Topological Representation Learning for E-commerce Shopping Behaviors

Yankai Chen[†], Quoc-Tuan Truong[⋄], Xin Shen[⋄], Ming Wang[⋄], Jin Li[⋄], Jim Chan[⋄], Irwin King[†]

[†]The Chinese University of Hong Kong {ykchen, king}@cse.cuhk.edu.hk

[⋄]Amazon.com, Inc. {truquoc, xinshen, mingww, jincli, jamchan}@amazon.com

ABSTRACT

Learning compact representation from customer shopping behaviors is at the core of web-scale E-commerce recommender systems. At Amazon, we put great efforts into learning embedding of customer engagements in order to fuel multiple downstream tasks for better recommendation services. In this work, we define the notion of *shopping trajectory* that consists of customer interactions at the categorical level of products, then construct an end-to-end model namely C-STAR which is capable of learning rich embedding for representing the variable-length customer trajectory. C-STAR explicitly captures the *trajectory distribution similarity* and *trajectory topological semantics*, providing a coarse-to-fine trajectory representation learning paradigm both structurally and semantically. We evaluate the model on Amazon proprietary data as well as four public datasets, where the learned embeddings have shown to be effective for customer-centric tasks including *customer segmentation* and *shopping trajectory completion*.

CCS CONCEPTS

• Information systems → Recommender systems; **• Computing methodologies → Learning latent representations**.

KEYWORDS

Topological Representation Learning; Shopping Trajectory; Amazon Recommendation

ACM Reference Format:

Yankai Chen[†], Quoc-Tuan Truong[⋄], Xin Shen[⋄], Ming Wang[⋄], Jin Li[⋄], Jim Chan[⋄], Irwin King[†] . 2023. Topological Representation Learning for E-commerce Shopping Behaviors. In *19th-Workshop on Mining and Learning with Graphs at SIGKDD Conference on Knowledge Discovery and Data Mining (MLG-KDD '23), August, 2023, Long Beach CA, USA.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

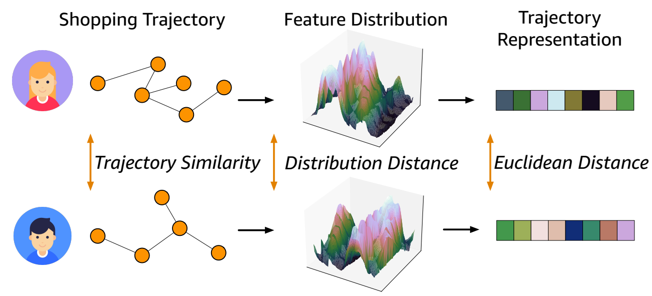

Figure 1: Illustration of trajectory-wise similarity learning.

1 INTRODUCTION

Amazon provides versatile recommendation services at different shopping stages, e.g., from product searching and browsing to shopping-cart checkout. To alleviate information overload, personalized recommendations assist customers in effectively and promptly discovering desirables from the large product corpus. Apart from prediction accuracy, the other prong of facilitating reliable recommendation lies in performing it in an *interpretable* manner to the end customers. This motivates us to understand customers' shopping preferences and intentions through a variety of historical engagements and contexts.

To achieve this goal, we aim to build up an adaptive algorithm that can effectively learn rich representations from engagement data. Such learned representations would benefit multiple downstream customer understanding and serving applications [7, 10, 11, 21, 27, 57]. For instance, on the one hand, segmenting embedded customers who are alike in the shopping behaviors provides a general understanding to recognize their similar interests and affinities, giving rise to inspirational recommendations. On the other hand, explicitly learning and aggregating customer shopping topology information is tractable in the embedding space, which develops a more focused individual analysis and prediction. With these enhanced techniques, we help to delight customers with more personalized shopping experiences.

Challenges. We identify the fundamental requirement for the learned representations is to jointly pose the properties of *accurate similarity measurement* and *informative semantic retention*. The technical challenges are thus twofold:

- Customer engagements vary both quantitatively and substantively. How to learn the fixed-size representations whilst thoroughly reflecting on customers' shopping dynamic and diverse activities is the key question that remains to be investigated. This is particularly important for customer segmentation because the estimated similarity is normally ascertained as their mutual distance in the embedding space, which should be matched with the real-world measurement. One straightforward implementation is to "aggregate" all kinds of latent information, e.g., concatenation or pooling of feature embeddings; this, however, may be naive to hardly provide the theoretical guarantee.

MLG '23, August, 2023, Long Beach CA, USA

ACM ISBN 978-1-4503-XXXX-X/18/06…$15.00

https://doi.org/10.1145/nnnnnnn.nnnnnnn

- Customer historical interactions reflect their preferences and interests. Vectorizing such information into the form of compact embeddings is a prerequisite for explicit analysis and inference of customers' shopping intents. Apart from the customer-wise similarity measurement, this unified representation should also retain semantics coming from the engaged product knowledge as much and effectively as possible. This is vital for various recommendation scenarios such as product complements.

Approach and Contributions. In this work, we investigate the aforementioned problem and introduce a novel Customer Shopping TrAjectory Representation Learning framework (C-STAR). Our proposed C-STAR can effectively encode customer variable-length engagements in the continuous Euclidean space, where they can be efficiently utilized for customer understanding and recommendations. Specifically, we first introduce Amazon PR-Graph [25], i.e., an internal knowledge base of product categories and relations that are organized in the graph format. Customers' isolated product interactions are mapped into PR-Graph forming the notion of *shopping trajectories*. Each trajectory can be essentially viewed as a sub-graph pattern of PR-Graph, which allows us to learn the customer trajectory representation with both structural and semantic information. The proposed C-STAR provides an end-to-end representation learning paradigm that produces a coarse-to-fine trajectory-wise similarity measurement as well as informative semantic enrichment in the embedding space. Concretely, we have made the following technical contributions:

(1) *Trajectory-wise Distribution Similarity.* Each customer is associated with a unique shopping trajectory, we make an assumption that elements constructing the trajectory are samples from an unknown probability distribution, in terms of the customer's underlying preferences. We then propose to capture the trajectory-wise similarity by measuring the distribution distance, and further embed such information into the trajectory representations. Underpinned by the Optimal Transport Theory, our proposed method thus presents a matched distance/similarity measurement between the realistic and embedding space. An illustrative process is demonstrated in Figure 1.
(2) *Trajectory Topological Semantics.* To capture the relational knowledge among the trajectory elements, we further propose to learn semantics from the structure posed by PR-Graph. This will not merely enrich the semantics of trajectory representations but also provide a *fine-grained* proximity measurement, refining its capability for shopping intent identification and complementary recommendation.
(3) *Large-Scale Evaluation.* Besides the methodology contributions, we propose two evaluation tasks and systematically conduct experiments on both large-scale Amazon internal datasets and four public datasets in order to measure the quality of the learned representations by C-STAR.

2 RELATED WORK

Probability Distribution Distance Learning. To quantify the distance between probability distributions, one may utilize *divergences* such as Kullback–Leibler divergence [37], Jensen–Shannon divergence [20], or *metrics* such as *Hellinger distance* [29]. Among these measurement tools, Wasserstein metric [30] with several rigorous mathematical properties has recently attracted the attention in the machine learning community, especially in generative modeling, e.g., generative adversarial networks [1] and variational autoencoders [53]. The major concern of the classic Wasserstein distance is the expensive computation cost for high-dimensional distributions. Different from numerical optimization methods [15, 34, 50], recent studies of *Sliced-Wasserstein distance* [3, 32] presents significantly lower computational requirements. The idea is to obtain adequate linear projections of the original distribution one-dimensional ones, and then average the distances between these projected counterparts, based on the fact that one-dimensional Wasserstein distance has a closed-form solution. Thus, the sliced-Wasserstein distance motivates a variety of practical tasks [5, 12, 33, 35, 42, 43, 47, 72], which also underpins our proposed modeling of the high-dimensional trajectory distribution similarity.

Representation Learning for Graphs. Graph representation learning aims to learn both topologies as well as the features of graph components, e.g., nodes, edges, and sub-graph patterns [51, 52, 64, 65, 71, 73–75]. Then the learned embeddings can be used as feature inputs for downstream machine-learning tasks. While traditional methods usually rely on summarizing graph statistics or manual feature engineering [24], graph convolutional networks (GCNs) are more flexible and adaptive to learning the latent graph information, and thus they have witnessed rapid development in recent years. While early GCN work studies the graph convolutions on the *spectral domain* [4, 16], *spatial-based* GCN models [2, 23] re-define the graph convolution operations by aggregating the neighborhood embeddings to update the target node's embedding. Due to their powerful ability to learn the hidden graph patterns, we thus employ the graph convolutions in our proposed framework for explicit knowledge extraction from graph-structured data.

Product Knowledge Mining at Amazon. In Amazon, there is a stream of research working on knowledge discovery of product semantics and relations [6, 18, 38, 40, 44, 46, 48, 68, 70, 76]. Product knowledge graph is essential for both product understanding as well as customer understanding. It builds the foundation to inspire various applications such as error detection [13], complementary and sequential recommendation [25, 39, 61], explainability [59], and product summarization [54].

In this work, we make use of PR-Graph, a graph of product category, which categorizes numerous products into around 15K nodes and generalizes their categorical mutual relations into 417K edges. Different from previous work that utilizes PR-Graph mainly to learn *node-level* embeddings [25, 61], our work investigates a novel learning setting of *sub-graph patterns* for customer shopping trajectories. We propose to jointly capture trajectory-wise similarity and semantics, enabling the learned C-STAR embeddings to be effectual for multiple downstream customer-centric tasks.

3 PROBLEM FORMULATION

PR-Graph. Amazon builds a product relational graph, namely PR-Graph, to summarize high-level product knowledge for different purposes of research and applications [25, 61]. PR-Graph is represented in the graph format as $\mathcal{G} = (\mathcal{T}, \mathcal{E}, \mathcal{V})$. While $\mathcal{T}$ and $\mathcal{E} \subseteq \mathcal{T} \times \mathcal{T}$ denote the lists of graph nodes and edges, $\mathcal{V} \in \mathbb{R}^{|\mathcal{T}| \times d}$ is the list of d-dimensional feature embeddings associated with nodes in $\mathcal{T}$. Concretely, Amazon categorizes all products into around 15K nodes[1] of $\mathcal{T}$, and creates 417K linkages of $\mathcal{E}$ to generalize the strongly-correlated product-product relations, e.g., *co-purchases*.

Problem Formulation. As $\mathcal{T} = [t_n]_{n=1}^{|\mathcal{T}|}$ denotes all observed nodes in $\mathcal{G}$, for each customer i, his/her all interactions over $\mathcal{T}$ can be snapshot as $\mathcal{T}_i \subseteq \mathcal{T}$, i.e., $\mathcal{T}_i = [t_n^i]_{n=1}^{N_i}$ with N_i elements. Intuitively, based on the topological knowledge in $\mathcal{G}$, customer shopping trajectory $\mathcal{T}_i$ is derived as the unique sub-graph pattern $\mathcal{G}_i = (\mathcal{T}_i, \mathcal{E}_i, \mathcal{V}_i)$. Hence, the goal is to learn the customer trajectory representation from knowledge in $\mathcal{G}_i$, such that the learned representation simultaneously satisfies the following criterion:

- **Trajectory Similarity Measurement.** This provides a macro view of sub-graph structure learning and thus is beneficial for applications, such as customer segmentation, that usually require a holistic measurement of customer-wise similarity.
- **Trajectory Feature Summarization.** This captures the semantics of micro trajectory elements to boost applications such as shopping trajectory completion.

4 C-STAR METHODOLOGY

We start with preliminaries in § 4.1 and formally derive our methodologies in § 4.2 and model training details in § 4.4. We explain the key notations used in this paper in Table 1.

4.1 Preliminaries

Optimal Transport and Wasserstein Metrics. Optimal transport (OT) is the general problem of moving one distribution of mass, e.g., P, to another, e.g., Q, as efficiently as possible. Among all possible transportation plans between P and Q, the one with the minimum cost is called the *optimal transport map*. The derived cost is defined as their distribution distance:

$$W_p\,(P,Q) = \Big(\inf_{f \in TP(P,Q)} \int \|\boldsymbol{x} - f(\boldsymbol{x})\|^p dP(\boldsymbol{x}) \Big)^{\frac{1}{p}},\ p \geq 1, \quad (1)$$

where the infimum is over $TP(P,Q)$ that denotes all transport plans between P and Q. If a minimizer exists, denoted by f^*, it is thus the solution to the OT problem. For one-dimensional distributions, there is a closed-form solution to compute such optimal transport map f^* as $f^*(x) := F_P^{-1}\left(F_Q(x)\right)$; F is the cumulative distribution function (CDF) associated with the underlying distribution.

For the *higher-dimensional* distributions, the metric of *sliced-Wasserstein distance* [32, 35, 43] is introduced and defined as:

$$SW_p\,(P,Q) = \Big(\int_{\mathbb{S}^{d-1}} \left(W_p(g_{\boldsymbol{\theta}\#}P, g_{\boldsymbol{\theta}\#}Q)\right)^p d\boldsymbol{\theta} \Big)^{\frac{1}{p}}, \quad (2)$$

where $g_{\boldsymbol{\theta}\#}P$ denotes the projection of P by function $g_{\boldsymbol{\theta}}$: $\mathbb{R}^d \rightarrow \mathbb{R}$ and $g_{\boldsymbol{\theta}}(\boldsymbol{x}) = \boldsymbol{\theta}^\top \boldsymbol{x}$, where $\boldsymbol{\theta} \in \mathbb{S}^{d-1}$ is a unit vector in the unit d-dimensional hypersphere. Since it satisfies *positive-definiteness*, *symmetry*, and *triangle inequality* [32, 35], it is qualified for distance measurement. Hence, we employ the sliced-Wasserstein distance as the theoretical foundation for our proposed model to capture trajectory similarities.

[1]Data statistics are in § 5.1. [2] Non-bold characters refer to general notations, e.g., scalars, functions and distributions; bold ones highlight high-dimensional objects, e.g., vectors and multivariate random variables.

Table 1: Notations and meanings.

Notation	Explanation
$\mathcal{G}$=$(\mathcal{T}, \mathcal{E}, \mathcal{V})$	PR-Graph with sets of nodes, edges, and features.
$\mathcal{G}_i$=$(\mathcal{T}_i, \mathcal{E}_i, \mathcal{V}_i)$	Customer trajectory pattern.
$\mathcal{T}_i$=$[t_n^i]_{n=1}^{N_i}$	Node list of N_i trajectory elements.
$\mathcal{V}_i$=$[\boldsymbol{v}_n^i]_{n=1}^{N_i}$	Feature list associated with N_i trajectory elements.
$f_\# P$	Pushfoward of distribution P.
$W_p(\cdot,\cdot)$, $SW_p(\cdot,\cdot)$	p-Wasserstein distance, Sliced p-Wasserstein distance.
$F_P(\cdot)$, $F_P^{-1}(\cdot)$	Cumulative distribution function, quantile function.
$\mathbb{R}$, $\mathbb{S}$	Euclidean space and unit hypersphere.
$\boldsymbol{\theta}$	Unit vector in $\mathbb{S}$.
$g_{\boldsymbol{\theta}}(\cdot)$	Linear projection function with parameter vector $\boldsymbol{\theta}$.
P_0, P_i	Reference distribution and input distribution.
$P_0^{\boldsymbol{\theta}}$, $P_i^{\boldsymbol{\theta}}$	The slices of P_0, P_i derived by $\boldsymbol{\theta}$.
$f^*(\cdot)$	Optimal transport map between two distributions.
$\delta(\cdot)$	Dirac delta function.
$\tau(\cdot\vert\cdot)$	Ascending rank in the sorting of the given list.
$\mathcal{V}_i^{\boldsymbol{\theta}}$, $\mathcal{V}_0^{\boldsymbol{\theta}}$	Feature lists associated with distribution slices.
$\mathtt{TSE}(\cdot)$	Trajectory Similarity Encoder.
$\boldsymbol{E}_i$	Embedding of $\mathcal{G}_i$ with trajectory similarity information.
$\boldsymbol{E}_i^{\prime}$	Embedding of $\mathcal{G}_i$ with trajectory structural information.
$\boldsymbol{E}_i^{\star}$	Ultimate trajectory representation.
$\mathcal{L}_{MRL}$, $\mathcal{L}$	Margin ranking loss term and objective function.
Δ	Set of all trainable embeddings and variables.

4.2 Trajectory Distribution Similarity

Consider we have a list of probability measures $[P_i]_{i=1}^M$ defined in $\mathbb{R}^d$ for M observed trajectories in total. For each shopping trajectory, there is a unique associated feature list $\mathcal{V}_i = [\boldsymbol{v}_{t_n^i} \in \mathbb{R}^d]_{n=1}^{N_i}$ with N_i elements. We assume that these feature elements are sampled from the underlying distribution P_i, and what we have snapshot is the *empirical (discrete) distribution* $\widehat{P}_i$ with its empirical CDF as:

$$F_{\widehat{P}_i}(\boldsymbol{x}) = \frac{1}{N_i} \sum\nolimits_{n=1}^{N_i} \delta(\boldsymbol{x} - \boldsymbol{v}_{t_n^i}). \quad (3)$$

$\delta(\cdot)$ returns 1 if the input is zero and 0 otherwise[3]. Generally, we believe these empirical distributions are representative, i.e., $\widehat{P}_i \approx P_i$; thus we would refer P_i to $\widehat{P}_i$ hereafter to avoid clutter notation.

To explicitly measure the trajectory-wise similarity, we propose to compare the input trajectory distribution with a certain *trainable reference* that functions as the "origin" in the trajectory embedding space. Specifically, we introduce a reference distribution P_0 with the embedding list $\mathcal{V}_0 = [\boldsymbol{v}_{t_n^0} \in \mathbb{R}^d]_{n=1}^{N}$, elements in which are the trainable embeddings. Then our target is: to get the distance between the distribution pair (P_0, P_i) to guide the learning of associated trajectory representations $(\boldsymbol{E}_0, \boldsymbol{E}_i)$ with a matched distance measurement back in the embedding space.

Directly solving the high-dimensional optimal transport is extremely difficult, we therefore conduct distribution slicing for computing one-dimensional Wasserstein distance. Let $g_{\boldsymbol{\theta}}(\boldsymbol{x})$ denote the linear projection function, i.e., $g_{\boldsymbol{\theta}}(\boldsymbol{x}) = \boldsymbol{\theta}^\top \boldsymbol{x}$, where $\boldsymbol{\theta} \in \mathbb{S}^{d-1}$ is a unit

[3]It is formally defined as Dirac delta function with $\int \delta(x)dx = 1$ for continuous inputs.

vector in the unit d-dimensional hypersphere. For notation simplicity, we use $P_i^{\boldsymbol{\theta}} := g_{\boldsymbol{\theta}\#}P_i$ to denote the slice of P_i *w.r.t.* $g_{\boldsymbol{\theta}}$ (i.e., $P_i^{\boldsymbol{\theta}}$ is the push-forwarded one-dimensional distribution in $\mathbb{R}$); similarly $P_0^{\boldsymbol{\theta}} := g_{\boldsymbol{\theta}\#}P_0$. To differentiate the high-dimensional input of Eqn.(3), $x^{\boldsymbol{\theta}}$ denotes the projected input that lives in $\mathbb{R}$. For each sliced empirical distribution $P_i^{\boldsymbol{\theta}}$, the corresponding features are $\mathcal{V}_i^{\boldsymbol{\theta}} = [\boldsymbol{\theta}^\top \boldsymbol{v}_{t_n^i}]_{n=1}^{N_i}$. Similarly, the sliced reference list is $\mathcal{V}_0^{\boldsymbol{\theta}} = [\boldsymbol{\theta}^\top \boldsymbol{v}_{t_n^0}]_{n=1}^{N}$. Notice that their empirical CDFs, e.g., $F_{P_0^{\boldsymbol{\theta}}}(x^{\boldsymbol{\theta}}) = \frac{1}{N}\sum_{n=1}^{N}\delta(x^{\boldsymbol{\theta}} - \boldsymbol{\theta}^\top \cdot \boldsymbol{v}_{t_n^0})$, is monotonically increasing. This implies that, if we know the ranking of each input $x^{\boldsymbol{\theta}}$ in the *ascending sorting* of $\mathcal{V}_0^{\boldsymbol{\theta}}$, denoted by $\tau(x^{\boldsymbol{\theta}}|\mathcal{V}_0^{\boldsymbol{\theta}})$, the optimal transport map f^* can be more quantitatively interpreted and approximated for the discrete case as follows: For $\mathcal{V}_i^{\boldsymbol{\theta}}$ and $\mathcal{V}_0^{\boldsymbol{\theta}}$, we implement $f^*(x^{\boldsymbol{\theta}}|\mathcal{V}_i^{\boldsymbol{\theta}}) = F^{-1}_{P_i^{\boldsymbol{\theta}}}(F_{P_0^{\boldsymbol{\theta}}}(x^{\boldsymbol{\theta}}))$ between $(\mathcal{V}_i^{\boldsymbol{\theta}}, \mathcal{V}_0^{\boldsymbol{\theta}})$ with the following mapping process:

$$f^*(x^{\boldsymbol{\theta}}|\mathcal{V}_i^{\boldsymbol{\theta}}) = \text{argmin}_{x' \in \mathcal{V}_i^{\boldsymbol{\theta}}} \left(\tau(x'|\mathcal{V}_i^{\boldsymbol{\theta}}) \geq \frac{N_i}{N} \cdot \tau(x^{\boldsymbol{\theta}}|\mathcal{V}_0^{\boldsymbol{\theta}})\right). \quad (4)$$

Please notice that, the indicator $\tau(\cdot)$ can be actually pre-processed via "`argsort`" to $\mathcal{V}_i^{\boldsymbol{\theta}}$ and "`sort`" to $\mathcal{V}_0^{\boldsymbol{\theta}}$. In Eqn.(4), to align the *feature list cardinalities* (i.e., $N_i \neq |N|$) but not demolish their original semantics, we provide a neat yet effective solution, i.e., conduct *linear interpolation*, as it is essentially a process for data continuing. For other techniques such as data augmentation over latent features [17, 41], they are orthogonal to our contribution and we leave them as future work.

Trajectory Similarity Encoding. For each pair of distribution slices, e.g., $(P_0^{\boldsymbol{\theta}}, P_i^{\boldsymbol{\theta}})$, their optimal transport map produces the shortest one-dimensional distance, i.e., $W_p(P_0^{\boldsymbol{\theta}}, P_i^{\boldsymbol{\theta}})$. According to the theory shown in Eqn.(2), the next step is to traverse all $\boldsymbol{\theta} \in \mathbb{S}^{d-1}$ for the ultimate transport integral between original distributions (P_0, P_i). However, this may be infeasible in practice to have an infinite number of projections drawn from $\mathbb{S}^{d-1}$; therefore, in this work, with $\boldsymbol{\theta}_s$ denoting the s-th projection parameter uniformly sampled from $\mathbb{S}^{d-1}$, we approach this target with the Monte-Carlo approximation. Consequently, this leads to a cumulative sliced-Wasserstein distance between the original trajectory distributions:

$$SW_p(P_0, P_i) \approx \left(\frac{1}{S}\sum_{s=1}^{S} W_p(P_0^{\boldsymbol{\theta}_s}, P_i^{\boldsymbol{\theta}_s})^p\right)^{\frac{1}{p}}. \quad (5)$$

Based on the algorithmic implementation shown in Eqn.(4) with the associated distance regularization, we proceed to encode the trajectory representation accordingly. Let $\Theta = \{\boldsymbol{\theta}_s\}_{s=1}^{S}$ denote the set of sampled projection parameters. Firstly, we encode the vector $\boldsymbol{O} \in \mathbb{R}^{N \cdot S}$ from the embedding reference $\mathcal{V}_0 = [\boldsymbol{v}_{t_n^0}]_{n=1}^{N}$ of P_0 as:

$$\boldsymbol{O} := \frac{1}{SN} \Big\|_{s=1}^{S} \Big\|_{n=1}^{N} \boldsymbol{\theta}_s{}^\top \boldsymbol{v}_{t_n^0}. \quad (6)$$

$\|$ denotes the concatenation operation along the innermost dimension. Given the input feature list $\mathcal{V}_i$, our `Trajectory Similarity Encoder` (`TSE`) is formally defined as follows:

$$\mathsf{TSE}(\mathcal{V}_i|\Theta) := \frac{1}{SN} \Big\|_{s=1}^{S} \Big\|_{n=1}^{N} f^*(\boldsymbol{\theta}_s^\top \boldsymbol{v}_{t_n^0}|\mathcal{V}_i^{\boldsymbol{\theta}_s}) - \boldsymbol{O}. \quad (7)$$

Let $\boldsymbol{E}_i \in \mathbb{R}^{N \cdot S}$ denote the encoded representation from `TSE`. By setting $p = 2$, $\|\boldsymbol{E}_i - \boldsymbol{E}_j\|_2$ is exactly the Euclidean distance form that is more favorable to scenarios for recalling vectorized objects.

Algorithm 1: C-STAR Learning Algorithm.

Input: Trajectories $\{\mathcal{G}_i\}_{i=1}^M$ with corresponding feature lists $\{\mathcal{V}_i\}_{i=1}^M$; variables $\Theta, \Delta, S, N, \mu, L, \rho, \cdots$

```
while not converge do
    for each trajectory G_i ∈ {G_i}_{i=1}^M do
        G_i ← Sampled trajectory of G_i ;
        V_i ← Updated feature list associated with G_i ;
        E_i ← TSE(V_i|Θ) ;                                   ▷ Eqn.(7)
        E'_i ← Encode from GCN [31] E*_i ← [E_i, E'_i] ;
        G_j, G_k ← Positive and negative samples from Ω+_i and Ω-_i ;
        E*_j, E*_k ← Trajectory representations of G_j and G_k ;
        D(G_i, G_j) ← ||E*_i − E*_j||_2 ;
        D(G_i, G_k) ← ||E*_i − E*_k||_2 ;
        L_MRL ← Update the margin ranking loss ;             ▷ Eqn.(9)
    L ← Optimize C-STAR with regularization ;                ▷ Eqn.(10)
return Well trained model C-STAR.
```

4.3 Trajectory Topological Semantics

Since customer trajectories are induced from PR-Graph, they essentially inherit the semantics embedded in PR-Graph. To fuse such knowledge and enrich the ultimate trajectory representations, we propose to learn the trajectory knowledgeable features.

We employ the graph convolutional paradigm due to its powerful ability to learn high-order graph information [60]. The general idea of Graph Convolution Network (GCN) is to encapsulate graph information into condensed outputs, via iteratively propagating and aggregating latent features of node neighbors via the graph topology [9, 23, 31, 56]:

$$\boldsymbol{v}_i^{(l)} = AGG\left(\boldsymbol{v}_i^{(l-1)}, \left\{\boldsymbol{v}_j^{(l-1)} : j \in \mathcal{N}(i)\right\}\right), \quad (8)$$

where $\boldsymbol{v}_i^{(l)} \in \mathbb{R}^d$ denotes node i's embedding after l-th iteration of graph convolutions, indexed in the input embedding table. $\mathcal{N}(i)$ is the set of i's neighbors. Function $AGG(\cdot,\cdot)$ is the information aggregation function, mainly aiming to transform the center node feature and the neighbor features.

To further learn the semantic knowledge of PR-Graph, we learn the representations output from the classic GCN [31], denoted by $\boldsymbol{E}'_i = \boldsymbol{v}_i^{(L)}$ after L layer of graph convolutions. We then complete the ultimate trajectory representation $\boldsymbol{E}_i^\star$ as: $\boldsymbol{E}_i^\star = [\boldsymbol{E}_i, \boldsymbol{E}'_i] \in \mathbb{R}^{2NS}$.

4.4 Model Training

Although C-STAR takes *variable-length* trajectory inputs for representation encoding, it is however more efficient and common to use fixed-size tensors, i.e., batches of trajectory feature lists, for model training. In this paper, we adopt the uniform sampling to make sure the sampled trajectories are representative and informative.

Objective Function. We proceed to the optimization paradigm of Margin Ranking Loss (MRL) with negative sampling:

$$\mathcal{L}_{MRL} = \sum_{i=1}^{M} \sum_{\mathcal{G}_j \in \Omega_i^+,\, \mathcal{G}_k \in \Omega_i^-} \max\left(0, D(\mathcal{G}_i, \mathcal{G}_j) - D(\mathcal{G}_i, \mathcal{G}_k) + margin\right), \quad (9)$$

where Ω_i^+, Ω_i^- are the sets of positive and negative samples associated with trajectory $\mathcal{G}_i$. Here $D(\cdot,\cdot)$ is the Euclidean distance computed from the trajectory representations, i.e., $\|\boldsymbol{E}_i^\star - \boldsymbol{E}_j^\star\|_2$. Then the complete objective function is formulated as:

$$\mathcal{L} = \mathcal{L}_{MRL} + \mu\|\Delta\|_2^2. \quad (10)$$

$\|\Delta\|_2^2$ is the L2-regularizer of all trainable embeddings and variables parameterized by hyper-parameter μ to avoid over-fitting. The pseudo-codes for training C-STAR are detailed in Algorithm 1.

5 EXPERIMENTS

Our experimental objective is to investigate the effectiveness of the proposed framework C-STAR in producing high quality embeddings serving different down-stream tasks. We propose and systematically evaluate the model performance on two tasks, both on Amazon internal datasets and publicly available datasets.

5.1 Experimental Setups

Evaluation Tasks and Metrics. In this work, we propose two tasks to evaluate the learned trajectory embedding quality. These tasks are important for customer understanding efforts. Specifically,

- **Task 1: Customer Segmentation.** The fundamental property required by customer segmentation is customer-wise similarity measurement. Thus, we propose this task to evaluate the model ranking capability, in which given a query customer, the model seeks to retrieve his/her Top-K most similar customers, based on their learned trajectory embeddings.
- **Task 2: Shopping Trajectory Completion.** Assuming the customers' shopping journeys have not yet finished, it aims to complete customers' trajectories by recommending relevant yet unexplored elements. This task is also formulated as Top-K retrieval.

We formulate Task 1 and 2 as ranking towards candidates of similar/relevant customers and product categories, respectively. Thus, Recall@K and NDCG@K are utilized as evaluation metrics.

Baselines. We include the following models: (1) shallow neural models (TPooling and MLP); (2) graph-based models (GCN$^+$, GAT$^+$, GraphSage$^+$); (3) language-based models (Transformer, Graph Transformer); and (4) general deep learning models (DeepSets, PSWE).

- **TPooling** is a straightforward implementation that aggregates all element embeddings of each customer trajectory. The pooling strategy could be *mean*, *max*, or *min*. We report the best performance of these strategies and denote it as TPooling.
- **MLP** is a fundamental neural network that first concatenates trajectory element embeddings as input and passes them through one hidden layer and finally arrives at the output layer.
- **GCN** [31] is one of the classic graph convolutional networks. We implement it on PR-Graph to gather information and further aggregate trajectory-level embeddings via TPooling (i.e., **GCN+TPooling**) or MLP (i.e., **GCN+MLP**). We use the notation **GCN$^+$** to denote the one with better metrics (e.g., on Recall@K and NDCG@K in ranking tasks).
- **GAT** [56] is the representative graph-based model with the attention mechanism. Similarly, we implement it for PR-Graph information propagation and summarize trajectory embeddings with two variant models (i.e., **GAT+TPooling** and **GAT+MLP**). Similarly, **GAT$^+$** denotes the better variant.
- **GraphSage** [23] is the graph convolutional network with the inductive learning setting. Similarly, we have two implementations with TPooling and MLP, and **GraphSage$^+$** is the better one.
- **Transformer** [55], denoted by TRFM, is another strong baseline with the self-attention mechanism. In our implementation, we input each customer trajectory as a language sentence to learn its embedding.
- **Graph Transformer** [19] is one of state-of-the-art Transformer-based model that deploys on the graph data. We implement it on PR-Graphand trajectory data to jointly learn the representations. We denote it as GTRFM.
- **DeepSets** [67] is an exemplary deep learning model that is originally proposed to learn representations for "*compound objects*" such as point clouds. In our experiments, it takes all trajectory units as a set and learns the unified representation while maintaining the intrinsic semantics of trajectory elements.
- **PSWE** [43] is the latest state-of-the-art method that subsumes the learning process under the Wasserstein metric framework. In this work, we reproduce it to learn the trajectory embeddings.

We exclude early collaborative-filtering-based methods [28, 36, 45] and recent GCN-based recommender models[27, 57, 66]. The reason is that these methods are *transductive*, which develops recommendations only for observed customers, but finds challenges in generalizing to unseen ones. Note that customer shopping trajectories evolve quickly over time, we therefore require a method that poses a good capability of doing *inductive* inference.

5.2 Evaluation on Amazon Datasets

In this section, we provide the empirical model analyses on Amazon data. For each task, we explain the evaluation protocol followed by the discussions of experimental results. We report the average results based on five times of training and evaluation in Tables 3-4, where the bold and the underlined represent the best- and second-best-performing cases.

Dataset Statistics. To prevent the risk of data leakage, we split data separately for different evaluation tasks with their statistics reported in Table 2. PR-Graph, as the prior knowledge, is universal throughout all tasks. We use customer engagements for the period of 28 days whereby the data is fully anonymized.

Table 2: The statistics of Amazon internal datasets.

	Training		PR-Graph statistics			Evaluation	
	#Instances	#Length	#Nodes	#Edges	#Density	#Instances	#Length
Task 1	100,000	27.516	14,695	416,610	0.0386	1,000,000	29.218
Task 2	100,000	27.481				5,000,000	26.517

Table 3: Results of customer segmentation task (%).

Metric	Top-5 Recall	Top-5 NDCG	Top-10 Recall	Top-10 NDCG	Top-20 Recall	Top-20 NDCG	Top-50 Recall	Top-50 NDCG	Top-100 Recall	Top-100 NDCG
TPooling	1.30	2.61	2.61	3.88	5.84	5.51	9.58	5.62	15.30	8.76
MLP	1.23	1.39	2.53	1.87	4.55	2.39	8.45	3.31	13.92	7.67
GCN$^+$	3.48	4.44	5.62	5.33	8.94	7.61	15.62	8.16	16.50	10.72
GAT$^+$	4.14	5.53	6.82	6.28	9.23	8.36	16.16	8.48	16.89	11.31
GraphSage$^+$	3.93	4.72	6.29	5.84	9.17	7.82	15.82	8.37	16.74	10.98
TRFM	5.59	8.64	9.52	8.83	12.21	11.05	21.54	15.38	30.53	17.96
GTRFM	5.18	8.37	9.33	8.27	11.26	10.67	20.66	14.83	28.77	17.21
DeepSets	5.77	8.87	9.75	9.11	12.47	11.29	21.96	15.60	31.04	18.17
PSWE	<u>6.19</u>	<u>9.87</u>	<u>11.41</u>	<u>9.94</u>	<u>14.67</u>	<u>13.85</u>	<u>24.19</u>	<u>17.27</u>	<u>33.22</u>	<u>20.57</u>
C-STAR	**6.82**	**10.74**	**12.40**	**10.78**	**16.12**	**15.29**	**26.33**	**19.43**	**34.87**	**21.46**
% Gain	10.18%	8.81%	8.68%	8.45%	9.88%	10.40%	8.87%	14.88%	9.91%	4.33%

Table 4: Results of shopping trajectory completion task (%).

	Top-5		Top-10		Top-20		Top-50		Top-100	
Metric	Recall	NDCG	Recall	NDCG	Recall	NDCG	Recall	NDCG	Recall	NDCG
TPooling	5.97	7.31	9.13	8.02	13.71	9.62	22.43	12.36	31.37	14.70
MLP	5.84	7.11	8.78	7.80	13.56	9.37	22.28	12.14	30.96	14.44
GCN+	6.74	8.35	10.64	9.13	15.25	10.79	25.27	13.59	35.69	15.98
GAT+	6.99	8.40	11.03	9.44	16.97	11.10	26.16	14.23	36.21	16.76
GraphSage+	6.73	8.48	10.79	9.26	16.48	11.20	26.11	14.22	36.16	16.74
TRFM	7.26	9.11	11.72	9.85	17.47	11.85	27.66	14.98	37.97	18.15
GTRFM	6.81	8.69	10.83	9.34	16.53	11.18	26.52	14.28	36.84	17.11
DeepSets	6.29	7.60	9.86	8.43	15.01	10.21	24.96	13.32	35.12	15.97
PSWE	7.87	9.27	12.11	10.25	17.87	12.26	28.09	15.23	38.05	18.13
C-STAR	**8.05**	**9.40**	**12.49**	**10.31**	**18.29**	**12.44**	**28.65**	**16.02**	**38.23**	**18.38**
% Gain	*2.29%*	*1.40%*	*3.14%*	*0.59%*	*2.35%*	*1.47%*	*1.99%*	*5.19%*	*0.47%*	*1.27%*

Table 5: Results of ablation study.

	Task 1		Task 2	
	Recall	NDCG	Recall	NDCG
w/o KE	33.41 (-4.19%)	20.92 (-2.52%)	33.94 (-11.22%)	17.18 (-6.53%)
w/o TSE	22.86 (-34.44%)	14.77 (-31.17%)	32.41 (-15.22%)	16.60 (-9.68%)
Best	**34.87**	**21.46**	**38.23**	**18.38**

Table 6: Public dataset statistics.

	BCrossing	**Gowalla**	**Pinterest**	**AMZ-Book**
#User trajectories	16,411	29,858	55,186	52,643
# Items	36,143	40,919	9,855	91,576
#Avg. trajectory length	35.711	49.272	26.516	56.686
# PR-Graph density	0.000119	0.000127	0.000372	0.0000648

5.2.1 Task 1: Customer Segmentation. As mentioned in § 5.1, the learned embeddings are expected to reflect realistic trajectory similarity, *both structurally and semantically*. For 1M evaluation data, we sort out their similar trajectories according to the number of overlapping trajectory elements; then we compare these ranking lists with the Top-K results obtained by the learning models.

As reported in Table 3, (1) assembling with TPooling or MLP, graph-based implementations (i.e., GCN+, GAT+, and GraphSage+) perform better than these two vanilla baselines, indicating that graph convolutional operations can effectively extract knowledge from PR-Graph to boost model performance. (2) Representative language models generally underperform state-of-the-art deep learning models (i.e., DeepSets and PSWE) for trajectory representation learning. One explanation is that these deep learning methods organize the trajectory into the set structure, which can well capture their collective information and thus improve their trajectory-wise similarity measurement. (3) Our C-STAR model consistently outperforms the second-best model by 8.68%~10.18% and 4.33%~14.88% *w.r.t.* Recall@K and NDCG@K (with K ranging in {5, 10, 20, 50, 100}). This validates C-STAR's effectiveness of jointly considering both the trajectory semantics and trajectory similarity, which not only enriches the latent semantics of trajectory embeddings but also well approximates the actual trajectory distribution proximity.

5.2.2 Task 2: Shopping Trajectory Completion. We collect 5M shopping trajectories and randomly hide 20% percent of trajectory elements; then we employ the trained models to predict the missing ones for trajectory completion, just like a "Cloze task".

From Table 4, we have two major observations. (1) Different from the under-performing situation in Task 1, language models, e.g., Transformer, work better in Task 2, compared to some recent deep-learning-based models. The main reason is that, these language models can well capture the semantic relations between a "trajectory" and its "elements", similar to the case between a "sentence" and its "words". (2) The state-of-the-art model PSWE generally performs the best throughout all competing models; meanwhile, our proposed model C-STAR further achieves at least 0.47% and 0.59% of improvements over Recall and NDCG metrics. Additionally, the stable performance of C-STAR to predict next-20% trajectory elements also provides the deployment flexibility for *bundle recommendation*.

5.3 Ablation Studies

We conduct ablation studies and report the results of overlapping structure similarity and trajectory completion in Table 5.

PR-Graph Necessity for Semantics Extraction. We omit the graph convolutions over PR-Graph and denote the variant as C-STAR$_{\text{w/o KE}}$. Specifically, we remove the representation $\boldsymbol{E}_i^{'}$ in $\boldsymbol{E}_i^{\star} = [\boldsymbol{E}_i, \boldsymbol{E}_i^{'}]$; and, to provide a fair comparison, we expand the dimensionality of $\boldsymbol{E}_i$ to $2NS$. As shown in Table 5, C-STAR$_{\text{w/o KE}}$ exhibits a conspicuous performance decay. This not only indicates the informativeness of PR-Graph in organizing multiple product-to-product relations at the category-level, but also the usefulness of graph convolutions for knowledge extraction.

Implementation Effectiveness of `TSE` **Module.** Variant C-STAR$_{\text{w/o TSE}}$ replaces the algorithmic implementation of Eqn.(7) by a two-layer of MLP to encode trajectory representations and measure trajectory distribution similarity. The performance gaps of these tasks between C-STAR$_{\text{w/o TSE}}$ and C-STAR prove the effectiveness of our proposed solution, in which we convert the measurement of trajectory-wise similarity to the *distribution distance* with the Optimal Transport methodology.

5.4 Evaluation on Public Datasets

Dataset Statistics. We collect four public datasets that are widely-evaluated [8, 9, 27, 62, 63, 69]. For these datasets, we synthesize their own "PR-Graph" by *creating edges if items are co-purchased by at least 20 different customers*. Dataset statistics are reported in Table 6.

- **BCrossing**[4] [77] is a public dataset of book ratings in Book-Crossing Community. To guarantee the dataset quality, we filter out readers and books with less than five interactions and then merge each reader's rated books into a unique trajectory.
- **Gowalla**[5] [27, 57, 58] is the check-in dataset [14] from Gowalla, where users share their locations by check-in. We directly use the dataset split by [57] where users and items are selected to have at least then interactions. We integrate each customer's all check-in locations into his/her trajectory.
- **Pinterest**[6] [22] is an implicit feedback dataset for image recommendation [22]. Each user is associated with his/her own trajectory towards 9,855 different images.

[4]https://www.kaggle.com/datasets/ruchi798/bookcrossing-dataset
[5]https://github.com/gusye1234/LightGCN-PyTorch/tree/master/data/gowalla
[6]https://sites.google.com/site/xueatalphabeta/dataset-1/pinterest_iccv

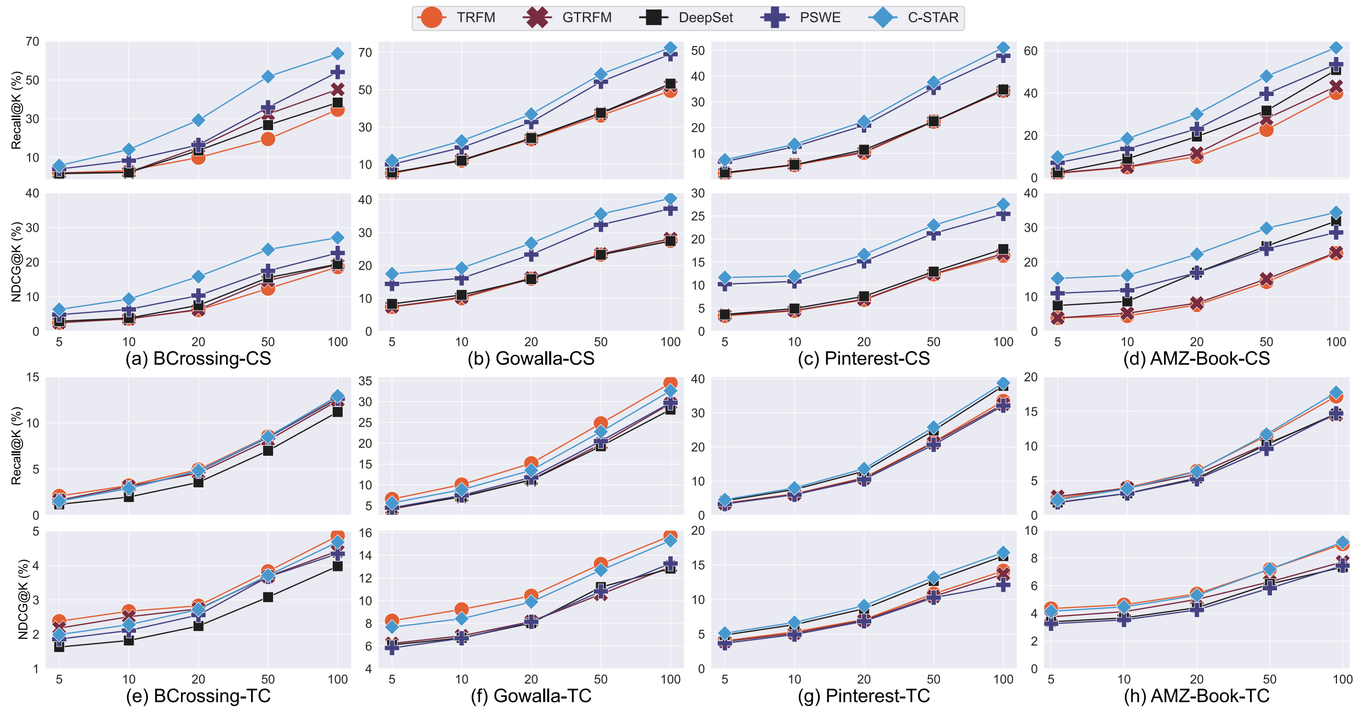

Figure 2: Recall@K and NDCG@K are respectively reported in the first and second row (best view in color).

- **AMZ-Book**[7] is the book review dataset between readers and book trajectories, organized from the book collection of Amazon-review [26]. We directly use the existing data split from [57], where each reader and book have at least ten interactions.

Evaluation Results. For these public datasets, we report recent language- and deep-learning-models with good performance in § 5 as competing methods. The evaluation protocols follow closely to the experiments with Amazon datasets, unless explicitly specified otherwise. There are two major observations.

First of all, the findings depicted in Figure 2(a)-(d) provide insights into the performance of various methodologies employed for the task of customer segmentation. Within this context, deep-learning-based approaches, including DeepSet and PSWE, demonstrate a notable advantage over conventional language-based methods such as Transformer and Graph Transformer. Notably, the employment of the Wasserstein-metric-based model PSWE yields even more promising results than DeepSets, reinforcing the superiority of deep-learning-based techniques for this task. Moreover, it is important to highlight that our model consistently outperforms the alternative methodologies across the evaluated public datasets, showing the efficacy of our proposed method in capturing trajectory-wise similarity in Customer Segmentation.

Second of all, from Figure 2(e)-(h), we notice that, in concurrence with the observations made in Section 5.2.2, it is evident that conventional language-based models exhibit competitive performance in this specific task. Our proposed model stands out as the top-performing approach among all the comparative models, with the exception of the Gowalla dataset where it ranks as the second-best performer. This consistent demonstration of superior performance across multiple datasets reinforces the effectiveness of our model in tackling the shopping trajectory completion task. Moreover, considering the superior performance in Customer Segmentation, our model essential exhibits a remarkable balance and adaptability across both tasks evaluated in this work.

6 CONCLUSION AND FUTURE WORK

In this paper, we present an end-to-end framework for customer shopping trajectory representation learning, namely C-STAR. The proposed methodology jointly captures the trajectory distribution similarity and the trajectory topological semantics, enriching the trajectory representations in a coarse-to-fine learning paradigm. The empirical results on both Amazon internal data and public datasets not only illustrate the usefulness of learned embeddings over two customer-centric evaluation tasks, but they also justify the effectiveness of all proposed modules.

As for future work, we plan to investigate two major directions. (1) It is worth integrating *temporal information* for model improvement to forecast the future trajectory evolution. Specifically, we may need to quantitatively integrate the appearance timestamp of each trajectory element, which is more complicated to deal with, as the model needs to understand and utilize the signals behind different time gaps. (2) In practice, trajectory data is continuously evolving. Instead of re-training the model, streaming methods via *Continual Learning* [49] can be more efficient to capture new emerging patterns while also maintaining the early model knowledge, which is particularly efficacious for large-scale settings.

[7] https://github.com/gusye1234/LightGCN-PyTorch/tree/master/data/amazon-book

ACKNOWLEDGMENTS

Yankai Chen and Irwin King were partially supported by grants from the National Key Research and Development Program of China (No. 2018AAA0100204) and from the Research Grants Council of the Hong Kong Special Administrative Region, China (CUHK 14222922, RGC GRF, No. 2151185).

REFERENCES

[1] Martin Arjovsky, Soumith Chintala, and Léon Bottou. 2017. Wasserstein generative adversarial networks. In ICML. PMLR, 214–223.
[2] James Atwood and Don Towsley. 2016. Diffusion-convolutional neural networks. NeurIPS 29 (2016).
[3] Nicolas Bonneel, Julien Rabin, Gabriel Peyré, and Hanspeter Pfister. 2015. Sliced and radon wasserstein barycenters of measures. JMIV 51, 1 (2015), 22–45.
[4] Joan Bruna, Wojciech Zaremba, Arthur Szlam, and Yann LeCun. 2014. Spectral networks and locally connected networks on graphs. ICLR (2014).
[5] Mathieu Carriere, Marco Cuturi, and Steve Oudot. 2017. Sliced Wasserstein kernel for persistence diagrams. In ICML. PMLR, 664–673.
[6] Wei-Cheng Chang, Daniel Jiang, Hsiang-Fu Yu, Choon Hui Teo, Jiong Zhang, Kai Zhong, Kedarnath Kolluri, Qie Hu, Nikhil Shandilya, Vyacheslav Ievgrafov, et al. 2021. Extreme multi-label learning for semantic matching in product search. In SIGKDD. 2643–2651.
[7] Yankai Chen, Yixiang Fang, Reynold Cheng, Yun Li, Xiaojun Chen, and Jie Zhang. 2018. Exploring communities in large profiled graphs. TKDE (2018).
[8] Yankai Chen, Yixiang Fang, Yifei Zhang, and Irwin King. 2023. Bipartite Graph Convolutional Hashing for Effective and Efficient Top-N Search in Hamming Space. In WebConf. ACM.
[9] Yankai Chen, Huifeng Guo, Yingxue Zhang, Chen Ma, Ruiming Tang, Jingjie Li, and Irwin King. 2022. Learning binarized graph representations with multi-faceted quantization reinforcement for top-k recommendation. In SIGKDD.
[10] Yankai Chen, Menglin Yang, Yingxue Zhang, Mengchen Zhao, Ziqiao Meng, Jianye Hao, and Irwin King. 2022. Modeling Scale-free Graphs with Hyperbolic Geometry for Knowledge-aware Recommendation. WSDM.
[11] Yankai Chen, Jie Zhang, Yixiang Fang, Xin Cao, and Irwin King. 2021. Efficient community search over large directed graphs: An augmented index-based approach. In IJCAI. 3544–3550.
[12] Yankai Chen, Yifei Zhang, Menglin Yang, Zixing Song, Chen Ma, and Irwin King. 2023. WSFE: Wasserstein Sub-graph Feature Encoder for Effective User Segmentation in Collaborative Filtering. SIGIR (2023).
[13] Kewei Cheng, Xian Li, Yifan Ethan Xu, Xin Luna Dong, and Yizhou Sun. 2022. Robust Product Graph Embedding Learning for Error Detection. VLDB (2022).
[14] Eunjoon Cho, Seth A Myers, and Jure Leskovec. 2011. Friendship and mobility: user movement in location-based social networks. In SIGKDD. 1082–1090.
[15] Marco Cuturi. 2013. Sinkhorn distances: Lightspeed computation of optimal transport. NeurIPS 26 (2013).
[16] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. 2016. CNNs on graphs with fast localized spectral filtering. NeurIPS 29 (2016).
[17] Terrance DeVries and Graham W Taylor. 2017. Dataset augmentation in feature space. ICLR (Workshop) (2017).
[18] Xin Luna Dong, Xiang He, Andrey Kan, Xian Li, Yan Liang, Jun Ma, Yifan Ethan Xu, Chenwei Zhang, Tong Zhao, Gabriel Blanco Saldana, et al. 2020. Self-driving knowledge collection for products of thousands of types. In SIGKDD. 2724–2734.
[19] Vijay Prakash Dwivedi and Xavier Bresson. 2020. A generalization of transformer networks to graphs. arXiv preprint arXiv:2012.09699 (2020).
[20] Dominik Maria Endres and Johannes E Schindelin. 2003. A new metric for probability distributions. IEEE TIT 49, 7 (2003), 1858–1860.
[21] Yixiang Fang, Reynold Cheng, Yankai Chen, Siqiang Luo, and Jiafeng Hu. 2017. Effective and efficient attributed community search. The VLDB Journal 26 (2017), 803–828.
[22] Xue Geng, Hanwang Zhang, Jingwen Bian, and Tat-Seng Chua. 2015. Learning image and user features for recommendation in social networks. In ICCV.
[23] William L Hamilton, Rex Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. In NeurIPS. 1025–1035.
[24] William L Hamilton, Rex Ying, and Jure Leskovec. 2017. Representation learning on graphs: Methods and applications. arXiv preprint arXiv:1709.05584 (2017).
[25] Junheng Hao, Tong Zhao, Jin Li, Xin Luna Dong, Christos Faloutsos, Yizhou Sun, and Wei Wang. 2020. P-companion: A principled framework for diversified complementary product recommendation. In CIKM. 2517–2524.
[26] Ruining He and Julian McAuley. 2016. Modeling the visual evolution of fashion trends with one-class collaborative filtering. In WWW. 507–517.
[27] 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. 639–648.
[28] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural collaborative filtering. In WWW. 173–182.
[29] Ernst Hellinger. 1909. Neue begründung der theorie quadratischer formen von unendlichvielen veränderlichen. Journal für die reine und angewandte Mathematik 1909, 136 (1909), 210–271.
[30] Leonid V Kantorovich. 1960. Mathematical methods of organizing and planning production. Management science 6, 4 (1960), 366–422.
[31] Thomas N. Kipf and Max Welling. 2016. Semi-Supervised Classification with Graph Convolutional Networks. In ICLR.
[32] Soheil Kolouri, Kimia Nadjahi, Umut Simsekli, Roland Badeau, and Gustavo Rohde. 2019. Generalized sliced wasserstein distances. NeurIPS 32 (2019).
[33] Soheil Kolouri, Phillip E Pope, Charles E Martin, and Gustavo K Rohde. 2018. Sliced Wasserstein auto-encoders. In ICLR.
[34] Soheil Kolouri, Akif B Tosun, John A Ozolek, and Gustavo K Rohde. 2016. A continuous linear optimal transport approach for pattern analysis in image datasets. Pattern recognition 51 (2016), 453–462.
[35] Soheil Kolouri, Yang Zou, and Gustavo K Rohde. 2016. Sliced Wasserstein kernels for probability distributions. In CVPR. 5258–5267.
[36] Yehuda Koren, Robert Bell, and Chris Volinsky. 2009. Matrix factorization techniques for recommender systems. Computer 42, 8 (2009), 30–37.
[37] Solomon Kullback and Richard A Leibler. 1951. On information and sufficiency. The annals of mathematical statistics 22, 1 (1951), 79–86.
[38] Jiacheng Li, Ming Wang, Jin Li, Jinmiao Fu, Xin Shen, Jingbo Shang, and Julian McAuley. 2023. Text Is All You Need: Learning Language Representations for Sequential Recommendation. arXiv preprint arXiv:2305.13731 (2023).
[39] Jiacheng Li, Tong Zhao, Jin Li, Jim Chan, Christos Faloutsos, George Karypis, Soo-Min Pantel, and Julian McAuley. 2022. Coarse-to-Fine Sparse Sequential Recommendation. SIGIR (2022).
[40] Rongmei Lin, Xiang He, Jie Feng, Nasser Zalmout, Yan Liang, Li Xiong, and Xin Luna Dong. 2021. PAM: understanding product images in cross product category attribute extraction. In SIGKDD. 3262–3270.
[41] Bo Liu, Xudong Wang, Mandar Dixit, Roland Kwitt, and Nuno Vasconcelos. 2018. Feature space transfer for data augmentation. In CVPR. 9090–9098.
[42] Antoine Liutkus, Umut Simsekli, Szymon Majewski, Alain Durmus, and Fabian-Robert Stöter. 2019. Sliced-Wasserstein flows: Nonparametric generative modeling via optimal transport and diffusions. In ICML. PMLR, 4104–4113.
[43] Navid Naderializadeh, Joseph F Comer, Reed Andrews, Heiko Hoffmann, and Soheil Kolouri. 2021. Pooling by sliced-Wasserstein embedding. NeurIPS 34 (2021), 3389–3400.
[44] Priyanka Nigam, Yiwei Song, Vijai Mohan, Vihan Lakshman, Weitian Ding, Ankit Shingavi, Choon Hui Teo, Hao Gu, and Bing Yin. 2019. Semantic product search. In SIGKDD. 2876–2885.
[45] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2012. BPR: Bayesian personalized ranking from implicit feedback. arXiv (2012).
[46] Xin Shen, Praful Agrawal, and Zhongwei Cheng. 2023. Data Efficient Training with Imbalanced Label Sample Distribution for Fashion Detection. arXiv:2305.04379 [cs.CV]
[47] Xin Shen, Kyungdon Joo, and Jean Oh. 2023. FishRecGAN: An End to End GAN Based Network for Fisheye Rectification and Calibration. arXiv preprint arXiv:2305.05222 (2023).
[48] Xin Shen, Yan Zhao, Sujan Perera, Yujia Liu, Jinyun Yan, and Mitchell Goodman. 2023. Learning Personalized Page Content Ranking Using Customer Representation. arXiv preprint arXiv:2305.05267 (2023).
[49] Hanul Shin, Jung Kwon Lee, Jaehong Kim, and Jiwon Kim. 2017. Continual learning with deep generative replay. NeurIPS 30 (2017).
[50] Justin Solomon, Fernando De Goes, Gabriel Peyré, Marco Cuturi, Adrian Butscher, Andy Nguyen, Tao Du, and Leonidas Guibas. 2015. Convolutional wasserstein distances: Efficient optimal transportation on geometric domains. ACM Transactions on Graphics (ToG) 34, 4 (2015), 1–11.
[51] Zixing Song, Ziqiao Meng, Yifei Zhang, and Irwin King. 2021. Semi-supervised multi-label learning for graph-structured data. In CIKM. 1723–1733.
[52] Zixing Song, Yifei Zhang, and Irwin King. 2022. Towards an optimal asymmetric graph structure for robust semi-supervised node classification. In SIGKDD. 1656–1665.
[53] Ilya Tolstikhin, Olivier Bousquet, Sylvain Gelly, and Bernhard Schoelkopf. 2018. Wasserstein auto-encoders. ICLR (2018).
[54] Quoc-Tuan Truong, Tong Zhao, Changhe Yuan, Jin Li, Jim Chan, Soo-Min Pantel, and Hady W Lauw. 2022. AmpSum: Adaptive Multiple-Product Summarization towards Improving Recommendation Captions. In WebConf. 2978–2988.
[55] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. NeurIPS 30 (2017).
[56] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. 2017. Graph attention networks. ICLR.
[57] Xiang Wang, Xiangnan He, Meng Wang, Fuli Feng, and Tat-Seng Chua. 2019. Neural graph collaborative filtering. In SIGIR. 165–174.
[58] Xiang Wang, Hongye Jin, An Zhang, Xiangnan He, Tong Xu, and Tat-Seng Chua. 2020. Disentangled graph collaborative filtering. In SIGIR. 1001–1010.
[59] Yikun Xian, Tong Zhao, Jin Li, Jim Chan, Andrey Kan, Jun Ma, Xin Luna Dong, Christos Faloutsos, George Karypis, Shan Muthukrishnan, et al. 2021. Ex3:

Explainable attribute-aware item-set recommendations. In RecSys. 484–494.
[60] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. 2019. How powerful are graph neural networks? ICLR (2019).
[61] An Yan, Chaosheng Dong, Yan Gao, Jinmiao Fu, Tong Zhao, Yi Sun, and Julian McAuley. 2022. Personalized complementary product recommendation. In WebConf. 146–151.
[62] Menglin Yang, Zhihao Li, Min Zhou, Jiahong Liu, and Irwin King. 2022. Hicf: Hyperbolic informative collaborative filtering. In SIGKDD. 2212–2221.
[63] Menglin Yang, Min Zhou, Jiahong Liu, Defu Lian, and Irwin King. 2022. HRCF: Enhancing collaborative filtering via hyperbolic geometric regularization. In WebConf. 2462–2471.
[64] Menglin Yang, Min Zhou, Lujia Pan, and Irwin King. 2023. κHGCN: Tree-likeness Modeling via Continuous and Discrete Curvature Learning. arXiv:2212.01793
[65] Menglin Yang, Min Zhou, Rex Ying, Yankai Chen, and Irwin King. 2023. Hyperbolic Representation Learning: Revisiting and Advancing. arXiv preprint arXiv:2306.09118 (2023).
[66] Rex Ying, Ruining He, Kaifeng Chen, Pong Eksombatchai, William L Hamilton, and Jure Leskovec. 2018. Graph convolutional neural networks for web-scale recommender systems. In SIGKDD. 974–983.
[67] Manzil Zaheer, Satwik Kottur, Siamak Ravanbakhsh, Barnabas Poczos, Russ R Salakhutdinov, and Alexander J Smola. 2017. Deep sets. NeurIPS 30 (2017).
[68] Nasser Zalmout, Chenwei Zhang, Xian Li, Yan Liang, and Xin Luna Dong. 2021. All You Need to Know to Build a Product KG. In SIGKDD. 4090–4091.
[69] Xinni Zhang, Yankai Chen, Cuiyun Gao, Qing Liao, Shenglin Zhao, and Irwin King. 2022. Knowledge-aware Neural Networks with Personalized Feature Referencing for Cold-start Recommendation. arXiv preprint arXiv:2209.13973 (2022).
[70] Xinyang Zhang, Chenwei Zhang, Xian Li, Xin Luna Dong, Jingbo Shang, Christos Faloutsos, and Jiawei Han. 2022. OA-Mine: Open-World Attribute Mining for E-Commerce Products with Weak Supervision. In WebConf. 3153–3161.
[71] Yifei Zhang, Yankai Chen, Zixing Song, and Irwin King. 2023. Contrastive Cross-scale Graph Knowledge Synergy. arxiv (2023).
[72] Yifei Zhang and Hao Zhu. 2020. Discrete wasserstein autoencoders for document retrieval. In ICASSP. IEEE, 8159–8163.
[73] Yifei Zhang, Hao Zhu, Ziqiao Meng, Piotr Koniusz, and Irwin King. 2022. Graph-adaptive rectified linear unit for graph neural networks. In WebConf. 1331–1339.
[74] Yifei Zhang, Hao Zhu, Zixing Song, Piotr Koniusz, and Irwin King. 2022. COSTA: Covariance-Preserving Feature Augmentation for Graph Contrastive Learning. In SIGKDD. 2524–2534.
[75] Yifei Zhang, Hao Zhu, Zixing Song, Piotr Koniusz, and Irwin King. 2022. Spectral Feature Augmentation for Graph Contrastive Learning and Beyond. arXiv preprint arXiv:2212.01026 (2022).
[76] Guineng Zheng, Subhabrata Mukherjee, Xin Luna Dong, and Feifei Li. 2018. Open attribute value extraction from product profiles. In SIGKDD. 1049–1058.
[77] Cai-Nicolas Ziegler, Sean M McNee, Joseph A Konstan, and Georg Lausen. 2005. Improving recommendation lists through topic diversification. In WWW. 22–32.