{j,u}^s)),\ s \in \mathcal{S} = \{L, T, C\}, \tag{12}$$

where COSINE($\cdot$) denotes the cosine similarity function. In this way, we obtain the pairwise pseudo-labels in each subspace s.

b) Pseudo-label enhanced joint learning. Given that we have learned η_k^s in section 3.2.1 b), we only randomly initialize the parameters for the new classes in η^s and further train η^s with the pseudo-label enhanced joint learning. For the labeled data $\mathcal{D}^l$, the extended classifier η^s should also classify the known intents correctly. Hence, we extend the cross-entropy (CE) loss in eq. (11) to η^s in three subspaces $\mathcal{S} = \{L, T, C\}$ as follows,

$$\begin{aligned} \mathcal{L}_{\text{CE}} &= -\frac{1}{M}\sum_{i=1}^{M} y_i^{l*} \log[\sum_{s\in\mathcal{S}} \alpha^s \cdot \eta^s(\mathbf{z}_i^s)], \\ \alpha^s &= \text{softmax}(\mathbf{q}^\top \mathbf{z}_i^s), \end{aligned} \tag{13}$$

where y_i^{l*} is the extended one-hot intent label (the dimensions of new classes are set to 0) for the labeled data.

For the unlabeled data, we use the obtained pairwise pseudo-labels in each subspace s to train the corresponding classifier η^s. Specifically, in each disentangled subspace s, we first calculate the

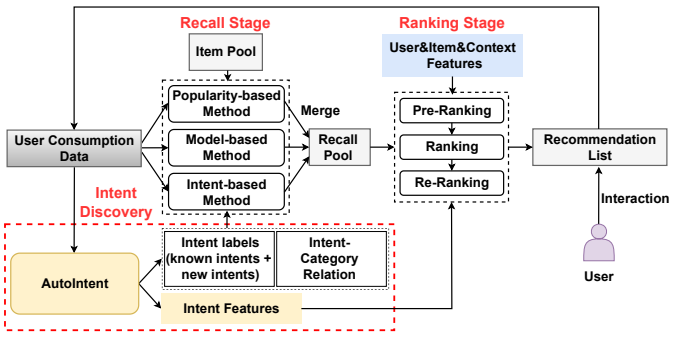


Figure 4: Industrial deployment of AutoIntent in the Meituan recommendation engine.

score for whether the given pairwise samples (x_i^u, x_j^u) belong to the same class or not, denoted as $\hat{s}_{ij}^s = \eta^s(\mathbf{z}_{i,u}^s)^\top \eta^s(\mathbf{z}_{j,u}^s)$. Then, we optimize η^s with the binary cross-entropy (BCE) loss, denoted as,

$$\mathcal{L}_{\text{BCE}} = -\frac{1}{N^2}\sum_{i=1}^{N}\sum_{j=1}^{N}\sum_{s\in\mathcal{S}}[r_{ij}^s \log \hat{s}_{ij}^s + (1-r_{ij}^s)\log(1-\hat{s}_{ij}^s)], \quad (14)$$

where N is the number of unlabeled data $\mathcal{D}^u$. $\mathcal{S} = \{L, T, C\}$ is the set of three disentangled subspaces and r_{ij}^s denotes the pseudo-label of pairwise samples (x_i^u, x_j^u) in subspace s.

To ensure the independence among disentangled intent features, we further introduce the independence loss in eq. (6) and define the overall loss function as follows,

$$\mathcal{L} = \mathcal{L}_{\text{CE}} + \mathcal{L}_{\text{BCE}} + \lambda \mathcal{L}_{\text{IND}}, \quad (15)$$

where λ denotes the hyperparameter to control the influence of independence constraints. Note that we also froze the encoders $\Phi^s(s \in \mathcal{S})$ to avoid over-fitting during the joint-learning process.

In this way, we successfully transfer the knowledge from known intents to unknown intents with the joint-learning framework. The joint-learning framework also creates a feedback loop that refines the intent features with the well-trained classifier η^s, which in turn generates better pairwise pseudo-labels for the training of η^s.

3.2.3 Output Stage. With the intent discovery decoders in three disentangled aspects $\mathcal{S} = \{L, T, C\}$, we can adapt the importance of each aspect to the final intent discovery results. Specifically, we first conduct aspect-level prediction and then leverage attentive fusion to fuse the results from different aspects.

a) Aspect-level prediction. With the well-trained encoder Φ^s and classifier η^s (for both known intents and unknown ones) in each disentangled aspect s, we calculate the prediction score p_i^s for any unlabeled data x_i^u, denoted as $p_i^s = \eta^s(\Phi^s(x_i^u))$.

b) Attentive Fusion. We assume that the well-trained intent feature in each disentangled aspect should reveal the importance of the corresponding result. In other words, our model should pay more attention to the aspect that generates more important intent feature. Hence, we first calculate the attention score α^s with intent features and then combine the predicted results from three aspects to obtain the final predicted intent class, formulated as follows,

$$\begin{aligned} \hat{y}_i^u &= \text{argmax}(\sum_{s\in\mathcal{S}} \alpha^s \cdot p_i^s), \\ \alpha^s &= \text{softmax}(\mathbf{q}^\top \Phi^s(x_i^u)), \end{aligned} \quad (16)$$

Table 1: Statistics of Two Datasets from Meituan.

Dataset	#Users	#Locations	#Time	#Category	#Intents	#Records
Beijing	38,702	13	96	748	19	7,075,926
Shanghai	44,186	13	96	792	19	8,634,379

where $\mathbf{q}$ denotes attention vector. In this way, we can assign any unlabeled sample to a certain known or unknown intent class.

To summarize, with the designed decoders, our AutoIntent can estimate the number of new intents and can further assign the unlabeled data to a certain intent class with the well-trained intent classifier. In other words, AutoIntent can assign the data into distinct intent clusters. Hence, we can obtain the intent-category relation from the intent clusters and further define the semantic information of the new intents by combining the popular categories in the corresponding clusters, which will further contribute to the downstream applications (*i.e.* recommendation) in Meituan.

3.3 Industrial Deployment of AutoIntent

In this section, we will introduce how to deploy our AutoIntent model in the recommendation engine for Meituan APP's homepage. As Figure 4 shows, the recommendation engine consists of two stages, *i.e.*, recall stage and ranking stage. For a certain user on Meituan platform, according to his/her historical consumption data, the recommendation engine first produces an candidate item set (recall pool) with distinct recall methods in the recall stage. Then, the above candidates (recall pool) are passed through multi-stages of ranking (*i.e.*, pre-ranking, ranking, and re-ranking) to generate the final recommendation list. We deploy AutoIntent in the recommendation engine by introducing an additional intent discovery stage, which is benefitial to both the recall and ranking stage.

With the user consumption data (only few data with intent label and most data without label), our AutoIntent can assign any unlabeled sample to a certain known or unknown intent class. Moreover, for each discovered new intent, AutoIntent can generate the corresponding intent-category relation according to the item category information in the user consumption data samples that belong to the current new intent. As illustrated in Figure 4, the generated intent labels and intent-category relation for new intents can enhance the intent-based recall method in the recall stage. For the ranking stage, the intent features captured by AutoIntent can contribute to the better personalized modeling for user consumption behaviors. At a high level, the system in Figure 4 is in a positive feedback loop. The intent discovery stage can constantly discover new intents from the user consumption data, which contributes to better recommendation and user growth. In return, better recommendation and user growth can provide more user consumption data to enhance intent discovery. We conduct evaluation of downstream recommendation in section 5 to verify the effectiveness of the deployment.

4 EVALUATION OF INTENT DISCOVERY

4.1 Experimental Settings

4.1.1 Datasets. We collect two large-scale user consumption data from **Meituan APP** in the two cities of China (Beijing and Shanghai), from Jan. 1st to Mar. 1st, 2021 (60 days), which contain 13 locations and 96 time-slots. To evaluate the model performance, we split the first 48 days' data as the training set, the following 6

Table 2: Performance comparisons on two datasets.

Method	Beijing			Shanghai		
	ACC	ARI	NMI	ACC	ARI	NMI
DeepFM-KM	54.41	35.83	29.17	51.98	33.69	37.63
LightGCN-KM	58.84	36.16	31.48	55.46	35.87	40.82
HAN-KM	60.32	38.59	33.74	58.76	36.65	42.29
HAN-CDAC+	67.34	42.53	36.82	66.18	40.47	45.86
HAN-DeepAligned	69.89	46.48	36.75	67.56	43.28	46.50
HAN-DTC	68.35	47.72	38.36	67.13	43.34	46.91
HAN-RankStat	70.24	49.45	40.29	68.58	44.92	47.46
AutoIntent	**81.07**	**57.34**	**46.81**	**77.39**	**50.27**	**53.35**
Improv.	15.42%	15.96%	16.18%	12.85%	11.91%	12.41%

days' data as validation set, and the last 6 days' data as testing set. Moreover, we select the first 10 intents as known intents and treat the remaining 9 intents as unknown ones. The details of datasets are provided in Table 1 and Appendix A.1.

4.1.2 Metrics. Following [39], we adopt three widely used clustering metrics, **ACC**[3] (Accuracy), **ARI** (Adjusted Rand Index), and **NMI** (Normalized Mutual Information), for evaluation.

4.1.3 Baselines. The two key designs of **AutoIntent** are 1) disentangled intent encoders and 2) intent discovery decoders, we compare AutoIntent with two categories of baselines. 1) Three SOTA feature generating methods[4] (DeepFM [10], LightGCN [14], and HAN [33]) to verify the effectiveness of our encoders. 2) Four SOTA deep clustering methods for intent discovery (CDAC+ [23] and DeepAligned [39]) and new category discovery (DTC [13] and RankStat [12])[5]. We provide the details of baselines in Appendix A.2.

4.2 Overall performance

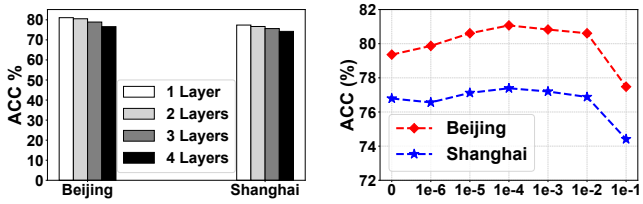
We compare our AutoIntent with SOTA baselines on two datasets. From the results in Table 2, we have the following observations.

- **Our proposed AutoIntent achieves the best performance.** Owing to the disentangled intent encoders and conducting intent discovery from the disentangled view, AutoIntent can capture the preferences of distinct aspects (*i.e.*, location-, time- and category-aspect) and achieves the best performance. On average, AutoIntent outperforms the best baseline by 14.14% on ACC, 13.93% on ARI, and 14.30% on NMI, respectively. The significant performance gains verify the effectiveness of our AutoIntent.
- **Modeling the preferences with the relations from distinct aspects is essential.** For the encoder baselines, HAN-KM and LightGCN-KM achieve better performance than DeepFM-KM, which demonstrates modeling the relations in user consumption behavior is necessary. Moreover, HAN-KM (separately modeling the location-, time- and category-aspect performance with distinct meta-paths) achieves the best performance, which verifies the necessity of modeling the user preferences from the disentangled views with dual hypergraphs in our AutoIntent.
- **For intent discovery, pairwise labeling is easier to optimize than the clustering methods** HAN-RankStat, using the pairwise pseudo-labels to train the classifiers for unlabeled data, achieves a better performance than clustering-based methods (HAN-CDAC+, HAN-DeepAligned, and HAN-DTC), which verifies that learning the classifiers for the unlabeled data is a better choice for intent discovery. Hence, it is necessary to sufficiently model the pairwise similarity from disentangled perspectives for better pseudo-labels in our proposed AutoIntent.

[3]When calculating **ACC**, we first match the predicted intent label and the ground-truth label with the Hungarian algorithm[20].
[4]We use the K-means (KM) [24] to cluster new intents for all encoder baselines.
[5]To ensure better performance, we adopt the best encoder in the SOTA feature generating methods (HAN[33]) as the feature encoder of the above deep clustering methods.

Table 3: Ablation Study of the key designs in AutoIntent.

Dataset	Beijing			Shanghai		
Model Variants	ACC	ARI	NMI	ACC	ARI	NMI
w/o DisenEncoder (HAN)	74.23	53.42	43.45	70.67	47.83	49.69
w/o DS	78.88	50.09	42.08	76.28	45.62	50.67
w/o $\mathcal{L}_{\text{CE}}$	76.28	51.47	42.96	74.19	46.08	50.83
w/o $\mathcal{L}_{\text{BCE}}$	36.26	26.45	20.37	32.41	24.19	27.32
w/o $\mathcal{L}_{\text{IND}}$	79.36	55.88	45.95	76.79	49.47	52.34
AutoIntent	**81.07**	**57.34**	**46.81**	**77.39**	**50.27**	**53.35**



(a) Impact of L (b) Impact of λ

Figure 5: Impact of model depth L (a) and independent coefficient λ (b) on two datasets from Meituan.

4.3 Studies of AutoIntent

4.3.1 Ablation Study. AutoIntent has the following key designs: 1) *Disentangled Intent Encoders* (DisenEncoder), 2) *Denoised Similarity* (DS), and 3) *Loss Functions in Intent Discovery Decoders* (*i.e.*, $\mathcal{L}_{\text{CE}}$, $\mathcal{L}_{\text{BCE}}$, and $\mathcal{L}_{\text{IND}}$). To evaluate the effectiveness of the above components, we compare the performance of the model variants that without (w/o) a certain component. From the results in Table 3, removing any component leads to a significant performance drop, which demonstrates the effectiveness of the above key components. Among them, removing the BCE loss causes a dramatic drop, which verifies that our denoised feature comparison with *Low-Pass Fast Fourier Transform* (LPFFT) can indeed generate reliable pairwise pseudo labels to train the classifiers for unlabeled data. Moreover, if we replace the designed *Disentangled Intent Encoders* with HAN (the SOTA baseline encoder), the performance also suffers a significant drop, which demonstrates the necessity of modeling the user preferences in distinct aspects with dual hypergraphs in *Disentangled Intent Encoders*. We further provide the ablation study of *Denoised Similarity Methods* and *Training Scheme* in Appendix A.5.

4.3.2 Hyper-parameter Study. In this part, we study the impact of two essential hyper-parameters in our proposed AutoIntent, *i.e.*, the model depth L and the independent coefficient λ.

1) Impact of Model Depth L. To study the impact of depth of dual hypergraphs in encoders, we vary L in $\{1, 2, 3, 4\}$. From the results in Figure 5 (a), the model achieves the best performance with one layer on both datasets. The possible reason is that the hypergraph can capture the high-order relations without stacking multiple layers, which means our hypergraph-based encoders are more efficient. Hence, we set L as 1 for both datasets.

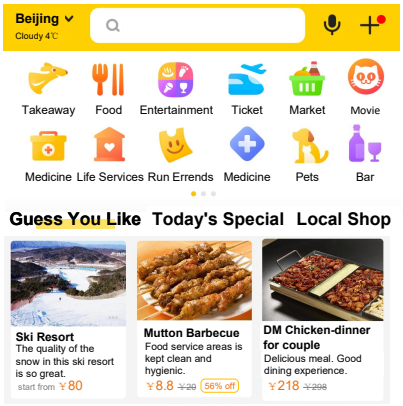


Figure 6: Illustration of Meituan App's homepage.

2) Impact of Independent Coefficient λ. To evaluate the impact of λ, we vary it in $\{1e^{-6}, 1e^{-5}, 1e^{-4}, 1e^{-3}, 1e^{-2}, 1e^{-1}\}$. According to the results in Figure 5 (b), our AutoIntent is not sensitive to λ (the performance remains relatively stable in a certain interval $1e^{-5} \sim 1e^{-3}$) and achieves the best performance when $\lambda = 1e^{-4}$. Hence, we set λ as $1e^{-4}$ for both datasets.

5 EVALUATION OF DOWNSTREAM RECOMMENDATION

In this section, we further evaluate whether our proposed AutoIntent can enhance the downstream recommendation in Meituan. Referring to the proposed deployment scheme in section 3.3, we conduct the downstream evaluations on the recommendation engine for the Meituan APP homepage (including recall and ranking stage) to evaluate the effectiveness of the deployment. Specifically, we obtain the recall pool by merging the results from all the three strategies (popularity-based, model-based, and intent-based). In the ranking stage, we further generate recommendation list from the recall pool. To evaluate the effectiveness intent discovery module in the deployment system, we compare the recommendation performance with (strategy A) and without (strategy B) AutoIntent. In strategy A (with AutoIntent), as shown in Figure 4, the discovered new intents and intent features are used in recall and ranking stage, respectively. The evaluation of recommendation are conducted in the Meituan APP homepage, as shown in Figure 6, involving about 8 million users. We compare the recommendation performance among the users in Beijing and Shanghai with two ranking-based metrics, recall@10 (R@10) and NDCG@10 (N@10). In Meituan APP, there are many different Business Units (BUs), such as `Takeaway` and `Pets` in Figure 6. Given that the user intents in different BUs may be different, we conduct the experiments with two settings, *i.e. intent discovery in known BUs* and *intent discovery in new BUs*.

a) Intent Discovery in Known BUs. We first compare the recommendation performance with (w) and without (w/o) AutoIntent in the known BUs. We use the data from all BUs in Beijing and Shanghai datasets to train our AutoIntent model. AutoIntent discovers 11 new intents with 19 known intents on both offline datasets. Then, we deploy the AutoIntent model on all BUs to evaluate the recommendation performance. From the results in Table 4, we can observe that the recommendation performance with AutoIntent improves by 9.54% on Recall and 11.23% on NDCG, respectively. Such a significant gain verifies that AutoIntent can enhance the recommendation with the discovered new intents.

Table 4: Evaluations of Downstream Recommendation (in %).

	Model	Beijing			Shanghai		
		R@10	N@10	#Intent	R@10	N@10	#Intent
Known BUs	w/o AutoIntent	14.25	11.43	19	12.94	10.08	19
	w AutoIntent	**15.57**	**12.68**	19+(11)	**14.21**	**11.24**	19+(11)
	Imp.	9.26%	10.94%	-	9.81%	11.51%	-
New BUs	w/o AutoIntent	13.27	10.57	15	11.62	9.06	15
	w AutoIntent	**15.08**	**12.32**	15+(7)	**13.45**	**10.74**	15+(7)
	Imp.	13.64%	16.56%	-	15.75%	18.54%	-

b) Intent Discovery in New BUs. Another important task in Meituan is the intent discovery in new BUs, which is a more challenging task with no known intents in new BUs. Here, we regard `Pets` as new BU and remove the known intents in `Pets` BU. Hence, the number of known intents reduce to 15 after removing the intent labels in `Pets` BU. In the offline training of AutoIntent, we only use the 15 known intents for training and discover 7 new intents in `Pets` BU. Then, we deploy the well-trained AutoIntent model in the `Pets` BU and evaluate the recommendation performance. As the results in Table 4, the recommendation model with AutoIntent achieves the performance gains of 14.70% on Recall and 17.55% on NDCG, which is a more significant improvement than in the known BUs. The possible reason is that AutoIntent can capture the common features in distinct intents and transfer the knowledge from the intents (in known BUs) to the new intents (in new BUs).

In short, the results demonstrate that a) AutoIntent can enhance the recommendation performance with the discovered intents and b) our proposed AutoIntent can transfer knowledge among BUs to achieve more significant improvement in New BUs.

6 RELATED WORK

Intent Discovery Intent discovery aims to discover new intents by transferring the knowledge from the known intents to the new ones and has been explored in dialogue systems [26, 27, 30, 31]. CDAC+ [23] propose to discover new intents via deep adaptive clustering with cluster refinement. DeepAligned [39] enhance the deep clustering with an alignment strategy to tackle the label inconsistency problem. Another research problem that is highly related to intent discovery is the new visual categories discovery[12, 13, 16, 40]. DTC [13] extends the Deep Embedded Clustering to a transfer learning setting by introducing the temporal ensemble and consistency. RankStat [12] propose to generate the pairwise pseudo-label with rank statistics to transform the clustering task to a binary classification task. Zhao *et al* [40] further extent RankStat in a two-branch learning framework. Different from the above works, we aim to discover intents from user consumption data, which is more challenging and needs to sufficiently capture the user preferences.

Hypergraph Learning Hypergraph [1] introduces hyperedge, a special edge to connect more than two nodes, to naturally capture high-order relations, which has widely used in recommendation [9, 21, 32, 36, 38]. The learning on hypergraph can be regarded as a two-stage process and has been well-explored. HGNN [8] introduces graph convolution to hypergraph to learn the graph embedding. HyperGAT [7] attentively aggregates the node information

and extends GAT to hypergraph. HGC-RNN [37] combines the hypergraph learning and RNN to learn temporal dependency among different hypergraphs. In this work, we use dual hypergraphs to model the user preferences.

Disentangled Representation Learning Disentangled representations can independently model a certain object from multiple aspects or factors [3]. The earlier works [4, 5, 15] learn the disentangled features via the regularized variational auto-encoders [19]. With the development of GNNs, there exist some works [21, 22, 34, 35, 41] that explore how to learn the disentangled embeddings on the graphs. In this work, we learn the disentangled intent features from distinct aspects and further conduct intent discovery in a disentangled manner.

7 CONCLUSION

In this work, we approach the new problem of user consumption intents discovery that is highly related to the recommendation in Meituan and develop a system named AutoIntent to automatically discover intents from the consumption data. AutoIntent first leverages dual hypergraph neural networks to learn the disentangled intent features with the disentangled intent encoders. Then, the intent discovery decoders transfer the knowledge from the known intents to discover new ones. Finally, we deploy AutoIntent in the Meituan recommendation engine for downstream evaluation. Experiments verify that AutoIntent can effectively discover unknown intents and enhance recommendation.

ACKNOWLEDGMENT

This work is supported in part by National Key Research and Development Program of China under 2020YFA0711403. This work is supported in part by National Natural Science Foundation of China under 61971267, 61972223, and U1936217. This work is also supported by Meituan.

REFERENCES

[1] Sameer Agarwal, Kristin Branson, and Serge Belongie. 2006. Higher order learning with graphs. In *ICML*. 17–24.
[2] John G Anderson and Susan E Hough. 1984. A model for the shape of the Fourier amplitude spectrum of acceleration at high frequencies. *Bulletin of the Seismological Society of America* 74, 5 (1984).
[3] Yoshua Bengio, Aaron Courville, and Pascal Vincent. 2013. Representation learning: A review and new perspectives. *TPAMI* 35, 8 (2013).
[4] Christopher P Burgess, Irina Higgins, Arka Pal, Loic Matthey, Nick Watters, Guillaume Desjardins, and Alexander Lerchner. 2018. Understanding disentangling in *beta*-VAE. *arXiv preprint arXiv:1804.03599* (2018).
[5] Ricky TQ Chen, Xuechen Li, Roger Grosse, and David Duvenaud. 2018. Isolating sources of disentanglement in variational autoencoders. *arXiv preprint arXiv:1802.04942* (2018).
[6] Zhiyong Cheng, Fan Liu, Shenghan Mei, Yangyang Guo, Lei Zhu, and Liqiang Nie. 2022. Feature-Level Attentive ICF for Recommendation. *TOIS* (2022).
[7] Kaize Ding, Jianling Wang, Jundong Li, Dingcheng Li, and Huan Liu. 2020. Be More with Less: Hypergraph Attention Networks for Inductive Text Classification. *arXiv preprint arXiv:2011.00387* (2020).
[8] Yifan Feng, Haoxuan You, Zizhao Zhang, Rongrong Ji, and Yue Gao. 2019. Hypergraph neural networks. In *AAAI*, Vol. 33.
[9] Chen Gao, Yu Zheng, Nian Li, Yinfeng Li, Yingrong Qin, Jinghua Piao, Yuhan Quan, Jianxin Chang, Depeng Jin, Xiangnan He, et al. 2021. Graph Neural Networks for Recommender Systems: Challenges, Methods, and Directions. *arXiv preprint arXiv:2109.12843* (2021).
[10] Huifeng Guo, Ruiming Tang, Yunming Ye, Zhenguo Li, and Xiuqiang He. 2017. DeepFM: a factorization-machine based neural network for CTR prediction. *arXiv preprint arXiv:1703.04247* (2017).
[11] Will Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. *NeurIPS* 30 (2017).
[12] Kai Han, Sylvestre-Alvise Rebuffi, Sebastien Ehrhardt, Andrea Vedaldi, and Andrew Zisserman. 2020. Automatically discovering and learning new visual categories with ranking statistics. *arXiv preprint arXiv:2002.05714* (2020).
[13] Kai Han, Andrea Vedaldi, and Andrew Zisserman. 2019. Learning to discover novel visual categories via deep transfer clustering. In *ICCV*.
[14] Xiangnan He, Kuan Deng, Xiang Wang, Yan Li, Yongdong Zhang, and Meng Wang. 2020. Lightgcn: Simplifying and powering graph convolution network for recommendation. In *SIGIR*.
[15] Irina Higgins, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. 2016. beta-vae: Learning basic visual concepts with a constrained variational framework. (2016).
[16] Yen-Chang Hsu, Zhaoyang Lv, Joel Schlosser, Phillip Odom, and Zsolt Kira. 2019. Multi-class classification without multi-class labels. *arXiv preprint arXiv:1901.00544* (2019).
[17] Manas R Joglekar, Cong Li, and Mei Chen. 2020. Neural input search for large scale recommendation models. In *KDD*.
[18] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).
[19] Diederik P Kingma and Max Welling. 2013. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114* (2013).
[20] Harold W Kuhn. 1955. The Hungarian method for the assignment problem. *Naval research logistics quarterly* 2, 1-2 (1955), 83–97.
[21] Yinfeng Li, Chen Gao, Hengliang Luo, Depeng Jin, and Yong Li. 2022. Enhancing Hypergraph Neural Networks with Intent Disentanglement for Session-based Recommendation. In *SIGIR*.
[22] Yinfeng Li, Chen Gao, Quanming Yao, Tong Li, Depeng Jin, and Yong Li. 2022. DisenHCN: Disentangled Hypergraph Convolutional Networks for Spatiotemporal Activity Prediction. In *2022 IEEE 38th International Conference on Data Engineering (ICDE)*. IEEE.
[23] Ting-En Lin, Hua Xu, and Hanlei Zhang. 2020. Discovering new intents via constrained deep adaptive clustering with cluster refinement. In *AAAI*, Vol. 34.
[24] James MacQueen et al. 1967. Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, Vol. 1. Oakland, CA, USA.
[25] Julian McAuley and Alex Yang. 2016. Addressing complex and subjective product-related queries with customer reviews. In *WWW*. 625–635.
[26] Qingkai Min, Libo Qin, Zhiyang Teng, Xiao Liu, and Yue Zhang. 2020. Dialogue state induction using neural latent variable models. *arXiv preprint arXiv:2008.05666* (2020).
[27] Hugh Perkins and Yi Yang. 2019. Dialog intent induction with deep multi-view clustering. *arXiv preprint arXiv:1908.11487* (2019).
[28] Yukun Ping, Chen Gao, Taichi Liu, Xiaoyi Du, Hengliang Luo, Depeng Jin, and Yong Li. 2021. User Consumption Intention Prediction in Meituan. In *KDD*.
[29] Steffen Rendle, Christoph Freudenthaler, and et al. 2012. BPR: Bayesian personalized ranking from implicit feedback. *arXiv preprint arXiv:1205.2618* (2012).
[30] Chen Shi, Qi Chen, Lei Sha, Sujian Li, Xu Sun, Houfeng Wang, and Lintao Zhang. 2018. Auto-dialabel: Labeling dialogue data with unsupervised learning. In *Proceedings of the 2018 conference on empirical methods in natural language processing*. 684–689.
[31] Nikhita Vedula, Rahul Gupta, Aman Alok, and Mukund Sridhar. 2020. Automatic discovery of novel intents & domains from text utterances. *arXiv preprint arXiv:2006.01208* (2020).
[32] Jianling Wang, Kaize Ding, Liangjie Hong, Huan Liu, and James Caverlee. 2020. Next-item recommendation with sequential hypergraphs. In *SIGIR*.
[33] Xiao Wang, Houye Ji, Chuan Shi, Bai Wang, Yanfang Ye, Peng Cui, and Philip S Yu. 2019. Heterogeneous graph attention network. In *WWW*.
[34] Xiang Wang, Hongye Jin, An Zhang, Xiangnan He, Tong Xu, and Tat-Seng Chua. 2020. Disentangled graph collaborative filtering. In *SIGIR*.
[35] Yifan Wang, Suyao Tang, Yuntong Lei, Weiping Song, Sheng Wang, and Ming Zhang. 2020. DisenHAN: Disentangled Heterogeneous Graph Attention Network for Recommendation. In *CIKM*.
[36] Hansheng Xue, Luwei Yang, Vaibhav Rajan, Wen Jiang, Yi Wei, and Yu Lin. 2021. Multiplex Bipartite Network Embedding using Dual Hypergraph Convolutional Networks. In *WWW*.
[37] Jaehyuk Yi and Jinkyoo Park. 2020. Hypergraph Convolutional Recurrent Neural Network. In *KDD*. 3366–3376.
[38] Wenhui Yu and Zheng Qin. 2020. Graph Convolutional Network for Recommendation with Low-pass Collaborative Filters. In *ICML*.
[39] Hanlei Zhang, Hua Xu, Ting-En Lin, and Rui Lyu. 2021. Discovering new intents with deep aligned clustering. In *AAAI*, Vol. 35.
[40] Bingchen Zhao and Kai Han. 2021. Novel visual category discovery with dual ranking statistics and mutual knowledge distillation. *NeurIPS* 34 (2021).
[41] Yu Zheng, Chen Gao, Xiang Li, Xiangnan He, Yong Li, and Depeng Jin. 2021. Disentangling User Interest and Conformity for Recommendation with Causal Embedding. In *WWW*.
[42] Chang Zhou, Jinze Bai, Junshuai Song, Xiaofei Liu, Zhengchao Zhao, Xiusi Chen, and Jun Gao. 2018. Atrank: An attention-based user behavior modeling framework for recommendation. In *AAAI*.

A APPENDIX FOR REPRODUCIBILITY

A.1 Datasets and Evaluation Setting

We collect two large-scale user consumption data from **Meituan APP** in the two cities of China (Beijing and Shanghai), from Jan. 1st to Mar. 1st, 2021 (60 days). According to the business logic of Meituan, there are 13 locations and 96 time-slots (dividing a day into 48 time-slots for weekends and weekdays) in the collected datasets. We only have 19 known intents among a small amount of data (∼ 17.4%), and most of the user consumption behaviors lack intent labels. To evaluate the model performance in the offline intent discovery task, we split the first 48 days' data as the training set, the following 6 days' data as validation set, and the last 6 days' data as testing set. Specifically, we select the first 10 intents as known intents and treat the remaining 9 intents as unknown ones (10 known + 9 unknown intents). We regard the data with known intents and part of the unknown intents data as training set[6]. For the evaluation, we test all the methods on the remaining unknown intents data to evaluate the performance.

A.2 Baseline Models

In this section, we give the detailed descriptions of all compared methods as follows.

(1) Feature generating methods:

- **DeepFM** [10] combines the FM and deep neural networks to capture the feature interactions. We concatenate the user ID, location ID, time-slot ID, and category ID as input.
- **LightGCN** [14] is the state-of-the-art GCN model for recommendation. We adopt it to our task by construct a graph with 6 types of edges among the four types of nodes (*i.e.*, user, location, time and category). We add the learned node embeddings for each record as the intent feature.
- **HAN** [33] is a general state-of-the-art heterogeneous graph learning method. We adapt it to the graph with four types of nodes (user, location, time, and category) by designing three types of meta-path (*user-location, user-time, user-category*) to represent user's preferences on distinct aspects.

We combine those feature generating baselines with K-means [24] to obtain **DeepFM-KM**, **LightGCN-KM**, and **HAN-KM** for the consumption intent discovery task in Meituan.

(2) Deep clustering methods:

- **CDAC+** [23] refines the cluster results by forcing the model to learn from the high confidence assignments for intent discovery.
- **DeepAligned** [39] is the SOTA method for intent discovery in dialogue systems, which proposes an alignment strategy to tackle the label inconsistency problem during clustering assignments.
- **DTC** [13] extends the Deep Embedded Clustering to a transfer learning setting by introducing the temporal ensemble and consistency for the new category discovery.
- **RankStat** [12] is the SOTA method for the new category discovery, which uses pairwise labeling and rank statistics to transfer knowledge of the labeled classes to the unlabeled data.

[6]We conduct sampling strategy when obtaining training set to ensure the proportion of known intents data (∼ 17.4%) is consistent with the distribution of the original data.

We replace the feature encoder of the deep clustering methods with HAN [33] (the encoder with best performance) to obtain **HAN-CDAC+**, **HAN-DeepAligned**, **HAN-DTC**, and **HAN-RankStat** for the consumption intent discovery task in Meituan.

A.3 Metrics

A.3.1 Metrics for the intent discovery task. Following [39], we adopt three widely used clustering metrics, **ACC** (Accuracy), **ARI** (Adjusted Rand Index), and **NMI** (Normalized Mutual Information), to evaluate the performance of the intent discovery task. Note that we first match the predicted intent label and the ground-truth label with the Hungarian algorithm[20] when calculating **ACC**.

A.3.2 Metrics for the recommendation task. Following [14, 17], we use two widely used ranking-based metrics, **Recall@K** and **NDCG@K** (we set K as 10 by following [14, 22]), to evaluate the performance of the recommendation task.

A.4 Implementation Details

A.4.1 Efficient Implementation for Large-scale Industrial Datasets. Given that the user number is huge in large scale datasets from Meituan, for all of the graph-based encoders (*i.e.*, LightGCN, HAN, and the dual-hypergraphs in our AutoIntent), the propagation on the graph is very time-consuming. Inspired by GraphSAGE [11], instead of training node embedding with its all neighbors, we propose to sample local neighborhoods for each node and update its embedding by aggregating features from its local neighborhoods. Specifically, we conduct the sample strategy in a mini-batch manner. Given B nodes in a mini-batch and the sample number is N, the cost of message passing is $O(BN)$, which is more efficient and only related to the mini-batch size and sample number. Hence, with the above sample strategy, the graph-based methods can be used in large-scale industrial datasets. Following GraphSAGE [11], we set the sample number N as 25 in our AutoIntent model for the two datasets from Meituan.

A.4.2 Hyper-parameter Settings. For all the models, the embedding size and hidden state size are set as 64, the batch size is set to 2048. We optimize all the methods with Adam [18] optimizer, which initializes learning rate as 0.001 and will decay it by 0.1 after every three epochs. We also utilize early stopping to detect over-fitting, and the training process will be stopped if ACC on the validation set does not increase for five epochs. For the baseline methods, we initialize the hyper-parameters as the original papers and carefully tune them to get optimal performance. For AutoIntent, we set the initial number of all intents K' as $3K^k$ when estimating intent number. We further study the impact of other essential hyper-parameters (the layer number of dual hypergraphs in intent encoders L and the independent coefficient λ) in section 4.3.2. *For all methods, we run ten times with the same partition and report the average results.*

A.5 Additional Ablation Study

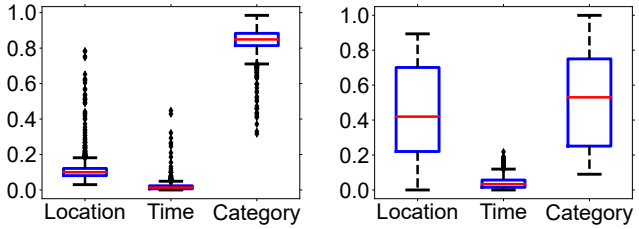
In this section, we further provide the additional ablation study of *Denoised Similarity Methods* and *Training Scheme*.

Table 5: Ablation Study of the Denoised Similarity Methods.

Dataset	Beijing			Shanghai		
Model Variants	**ACC**	**ARI**	**NMI**	**ACC**	**ARI**	**NMI**
RS	78.35	49.71	41.87	76.20	45.54	50.56
Cosine	78.88	50.09	42.08	76.28	45.62	50.67
HPFFT	72.63	31.77	31.10	76.25	45.34	50.61
BPFFFT	75.99	41.96	36.53	76.79	49.75	50.73
LPFFT	**81.07**	**57.34**	**46.81**	**77.39**	**50.27**	**53.35**

Table 6: Ablation Study of the training scheme.

Dataset	Beijing			Shanghai		
Model Variants	**ACC**	**ARI**	**NMI**	**ACC**	**ARI**	**NMI**
w/o PRE	77.96	53.43	44.25	74.71	47.49	51.18
w/o FT	80.15	56.98	45.52	76.81	49.75	52.84
w/o JL	80.42	57.16	46.58	76.94	49.91	52.95
AutoIntent	**81.07**	**57.34**	**46.81**	**77.39**	**50.27**	**53.35**



(a) Snack (b) Doing Exercise

Figure 7: The attention distributions of each disentangled aspects for two intents (Snack and Doing Exercise).

A.5.1 Ablation Study of Denoised Similarity Methods. We combine the *Low-Pass Fast Fourier Transform* (LPFFT) and Cosine similarity to calculate the pairwise pseudo labels in eq. (12). In this section, we further compare the LPFFT with other alternative denoised similarity methods, *i.e.*, ranking statistics proposed in [12] (RS), cosine similarity without denoising (Cosine), High-Pass Fast Fourier Transform (HPFFT), Band-Pass Fast Fourier Transform (BPFFT). From the results in Table 5, the denoised methods (RS, LPFFT) achieve the better performance. Among the methods with filter algorithms, HPFFT achieves the worst performance while LPFFT achieves the best, which verifies that the higher-frequency signals are more likely to be noisy and filtering out them contributes to better performance.

A.5.2 Ablation Study of Training Scheme. As for the training scheme of AutoIntent, we first pre-train the disentangled intent encoders with both the labeled and unlabeled data (don't use intent label). Then, we fine-tune the classifier for known intents on the labeled data. Finally, we transfer the knowledge form the known intents to unknown ones with the peseudo-label enhanced joint learning. To verify the effectiveness of the each step in training scheme, we compare the performance of model variants that without pre-training (w/o PRE), without fine-tuning (w/o FT), without peseudo-label enhanced joint learning (w/o JL) and AutoIntent. From the results in Table 6, we can observe that removing any training step will cause significant performance drop. Among them, removing the pre-training of disentangled intent encoders causes a dramatic drop , which verifies that the pre-training without intent label indeed captures the user preference without bias and can generate the transferable intent features.

A.6 Case Study

We assume that user consumption behaviors are related to multiple aspects (*i.e.*, location, time, and category). Hence, we design disentangled intent encoders to obtain the intent feature in each aspect and further conduct intent discovery in a disentangled manner. To verify the effectiveness of the disentangled setting in the user consumption intent discovery task, we select two kinds of discovered new intents (*i.e.*, `snack` and `doing exercise`) for case study. Specifically, we conduct the attention distribution analysis with a box plot for the attention weights calculated from the disentangled intent features in Eq. (16). From the results in Figure 7, we can observe that the attention distributes in different intents are quite different. For the intent `snack`, the attention values in the category aspect are larger than other aspects, which means that the consumption intent `snack` is more related to users' intrinsic preference, such as taste and brand. For the intent `doing exercise`, the attention values in location and category aspects are larger than the time aspect, which means the user who wants to do exercise can be more likely to be influenced by the location factor (where he/she is) and intrinsic preference (which sport he/she likes). In short, the results of the case study further verify the effectiveness of the disentangled setting in our AutoIntent model.