

# HSR: Hyperbolic Social Recommender

Anchen Li

College of Computer Science and Technology, Jilin  
University, China  
liac20@mails.jlu.edu.cn

Bo Yang\*

College of Computer Science and Technology, Jilin  
University, China  
ybo@jlu.edu.cn

## ABSTRACT

With the prevalence of online social media, users' social connections have been widely studied and utilized to enhance the performance of recommender systems. In this paper, we explore the use of hyperbolic geometry for social recommendation. We present **Hyperbolic Social Recommender (HSR)**, a novel social recommendation framework that utilizes hyperbolic geometry to boost the performance. With the help of hyperbolic spaces, HSR can learn high-quality user and item representations for better modeling user-item interaction and user-user social relations. Via a series of extensive experiments, we show that our proposed HSR outperforms its Euclidean counterpart and state-of-the-art social recommenders in click-through rate prediction and top- $K$  recommendation, demonstrating the effectiveness of social recommendation in the hyperbolic space.

## KEYWORDS

Social Recommendation; Social Network; Hyperbolic Space.

## 1 INTRODUCTION

In the era of information explosion, recommender systems have been playing an indispensable role in meeting user preferences by recommending relevant items. Collaborative Filtering (CF) is one of the dominant techniques used in recommender systems [19, 20, 37, 45]. Traditional CF-based methods mainly rely on the history of user-item interaction to generate recommendations. However, they are often impeded by data sparsity and cold start issues in real recommendation scenarios.

With the emergence of online social media, many E-commerce sites have become popular social platforms in which users can not only select items they love but also follow other users. According to the social influence theory [2, 3, 23], users' preferences are similar to or influenced by their social neighbors. Therefore, researchers propose using social network as another information stream to mitigate the lack of user-item interaction and enhance recommendation performance, also known as the social recommendation [27]. To tackle the social recommendation problem, a diverse plethora of social-aware models have been proposed, which utilizes different techniques to integrate social relations into recommendation, such as matrix factorization [16, 28], multi-layer perceptron [11] and graph neural networks [25, 44].

Notably, all of these social recommenders operate in Euclidean spaces. In a real-world scenario, user-item interaction and user-user social relations exhibit power-law structures. Also, social networks present a hierarchical structure [1, 10, 15]. Recent research shows that hyperbolic geometry enables embeddings with much smaller distortion when embedding data with the power-law distribution and hierarchical structure [6, 31]. This motivates us to consider

whether we can utilize hyperbolic geometry for boosting performance of social recommendation.

**Our approach.** In this work, we propose a novel social recommendation model, namely, **Hyperbolic Social Recommender (HSR)**. Specifically, HSR learn user and item representations in the hyperbolic space – or more precisely in a Poincaré ball. The key component of our framework is that we design a hyperbolic aggregator on the users' social neighbor sets to take full advantage of the social information. With the help of hyperbolic geometry, HSR can better model user-item interaction and user-user social relations to enhance the performance in social recommendation.

**Our contributions.** In summary, our main contributions in this paper are listed as follows:

- We propose a novel framework HSR, which utilizes hyperbolic geometry for social recommendation. To the best of our knowledge, this is the first study to make use of hyperbolic space for social recommendation task.
- Experimental results show that our proposed HSR not only outperforms its Euclidean counterpart but also boosts the performance over the state-of-the-art social recommenders in click-through rate prediction and top- $K$  recommendation, demonstrating the effectiveness of social recommendation in hyperbolic geometry.

The remainder of this paper is organized as follows. Section 2 discusses the relevant background that forms the basis of our work. In Section 3, we introduce the proposed method HSR. In Section 4, we conduct experiments on four real-world datasets and present the experimental results. In Section 5, we review work related to our methods, followed by a conclusion in Section 6.

## 2 BACKGROUND

In this section, we review the background of hyperbolic geometry and gyrovector space, which forms the basis of our method.

### 2.1 Hyperbolic Geometry

The hyperbolic space is uniquely defined as a complete and simply connected Riemannian manifold with constant negative curvature [22]. A key property of hyperbolic spaces is that they expand faster than Euclidean spaces. To describe the hyperbolic space, there are multiple commonly used models of hyperbolic geometry, such as the Poincaré model, hyperboloid model, and Klein model [47]. These models are all connected and can be converted into each other. In this paper, we work with the Poincaré ball model because it is well-suited for gradient-based optimization [31].

**Poincaré ball model.** Let  $\mathbb{D}^n = \{\mathbf{x} \in \mathbb{R}^n : \|\mathbf{x}\| < 1\}$  be the open  $n$ -dimensional unit ball, where  $\|\cdot\|$  denotes the Euclidean norm. The Poincaré ball model is the Riemannian manifold  $(\mathbb{D}^n, g^{\mathbb{D}})$ , which is defined by the manifold  $\mathbb{D}^n$  equipped with the Riemannian metric

\*Corresponding author.

tensor  $g_{\mathbf{x}}^{\mathbb{D}} = \lambda_{\mathbf{x}}^2 g^{\mathbb{B}}$ , where  $\lambda_{\mathbf{x}} = \frac{2}{1-\|\mathbf{x}\|^2}$ ;  $\mathbf{x} \in \mathbb{D}^n$ ; and  $g^{\mathbb{B}} = \mathbf{I}$  denotes the Euclidean metric tensor.

## 2.2 Gyrovector Spaces

The framework of gyrovector spaces provides vector operations for hyperbolic geometry [14]. We will make extensive use of these gyrovector operations to design our model. Specifically, these operations in gyrovector spaces are defined in an open  $n$ -dimensional ball  $\mathbb{D}_c^n = \{\mathbf{x} \in \mathbb{R}^n : c \|\mathbf{x}\|^2 < 1\}$  of radius  $\frac{1}{\sqrt{c}}$  ( $c \geq 0$ ). Some widely used vector operations of gyrovector spaces are defined as follows:

- *Möbius addition*: For  $\mathbf{x}, \mathbf{y} \in \mathbb{D}_c^n$ , the Möbius addition of  $\mathbf{x}$  and  $\mathbf{y}$  is defined as follows:

$$\mathbf{x} \oplus_c \mathbf{y} = \frac{(1 + 2c \langle \mathbf{x}, \mathbf{y} \rangle + c \|\mathbf{y}\|^2) \mathbf{x} + (1 - c \|\mathbf{x}\|^2) \mathbf{y}}{1 + 2c \langle \mathbf{x}, \mathbf{y} \rangle + c^2 \|\mathbf{x}\|^2 \|\mathbf{y}\|^2}. \quad (1)$$

In general, this operation is not commutative nor associative.

- *Möbius scalar multiplication*: For  $c > 0$ , the Möbius scalar multiplication of  $\mathbf{x} \in \mathbb{D}_c^n \setminus \{\mathbf{0}\}$  by  $r \in \mathbb{R}$  is defined as follows:

$$r \otimes_c \mathbf{x} = \frac{1}{\sqrt{c}} \tanh \left( r \tanh^{-1} \left( \sqrt{c} \|\mathbf{x}\| \right) \right) \frac{\mathbf{x}}{\|\mathbf{x}\|}, \quad (2)$$

and  $r \otimes_c \mathbf{0} = \mathbf{0}$ . This operation satisfies associativity:

- *Möbius matrix-vector multiplication*: For  $\mathbf{M} \in \mathbb{R}^{n' \times n}$  and  $\mathbf{x} \in \mathbb{D}_c^n$ , if  $\mathbf{M}\mathbf{x} \neq \mathbf{0}$ , the Möbius matrix-vector multiplication of  $\mathbf{M}$  and  $\mathbf{x}$  is defined as follows:

$$\mathbf{M} \otimes_c \mathbf{x} = \frac{1}{\sqrt{c}} \tanh \left( \frac{\|\mathbf{M}\mathbf{x}\|}{\|\mathbf{x}\|} \tanh^{-1} \left( \sqrt{c} \|\mathbf{x}\| \right) \right) \frac{\mathbf{M}\mathbf{x}}{\|\mathbf{M}\mathbf{x}\|}. \quad (3)$$

This operation satisfies associativity.

- *Möbius exponential map and logarithmic map*: For  $\mathbf{x} \in \mathbb{D}_c^n$ , it has a tangent space  $T_{\mathbf{x}}\mathbb{D}_c^n$  which is a local first-order approximation of the manifold  $\mathbb{D}_c^n$  around  $\mathbf{x}$ . The logarithmic map and the exponential map can move the representation between the two manifolds in a correct manner. For any  $\mathbf{x} \in \mathbb{D}_c^n$ , given  $\mathbf{v} \neq \mathbf{0}$  and  $\mathbf{y} \neq \mathbf{x}$ , the Möbius exponential map  $\exp_{\mathbf{x}}^c : T_{\mathbf{x}}\mathbb{D}_c^n \rightarrow \mathbb{D}_c^n$  and logarithmic map  $\log_{\mathbf{x}}^c : \mathbb{D}_c^n \rightarrow T_{\mathbf{x}}\mathbb{D}_c^n$  are defined as follows:

$$\exp_{\mathbf{x}}^c(\mathbf{v}) = \mathbf{x} \oplus_c \left( \tanh \left( \sqrt{c} \frac{\lambda_{\mathbf{x}}^c \|\mathbf{v}\|}{2} \right) \frac{\mathbf{v}}{\sqrt{c} \|\mathbf{v}\|} \right), \quad (4)$$

$$\log_{\mathbf{x}}^c(\mathbf{y}) = \frac{2}{\sqrt{c} \lambda_{\mathbf{x}}^c} \tanh^{-1} \left( \sqrt{c} \|\mathbf{x} \oplus_c \mathbf{y}\| \right) \frac{-\mathbf{x} \oplus_c \mathbf{y}}{\|\mathbf{x} \oplus_c \mathbf{y}\|}, \quad (5)$$

where  $\lambda_{\mathbf{x}}^c = \frac{2}{1-c\|\mathbf{x}\|^2}$  is the conformal factor of  $(\mathbb{D}_c^n, g^c)$ , where  $g^c$  is the generalized hyperbolic metric tensor.

- *Distance*: For  $\mathbf{x}, \mathbf{y} \in \mathbb{D}_c^n$ , the generalized distance between them in Gyrovector spaces are defined as follows:

$$d_c(\mathbf{x}, \mathbf{y}) = \frac{2}{\sqrt{c}} \tanh^{-1} \left( \sqrt{c} \|\mathbf{x} \oplus_c \mathbf{y}\| \right). \quad (6)$$

We will make use of these Möbius gyrovector space operations to design our recommendation framework.

## 3 METHODOLOGY

In this section, we first introduce the notations and formulate the problems. We then elaborate our proposed HSR method. Based on that, we introduce two strategies to further improve our model. Finally, we discuss the learning algorithm of our model.

### 3.1 Problem Definition

In a typical recommendation scenario, we suppose there are  $M$  users  $\mathcal{U} = \{u_1, u_2, \dots, u_M\}$  and  $N$  items  $\mathcal{V} = \{v_1, v_2, \dots, v_N\}$ . We define  $\mathbf{Y} \in \mathbb{R}^{M \times N}$  as the user-item historical interaction matrix whose element  $y_{ai} = 1$  if user  $a$  is interested in item  $i$  and zero otherwise. In addition to the interaction matrix  $\mathbf{Y}$ , we also have a social network  $\mathbf{S} \in \mathbb{R}^{N \times N}$  among users  $\mathcal{U}$ , where its element  $s_{ab} = 1$  if  $u_a$  trust  $u_b$  and zero otherwise.

The goal of our method HSR is to utilize the historical interaction and the social information to predict user's personalized interests in items. Specifically, given the user-item interaction matrix  $\mathbf{Y}$  as well as user social network  $\mathbf{S}$ , HSR aims to learn a prediction function  $\hat{y}_{ai} = \mathcal{F}(u_a, v_i | \Theta, \mathbf{Y}, \mathbf{S})$ , where  $\hat{y}_{ai}$  is the preference probability from user  $u_a$  to item  $v_i$  which she has never engaged before, and  $\Theta$  is the model parameters of function  $\mathcal{F}$ .

### 3.2 Model Formulation

We now present the our method HSR. There are three components in the model: the embedding layer, the aggregation layer, and the prediction layer. Details of each part are described in the following.

**3.2.1 Embedding Layer.** Such layer takes a user and an item as an input, and encodes them with dense low-dimensional embedding vectors. Specifically, given one hot representations of target user  $u_a$  and target item  $v_i$ , the embedding layer outputs their embeddings  $\mathbf{u}_a$  and  $\mathbf{v}_i$ , respectively. We will learn user and item embedding vectors in the hyperbolic space  $\mathbb{D}_c^d$ .

**3.2.2 Aggregation Layer.** Due to the social influence theories [2, 3, 23], user's preference will be indirectly influenced by her social friends. We should utilize social information for better user embedding modeling. Specially, we devise a social aggregator on the users' trusted neighbors to refine users' embeddings in the hyperbolic space, formulating the aggregation process with two major operations: *neighbor aggregation* and *feature update*.

The neighborhood aggregation stage learns the representation of a neighborhood by transforming and aggregating the feature information from the neighborhood. The center-neighbor combination stage learns the representation of the central node by combining the representation of the neighborhood with the features of the central node.

*Neighbor aggregation.* Given a user, neighbor aggregation first aggregates the given user's neighbor representations into a single embeddings, and then combines the representation of the neighborhood with the feature of the given user. We can directly utilize Möbius addition to achieve the neighbor aggregation. Denote  $\mathcal{N}(a)$  as user  $u_a$ 's social neighbor set, the neighbor aggregation for user  $u_a$  is defined as follows:

$$\begin{aligned} \mathbf{u}_{\mathcal{N}(a)} &= \sum_{b \in \mathcal{N}(a)}^{\oplus_c} \mathbf{u}_b \\ \mathbf{u}_a^{AGG} &= \mathbf{u}_a \oplus_c \left( \gamma \otimes_c \mathbf{u}_{\mathcal{N}(a)} \right) \end{aligned} \quad (7)$$

where  $\sum^{\oplus_c}$  is the accumulation of Möbius addition, and  $\gamma$  is a coefficient which controls the social influence.

*Feature update.* In this stage, we further update the aggregated representations to obtain sufficient representation power as:

$$\mathbf{u}_a^* = \sigma \left( \mathbf{M} \otimes_c \mathbf{u}_a^{AGG} \right) \quad (8)$$

where  $\mathbf{M} \in \mathbb{R}^{d \times d}$  is the trainable matrix, and  $\sigma$  is the nonlinear activation function defined as LeakyReLU [29].

*High-order aggregation.* Through a single aggregation layer, user representation is dependent on itself as well as the direct social neighbors. We can further stack more layers to obtain high-order information from multi-hop neighbors of users. More formally, in the  $\ell$ -th layer, for user  $u_a$ , her representation is defined as:

$$\mathbf{u}_a^\ell = \sigma \left( \mathbf{M}^\ell \otimes_c \left( \mathbf{u}_a^{\ell-1} \oplus_c \left( \gamma \otimes_c \sum_{b \in N(a)}^{\oplus_c} \mathbf{u}_b^{\ell-1} \right) \right) \right) \quad (9)$$

where  $\mathbf{M}^\ell$  is the trainable matrix in the  $\ell$ -th layer and  $\mathbf{u}_a^1 = \mathbf{u}_a^*$ .

**3.2.3 Prediction Layer.** After the  $L$  aggregation layer, for target user  $u_a$ , we obtain her final representation  $\mathbf{u}_a^L$ . We then feed user representation  $\mathbf{u}_a^L$  and target item representation  $\mathbf{v}_i$  into a function  $p$  for predicting the probability of  $u_a$  engaging  $v_i$ :  $\hat{y}_{ai} = p(\mathbf{u}_a^L, \mathbf{v}_i)$ . Here we implement the prediction function  $f$  as the Fermi-Dirac decoder [22, 31], a generalization of sigmoid function, to compute probability scores between  $u_a$  and  $v_i$ , as follows:

$$\hat{y}_{ai} = \frac{1}{e^{(d_c(\mathbf{u}_a^L, \mathbf{v}_i) - r)/t} + 1} \quad (10)$$

where  $r$  and  $t$  are hyper-parameters.

### 3.3 Model Improvement

In this subsection, we further improve our method from two aspects.

**3.3.1 Acceleration strategy.** Since Möbius addition operation is not commutative nor associative [14], we have to calculate the accumulation of Möbius addition by order in Equation (7), as follows:

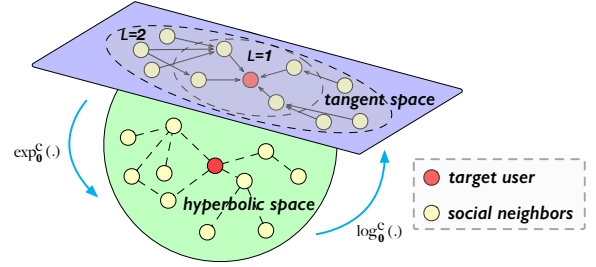
$$\begin{aligned} \mathbf{u}_a^{AGG} &= \mathbf{u}_a \oplus_c \left( \gamma \otimes_c \sum_{b \in N(a)}^{\oplus_c} \mathbf{u}_b \right) \\ &= \mathbf{u}_a \oplus_c \left( \gamma \otimes_c \left( (\mathbf{u}_{b_1} \oplus_c \mathbf{u}_{b_2}) \oplus_c \mathbf{u}_{b_3} \oplus_c \dots \right) \right) \end{aligned} \quad (11)$$

As is known to all, there exist some active users who have many social behaviors in social network, so the calculation in Equation (11) is seriously time-consuming, which will affect the efficiency of our method HSR. Therefore, it is necessary to devise a new way to calculate the aggregation.

We resort to Möbius logarithmic map and exponential map, as illustrated in Figure 1. Specifically, we first utilizes the logarithmic map to project user representations into a tangent space, then perform the accumulation operation to aggregate the user representations in the tangent space, and finally project aggregated representations back to the hyperbolic space with the exponential map. The process is formulated as:

$$\mathbf{u}_a^{AGG} = \exp_0^c \left( \log_0^c(\mathbf{u}_a) + \gamma \cdot \sum_{b \in N(a)} \log_0^c(\mathbf{u}_b) \right) \quad (12)$$

Different from Equation (7), we can calculate the results in a parallel way in Equation (12) because the accumulation operation in the tangent space is commutative and associative, which enables our model more efficient.



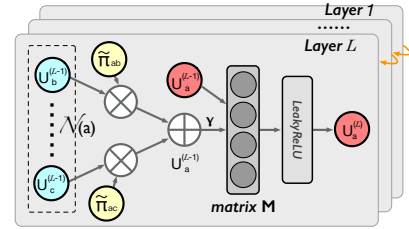
**Figure 1: Illustration of the acceleration strategy of second-order embedding propagation for the target user.**

**3.3.2 Attention mechanism.** For a user, social influence strength of her friends should be different and dynamic. Therefore, we introduce an attention mechanism, which assigns non-uniform weights to users' social neighbors. We denote  $\pi_{ab}$  as the social influence strength from user  $u_a$ 's social neighbor  $u_b$  to  $u_a$ , which can be compute as follows:

$$\pi_{ab} = (\log_0^c(\mathbf{u}_a) \odot \log_0^c(\mathbf{u}_b))^T \tanh(\mathbf{w}^T [\log_0^c(\mathbf{u}_b), \log_0^c(\mathbf{v}_i)]) \quad (13)$$

where  $[\cdot, \cdot]$  and  $\odot$  mean concatenation operation and element-wise product between two vectors in the tangent space.  $\mathbf{w} \in \mathbb{R}^{2d \times d}$  is the trainable weighted matrix of the attention mechanism. We also employ tanh as the nonlinear activation function.

We consider target user, target item and target user's social neighbor to design the attention mechanism. The first term calculates the compatibility between user  $u_a$  and her social neighbor  $u_b$ , and the second term computes the opinions of the neighbor  $u_b$  on the target item  $v_i$ . Here we simply employ inner product on the two terms, one can design a more sophisticated attention mechanism.



**Figure 2: Illustration of the aggregation operation.**

By integrating Equation (12) and (13), the final aggregation rule (as illustrated in Figure 2) for each user  $u_a$  is calculated as follows:

$$\mathbf{u}_a^\ell = \sigma \left( \mathbf{M}^\ell \otimes_c \exp_0^c \left( \log_0^c(\mathbf{u}_a^{\ell-1}) + \gamma \cdot \sum_{b \in N(a)} \tilde{\pi}_{ab} \cdot \log_0^c(\mathbf{u}_b^{\ell-1}) \right) \right) \quad (14)$$

where  $\tilde{\pi}_{ab}$  is the normalized attention weight which can be calculated by softmax function with temperature parameter  $\tau$ :

$$\tilde{\pi}_{ab} = \frac{\exp(\pi_{ab}/\tau)}{\sum_{b' \in N(a)} \exp(\pi_{ab'}/\tau)} \quad (15)$$

### 3.4 Model Optimization

**3.4.1 Objective Function.** The complete objective function consists of two parts: recommendation loss and social relation loss.

*Recommendation Loss.* For the recommendation task, the loss function of recommendation is defined as:

$$\mathcal{L}_r = - \sum_{(u,i,j) \in \mathcal{D}_r} (y_{ui} \log(\hat{y}_{ui}) + (1 - y_{uj}) \log(1 - \hat{y}_{uj})) \quad (16)$$

where  $\mathcal{D}_r$  is the set of training triplets. We denote  $\mathcal{V}_u$  is the item set which user  $u$  has interacted with, and  $\mathcal{D}_r$  can be defined as:

$$\mathcal{D}_r = \{(u, i, j) \mid u \in \mathcal{U} \wedge i \in \mathcal{V}_u \wedge j \in \mathcal{V} \setminus \mathcal{V}_u\} \quad (17)$$

*Social Relation Loss.* To model user social relations in social network, we devise the social relation loss. We first define the score (similarity) function for a user pair  $(u_a, u_b)$  according to their distance in the hyperbolic space, as follows:

$$\text{score}(u_a, u_b) = -d_c(\mathbf{u}_a, \mathbf{u}_b) \quad (18)$$

We want the observed trusted user pairs to have higher scores than unobserved user pairs in social network. We utilize the Bayesian personalized ranking (BPR) method to define  $\mathcal{L}_s$ , as follows:

$$\mathcal{L}_s = - \sum_{(u,p,q) \in \mathcal{D}_s} \log \sigma(\text{score}(u, p) - \text{score}(u, q)) \quad (19)$$

where  $\sigma$  is the sigmoid function  $\sigma(x) = 1/(1 + e^{-x})$ , and  $\mathcal{D}_s$  is defined as follow:

$$\mathcal{D}_s = \{(u, p, q) \mid u \in \mathcal{U} \wedge p \in \mathcal{N}_{(u)} \wedge q \in \mathcal{U} \setminus \mathcal{N}_{(u)}\} \quad (20)$$

*Multi-Task Learning.* We integrate the recommendation loss  $\mathcal{L}_r$  and the social relation loss  $\mathcal{L}_s$  into an end-to-end fashion through a multi-task learning framework, and utilize  $\lambda$  to balance two terms. The complete loss function of HSR is defined as follows:

$$\min_{\Theta} \mathcal{L} = \mathcal{L}_r + \lambda \mathcal{L}_s \quad (21)$$

where  $\Theta$  is the total parameter space, including user embeddings  $\{\mathbf{u}_i\}_{i=1}^{|\mathcal{U}|}$ , item embeddings  $\{\mathbf{v}_i\}_{i=1}^{|\mathcal{V}|}$ , and weight parameters of the networks  $\{\mathbf{M}^i, \mathbf{w}^i\}_{i=1}^L$ .

**3.4.2 Gradient Conversion.** Since the Poincaré Ball has a Riemannian manifold structure, we utilize Riemannian stochastic gradient descent (RSGD) to optimize our model [4]. As similar to [31], the parameter updates are of the following form:

$$\theta_{t+1} = \mathfrak{R}_{\theta_t}(-\eta_t \nabla_R \mathcal{L}(\theta_t)), \quad (22)$$

where  $\mathfrak{R}_{\theta_t}$  denotes a retraction onto  $\mathbb{D}$  at  $\theta$  and  $\eta_t$  denotes the learning rate at time  $t$ . The Riemannian gradient  $\nabla_R$  can be computed by rescaling the Euclidean gradient  $\nabla_E$  with the inverse of the Poincaré ball metric tensor as  $\nabla_R = \frac{(1 - \|\theta_t\|^2)^2}{4} \nabla_E$ .

**3.4.3 Learning Algorithm.** The training process of the HSR is summarized in Algorithm 1.

## 4 EXPERIMENTS

### 4.1 Experiment Setup

In this subsection, we introduce the datasets, baselines, evaluation protocols, and the choice of hyper-parameters.

#### Algorithm 1 Training algorithm of HSR

**Input:** Interaction matrix  $\mathbf{Y}$ ; social network  $\mathbf{S}$

**Output:** Prediction function  $\mathcal{F}(u, v | \Theta, \mathbf{Y}, \mathbf{S})$

```

1: Initialize all parameters in  $\Theta$ 
2: repeat
3:   Draw a mini-batch of  $(u, i, j)$  from  $\mathcal{D}_r$ 
4:   Draw a mini-batch of  $(u, p, q)$  from  $\mathcal{D}_s$ 
5:   Calculate  $\mathcal{L}_r$  according to Equation (7) to (17)
6:   Calculate  $\mathcal{L}_s$  according to Equation (18) to (20)
7:   Calculate  $\mathcal{L} \leftarrow \mathcal{L}_r + \lambda \mathcal{L}_s$ 
8:   Update parameters of  $\mathcal{F}$  by using RSGD to optimize  $\mathcal{L}$ 
9: until  $\mathcal{L}$  converges or is sufficiently small
10: return  $\mathcal{F}(u, v | \Theta, \mathbf{Y}, \mathbf{S})$ 

```

**4.1.1 Datasets.** We experimented with four datasets: Ciao<sup>1</sup>, Yelp<sup>2</sup>, Epinion<sup>3</sup>, and Douban<sup>4</sup>. Each dataset contains users' ratings to the items and the social connections between users. In the data preprocessing step, we transform the ratings into implicit feedback (denoted by "1") indicating that the user has rated the item positively. Then, for each user, we sample the same amount of negative samples (denoted by "0") as their positive samples from unwatched items. Also, we filtered out users with no links in social networks. The statistics of the datasets are summarized in Table 1.

**Table 1: Statistical details of the four datasets. "interactions" means user-item historical records, and "relations" denotes user-friend connections in social network.**

dataset	# users	# items	# interactions	# relations
Ciao	7,210	11,211	147,590	111,781
Yelp	10,580	13,870	342,204	158,590
Epinion	19,600	23,585	443,640	351,485
Douban	12,748	22,347	1,570,544	169,150

**4.1.2 Comparison Methods.** To verify the performance of our proposed method HSR, we compared it with the following state-of-art social recommendation methods. The characteristics of the comparison methods are listed as follows:

- **FM** is a feature enhanced factorization model. Here we utilize the social information as additional input features. Specifically, we concatenate the user embedding, item embedding and the average embeddings of user social neighbors as the inputs [33].
- **DeepFM** is also a feature enhanced factorization model, which combines factorization machines and deep neural networks. We provide the same inputs as FM for DeepFM [17].
- **SoReg** models users' social relation as regularization terms to constrain the matrix factorization framework [28].
- **TrustSVD** extends SVD++ [21] framework by modelling both user-user social relations and user-item interaction [16].
- **DeepSoR** combines deep neural networks to learn users' preferences from social networks and integrate users' preferences into PMF [36] framework for recommendation [11].

<sup>1</sup>Ciao: <http://www.cse.msu.edu/~tangjili/index.html>

<sup>2</sup>Yelp: <http://www.yelp.com/>

<sup>3</sup>Epinion: <http://alchemy.cs.washington.edu/data/epinions/>

<sup>4</sup>Douban: <http://book.douban.com>

**Table 2: The results of AUC and Accuracy in CTR prediction on four datasets. \*\* denotes the best values among all methods, and \* denotes the best values among all competitors.**

Method	Ciao		Yelp		Epinion		Douban	
	AUC	ACC	AUC	ACC	AUC	ACC	AUC	ACC
FM	0.7538	0.6828	0.8153	0.7574	0.8048	0.7378	0.8304	0.7595
DeepFM	<b>0.7706*</b>	0.6894	0.8302	0.7640	0.8158	0.7380	0.8347	0.7580
SoReg	0.7346	0.6663	0.8194	0.7546	0.7968	0.7269	0.8262	0.7481
TrustSVD	0.7539	0.6765	0.8263	0.7567	0.8072	0.7404	0.8288	0.7607
DeepSoR	0.7646	0.6901	0.8308	0.7645	0.8171	0.7394	0.8303	0.7555
DiffNet	0.7704	<b>0.6924*</b>	<b>0.8385*</b>	<b>0.7686*</b>	0.8162	0.7426	0.8295	0.7581
HOSR	0.7682	0.6895	0.8348	0.7615	<b>0.8219*</b>	<b>0.7445*</b>	<b>0.8353*</b>	<b>0.7629*</b>
HSR	<b>0.7889**</b>	<b>0.7183**</b>	<b>0.8552**</b>	<b>0.7919**</b>	<b>0.8295**</b>	<b>0.7580**</b>	<b>0.8477**</b>	<b>0.7743**</b>
ESR	0.7645	0.6881	0.8330	0.7682	0.8063	0.7405	0.8307	0.7581

- **DiffNet** is a graph neural network based social recommender, which designs a layer-wise influence propagation structure for better user embedding modeling [44].
- **HOSR** is a another graph neural network based social recommender, which propagates user embeddings along the social network and designs a attention mechanism to study the importance of different neighbor orders [25].
- **ESR** is the Euclidean counterpart of HSR, which replaces Möbius addition, Möbius matrix-vector multiplication, Gyrovector space distance with Euclidean addition, Euclidean matrix multiplication, Euclidean distance, and remove Möbius logarithmic map and exponential map.
- **HSR** is our complete model.

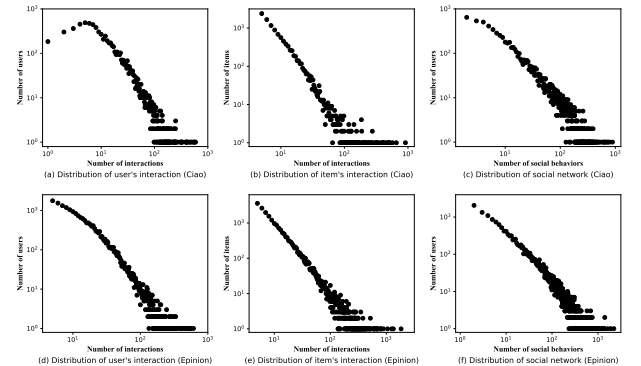
**4.1.3 Parameter Settings.** We implemented our methods with Pytorch which is a Python library for deep learning. For each dataset, we randomly split it into training, validation, and test sets following 7 : 1 : 2. The hyper-parameters were tuned on the validation set by a grid search. Specifically, the learning rate  $\eta$  is tuned among  $[10^{-4}, 5 \times 10^{-4}, 10^{-3}, 5 \times 10^{-3}]$ ; the embedding size  $d$  is searched in  $[8, 16, 32, 64, 128]$ ; the balancing factor  $\lambda$  is chosen from  $[10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}]$ ; and the temperature  $\tau$  is tuned amongst  $[0.01, 0.05, 0.1, 0.5]$ . For the aggregation layer size  $L$ , we set  $L = 1$  for all datasets and find them sufficiently good. We find using 2 or more layers can slightly increase the performance but take much longer training time. In addition, we set batch size  $b = 1024$ , curvature  $c = 1$ , social coefficient  $\gamma = 1$ , and Fermi-Dirac decoder parameters  $r = 2$ ,  $t = 1$ . We will further study the impact of key hyper-parameters in the following subsection. The best settings for hyper-parameters in all baselines are reached by either empirical study or following their original papers.

**4.1.4 Evaluation Protocols.** We evaluate our methods in two scenarios: (i) in click-through rate (CTR) prediction, we adopt two metrics AUC (area under the curve) and Accuracy, which are widely utilized in binary classification problems; and (ii) in top-K recommendation, we use the model obtained in CTR prediction to generate top-K items. Since it is time-consuming to rank all items for each user in the evaluation procedure, to reduce the computational cost, following the strategy in [18, 44], for each user, we randomly sample 500 unrated items at each time and combine them with the positive items in the ranking process. We use *Precision@K* and *Recall@K*

to evaluate the recommended sets. We repeated each experiment 5 times and reported the average performance.

## 4.2 Empirical Study

Researches show that data with a power-law structure can be naturally modeled in the hyperbolic space [22, 31, 43]. Therefore, we conduct an empirical study to check whether the power-law distribution also exists in the user-item interaction and user-user social relations. For the user-item interaction relation, we present the distribution of the number of interactions for users and items, respectively. For the user-user social relation, we present the distribution of number of social behaviors for users. We show the distributions of these two relations on the Ciao and Epinion datasets in Figure 3. We observed that these distributions show the power-law distribution: (i) for the user-item interaction relation, a majority of users/items have very few interactions and a few users/items have a huge number of interactions; and (ii) for the user-user social relation, a majority of users have very few social behaviors and a few users have a huge number of social behaviors. The above findings empirically demonstrate user-item interaction and user-user social relations exhibit power-law structure, thus we believe that using hyperbolic geometry might be suitable for the social recommendation.



**Figure 3: Distributions of user-item interaction and user-user social relations on Ciao (in top row) and Epinion datasets (in bottom row).**

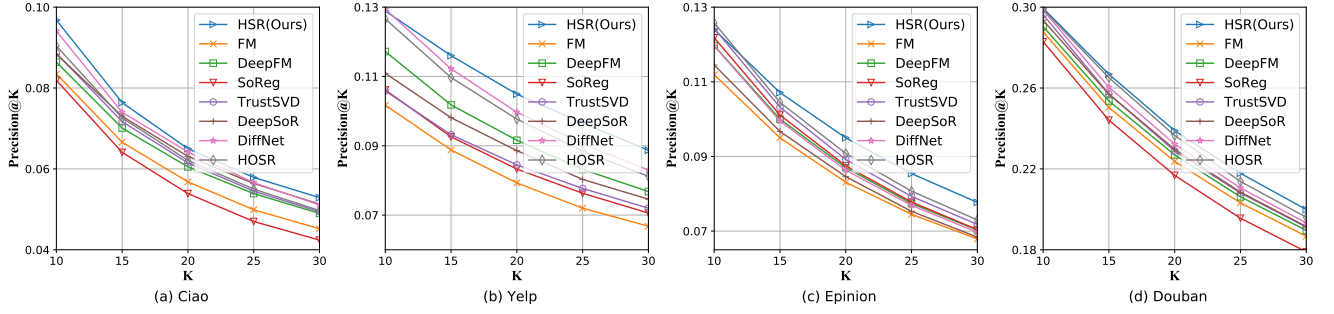


Figure 4: The results of  $Precision@K$  in top- $K$  recommendation on four datasets

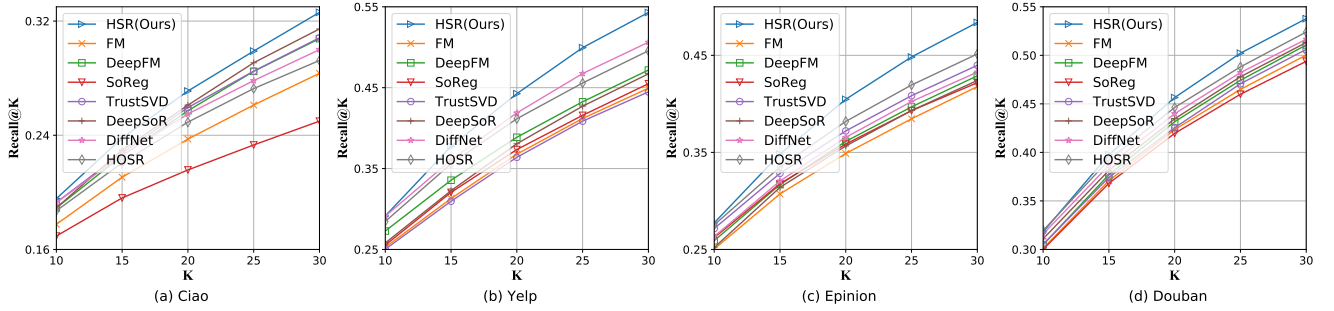


Figure 5: The results of  $Recall@K$  in top- $K$  recommendation on four datasets

### 4.3 Performance Comparison

Table 2 and Figures 4, 5 show the performance of all compared methods in CTR prediction and top- $K$  recommendation, respectively (ESR are not plotted in Figure 4, 5 for clarity). From the results, we have the following main observations:

(i) Deep learning based models (i.e., DeepFM, DeepSoR, DiffNet and HOSR) generally outperform shallow representation methods (i.e., FM, SoReg and TrustSVD), which indicates the effectiveness of applying neural components for social recommendation.

(ii) Among baselines, graph neural network based social recommenders DiffNet and HOSR achieve strongly competitive performance. Such improvement verifies graph neural networks are powerful in representation learning for graph data, since it integrates the users' social information as well as users's topological structure in social network.

(iii) Intuitively, HSR has made great improvements over state-of-the-art baselines in both recommendation scenarios. For CTR prediction task, our method HSR consistently yields the best performance on four datasets. For example, HSR improves over the strongest baselines *w.r.t.*  $Accuracy$  by 3.73%, 3.03%, 1.81%, and 1.49% in Ciao, Yelp, Epinion and Douban datasets, respectively. In top- $K$  recommendation, HSR achieves 3.79%, 5.68%, 5.97%, and 2.46% performance improvement against the strongest baseline *w.r.t.*  $Recall@20$  in Ciao, Yelp, Epinion and Douban datasets, respectively. Considering the Euclidean counterpart of HSR, HSR achieves better scores than ESR. This indicates the effectiveness of social recommendation in the hyperbolic space.

### 4.4 Handling Data Sparsity Issue

The data sparsity problem is a great challenge for most recommender systems. To investigate the effect of data sparsity, we bin the test users into four groups with different sparsity levels based on the number of observed ratings in the training data, meanwhile keep each group including a similar number of interactions. For example, [11,26] in the Ciao dataset means for each user in this group has at least 11 interaction records and less than 26 interaction records. Figure 6 shows the  $Accuracy$  results on different user groups with different models on Ciao and Epinion datasets. From the results, we observe that HSR consistently outperforms the other methods including the state-of-the-art social enhanced methods like DiffNet and HOSR, which verifies that our method is able to maintain a decent performance in different sparse scenarios.

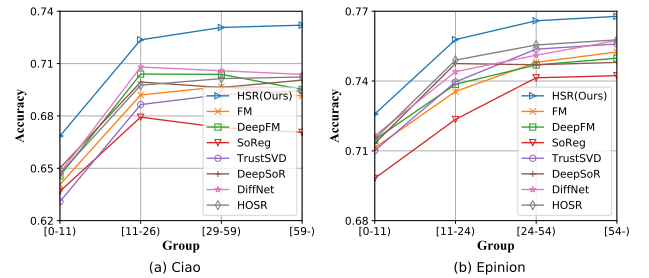


Figure 6: Performance comparison over the sparsity distribution of user groups on Ciao and Epinion datasets.



#### 4.5 Parameter Sensitivity

We explore the impact of four hyper-parameters: embedding size  $d$ , balancing factor  $\lambda$ , and temperature  $\tau$ . The results on Ciao and Yelp datasets are plotted in Figure 7. We have the following observations: (i) A proper embedding size  $d$  is needed. If it is too small, the model lacks expressiveness, while a too large  $d$  increases the complexity of the recommendation framework and may overfit the datasets. In addition, we observe that HSR always significantly outperforms ESR regardless of the embedding size, especially in a low-dimensional space, which shows that utilizing hyperbolic geometry can effectively learn high-quality representations for social recommendation. (ii) For balancing factor  $\lambda$ , we find that when  $\lambda = 10^{-2}$  is good enough on Ciao and Yelp datasets because a larger  $\lambda$  will let the model focus on modeling social relation task. (iii) We find the empirically optimal temperature  $\tau$  to be 0.1 on Ciao and Yelp datasets. The performance increases when the temperature is tuned from 0 to the optimal value and then drops down afterwards, which indicates that proper value of  $\tau$  can distill useful information for social neighbor aggregation.

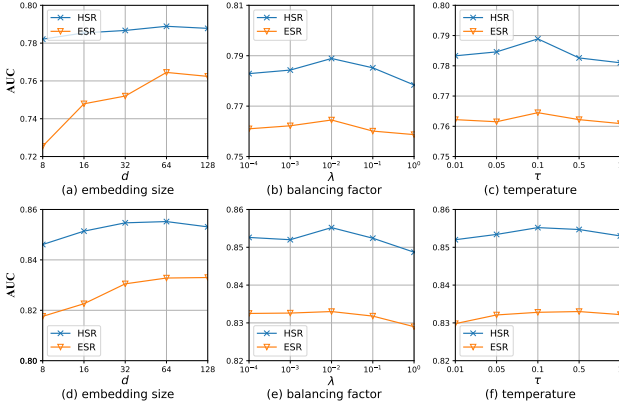


Figure 7: Parameter sensitivity on Ciao and Yelp datasets.

#### 4.6 Ablation Study

To explore the effect of our neighbor attention mechanism and social relation modeling task, we conducted experiments with the two variants of HSR and ESR: (1) HSR-A and ESR-A (performing a mean operation in Equation (12), and (2) HSR-S and ESR-S (only modeling the recommendation task in Equation (21)). Table 3 shows the AUC results on four datasets. From the results, we find that removing any components will decrease recommendation performance of our models. For example, HSR-A performs worse than the complete model HSR, which shows that considering different influences of social neighbors in aggregation operation is necessary to model user preference. HSR also achieves better scores than HSR-S. This validates that explicitly modeling social relations is helpful for improving the model performance.

#### 4.7 Case Study

**4.7.1 Embedding Analysis.** Social networks often present a hierarchical structure [1, 10, 15]. In this case study, we evaluate whether

Table 3: Effect of the attention mechanism and social relation modeling on four datasets.

Dataset	Ciao	Yelp	Epinion	Douban
HSR	0.7889	0.8552	0.8295	0.8477
HSR-A	0.7773	0.8490	0.8234	0.8431
HSR-S	0.7784	0.8465	0.8247	0.8426
ESR	0.7645	0.8330	0.8063	0.8307
ESR-A	0.7593	0.8278	0.8002	0.8267
ESR-S	0.7581	0.8295	0.8036	0.8253

our models can reflect hierarchical structures of social behaviors. In general, the distances between embeddings and the origin can reflect the latent hierarchy of graphs [31, 43]. We utilize Gyrovectorspace distance and Euclidean distance to calculate the distance to the origin for HSR and ESR, respectively. We bin the nodes in the user-user social graph into four groups according to their distances to the origin (from the near to the distant), meanwhile, keep each group including a similar number of nodes. For example, nodes in group 1 have the nearest distances to the origin while nodes in group 4 have the furthest distances to the origin. To evaluate the nodes' activity in the social graph, we compute the average number of nodes' social behaviors in each group. Figure 8 shows the results on Epinion and Douban datasets. From the results, we can see that the average number of interaction behaviors decreases from group 1 to group 4. This result indicates that hierarchy of social behaviors can be modeled by our methods HSR and ESR. Compared with ESR, we find that HSR more clearly reflects the hierarchical structure, which indicates that hyperbolic space is more suitable than Euclidean space to embed data with the hierarchical structure.

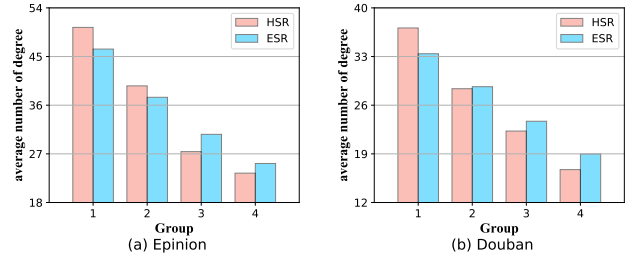
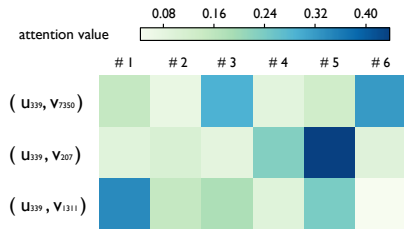


Figure 8: Analysis of hierarchical structure in social networks on Epinion and Douban datasets.

**Attention Analysis.** Benefiting from the attention mechanism, we can visualize the attention weights placed on the social neighbors for users, which reflects how the model learned. We randomly selected one user  $u_{339}$  who has six friends from Yelp dataset, and three relevant items  $v_{7350}$ ,  $v_{207}$ ,  $v_{1311}$  (from the test set). Figure 9 shows the attention weights of the user  $u_{339}$ 's social neighbors for the three user-item pairs. For convenience, we label neighbor ID starting from 1, which may not necessarily reflect the true ID from the dataset. From the heatmap, we have the following findings: (i) Not all neighbors have the same contribution when generating recommendations. For instance, for the user-item pair ( $u_{339}$ ,  $v_{7350}$ ), the attention weights of user  $u_{339}$ 's neighbor # 3 and # 6 are relatively high. The reason may be that neighbor # 3 and # 6 have

rated item  $v_{7350}$  in the training set. Therefore, neighbor # 3 and # 6 will provide more useful information when making recommendations. (ii) For different items, the attention distributions of the neighbors are different, which reflects the attention mechanism that can adaptively measure the influence strength of neighbors.



**Figure 9: Attention heatmap for the neighbors of three user-item pairs from Yelp dataset.**

## 5 RELATED WORK

In this section, we provide a brief overview of two areas that are highly relevant to our work.

*Social Recommendation.* With the prevalence of online social media, social relations have been widely studied and exploited to boost the recommendation performance. The most common approach in social recommendation is to design loss terms for social influence and integrate both social loss and recommendation loss into a unified loss function for jointly optimizing [24, 28, 39]. For instance, SoReg assumes that social neighbors share similar feature representations and devise regularization terms to smooth connected users’ embeddings [28]; CSR models the characteristics of social relations and designs a characterized regularization term to improve SoReg [24]. In addition to the regularization-based social recommenders, a more explicit and straightforward way is to explicitly models social relations in the predictive model [7, 16, 26]. For example, TrustSVD extended SVD++ [21] by incorporating each user’s social neighbors’ preferences as the auxiliary feedbacks [16]. Inspired by the immense success of deep learning, some recent proposed social recommenders utilize neural components to enhance recommendation performance [8, 9, 11, 25, 44]. For instance, DeepSoR combines a multi-layer perceptron to learn latent preferences of users from social networks with probabilistic matrix factorization [11]; SAMN considers both aspect-level and friend-level differences and utilizes memory network and attention mechanisms for social recommendation [8]; DiffNet stacks more graph convolutional layers and propagate users embeddings along the social network to capture high-order neighbor information [44]. Different from above-mentioned social recommenders based on Euclidean representation methods, we propose a hyperbolic representation model HSR for social recommendation, which utilizes hyperbolic geometry to learn high-quality user and item representations in the non-Euclidean space.

*Hyperbolic Representation Learning.* In recent years, representation learning in hyperbolic spaces has attracted an increasing amount of attention. Specifically, [31] embedded hierarchical data

into the Poincaré ball, showing that hyperbolic embeddings can outperform Euclidean embeddings in terms of both representation capacity and generalization ability. [32] focused on learning embeddings in the Lorentz model and showed that the Lorentz model of hyperbolic geometry leads to substantially improved embeddings. [13] extended Poincaré embeddings to directed acyclic graphs by utilizing hyperbolic entailment cones. [35] analyzed representation trade-offs for hyperbolic embeddings and developed and proposed a novel combinatorial algorithm for embedding learning in hyperbolic space. Besides, researchers began to combining hyperbolic embedding with deep learning. [14] introduced hyperbolic neural networks which defined core neural network operations in hyperbolic space, such as Möbius addition, Möbius scalar multiplication, exponential and logarithmic maps. After that, hyperbolic analogues of other algorithms have been proposed, such as Poincaré Glove [40] and hyperbolic attention networks [46].

There are some recent works using hyperbolic representation learning for recommendation [5, 12, 30, 38, 41, 42]. For instance, HyperBPR learns user and item hyperbolic representations and utilizes BPR [34] framework for collaborative filtering [42]. HME is designed to solve next-Point-of-Interest (POI) recommendation task, which projects the check-in data into a hyperbolic space for representation learning [12]. Different from the above literature, we propose HSR which utilizes hyperbolic geometry for social recommendation. To our knowledge, HSR is the first work that uses the hyperbolic space for social recommendation tasks.

## 6 CONCLUSION AND FUTURE WORK

In this work, we develop a novel method called HSR which leverages hyperbolic geometry for social recommendation. The key component of HSR is that we design a hyperbolic aggregator on the users’ social neighbor sets to take full advantage of the social information. We further introduce acceleration strategy and attention mechanism to improve our method HSR. Additionally, we also devise the social relation modeling task to make the user embeddings be reflective of the relational structure in social network, and jointly train it with recommendation task. To the best of our knowledge, HSR is the first model to explore the hyperbolic space in social recommendation. We conduct extensive experiments on four real-world datasets. The results demonstrate (i) the superiority of HSR compared to strong baselines, and (ii) the effectiveness of social recommendation in hyperbolic geometry.

For future work, we will (i) extend our model for temporal social recommendation to consider users’ dynamic preferences, and (ii) try to generate recommendation explanations for comprehending the user behaviors and item attributes.



## REFERENCES

- [1] Clauset Aaron, Cristopher Moore, and Mark EJ Newman. 2008. Hierarchical structure and the prediction of missing links in networks. *Nature* 453.7191 (2008), 98–101.
- [2] Aris Anagnostopoulos, Ravi Kumar, and Mohammad Mahdian. 2008. Influence and correlation in social networks. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.
- [3] Robert M Bond, Christopher J Fariss, Jason J Jones, Adam DI Kramer, Cameron Marlow, Jaime E Settle, and James H Fowler. 2012. A 61-million-person experiment in social influence and political mobilization. (2012), 295–298.
- [4] Silvere Bonnabel. 2013. Stochastic Gradient Descent on Riemannian Manifolds. *IEEE Trans. Automat. Control* 58, 9 (2013), 2217–2229.
- [5] Benjamin Paul Chamberlain, Stephen R. Hardwick, David R. Wardrope, Fabon Dzogang, Fabio Daolio, and Saúl Vargas. 2019. Scalable Hyperbolic Recommender Systems. In *CoRR abs/1902.08648*.
- [6] Ines Chami, Zitao Ying, Christopher Ré, and Jure Leskovec. 2019. Hyperbolic Graph Convolutional Neural Networks. In *Advances in Neural Information Processing Systems* 32. 4869–4880.
- [7] Allison June-Barlow Chaney, David M. Blei, and Tina Eliassi-Rad. 2015. A Probabilistic Model for Using Social Networks in Personalized Item Recommendation. In *Proceedings of the 9th ACM Conference on Recommender Systems*. 43–50.
- [8] Chong Chen, Min Zhang, Yiqun Liu, and Shaoping Ma. 2019. Social Attentional Memory Network: Modeling Aspect- and Friend-Level Differences in Recommendation. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*. 177–185.
- [9] Chong Chen, Min Zhang, Chenyang Wang, Weizhi Ma, Minming Li, Yiqun Liu, and Shaoping Ma. 2019. An Efficient Adaptive Transfer Neural Network for Social-aware Recommendation. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*. 225–234.
- [10] Fengjiao Chen and Kan Li. 2015. Detecting hierarchical structure of community members in social networks. *Knowledge Based Systems* 87 (2015), 3–15.
- [11] Wenqi Fan, Qing Li, and Min Cheng. 2018. Deep Modeling of Social Relations for Recommendation. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence*. 8075–8076.
- [12] Shanshan Feng, Lucas Vinh Tran, Gao Cong, Lisi Chen, Jing Li, and Fan Li. 2020. HME: A Hyperbolic Metric Embedding Approach for Next-POI Recommendation. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval*. 1429–1438.
- [13] Octavian-Eugen Ganea, Gary Bécigneul, and Thomas Hofmann. 2018. Hyperbolic Entailment Cones for Learning Hierarchical Embeddings. In *Proceedings of the 35th International Conference on Machine Learning*. 1632–1641.
- [14] Octavian-Eugen Ganea, Gary Bécigneul, and Thomas Hofmann. 2018. Hyperbolic Neural Networks. In *Annual Conference on Neural Information Processing Systems 2018*. 5350–5360.
- [15] Frédéric Gilbert, Paolo Simonetto, Faraz Zaidi, Fabien Jourdan, and Romain Bourqui. 2011. Communities and hierarchical structures in dynamic social networks: analysis and visualization. *Social Network Analysis and Mining* 1, 2 (2011), 83–95.
- [16] Guibing Guo, Jie Zhang, and Neil Yorke-Smith. 2015. TrustSVD: Collaborative Filtering with Both the Explicit and Implicit Influence of User Trust and of Item Ratings. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*. 123–129.
- [17] Huifeng Guo, Ruiming Tang, Yunming Ye, Zhenguo Li, and Xiuqiang He. 2017. DeepFM: A Factorization-Machine based Neural Network for CTR Prediction. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*. 1725–1731.
- [18] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, and Tat-Seng Chua Xia Hu. 2017. Neural collaborative filtering. In *Proceedings of the 26th International Conference on World Wide Web*. 173–182.
- [19] Xiangnan He, Hanwang Zhang, Min-Yen Kan, and Tat-Seng Chua. 2016. Fast Matrix Factorization for Online Recommendation with Implicit Feedback. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*. 549–558.
- [20] Yifan Hu, Yehuda Koren, and Chris Volinsky. 2008. Collaborative Filtering for Implicit Feedback Datasets. In *Proceedings of the 8th IEEE International Conference on Data Mining*. 263–272.
- [21] Yehuda Koren. 2008. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 426–434.
- [22] Dmitri V. Krioukov, Fragkiskos Papadopoulos, Maksim Kitsak, Amin Vahdat, and Marián Boguñá. 2010. Hyperbolic Geometry of Complex Networks. In *CoRR abs/1006.5169*.
- [23] Kevin Lewis, Marco Gonzalez, and Jason Kaufman. 2012. Social selection and peer influence in an online social network. (2012), 68–72.
- [24] Tzu-Heng Lin, Chen Gao, and Yong Li. 2018. Recommender Systems with Characterized Social Regularization. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*. 1767–1770.
- [25] Yang Liu, Liang Chen, Xiangnan He, Jiaying Peng, Zibin Zheng, and Jie Tang. 2020. Modelling High-Order Social Relations for Item Recommendation. In *CoRR abs/2003.10149*.
- [26] Hao Ma, Irwin King, and Michael R. Lyu. 2009. Learning to recommend with social trust ensemble. In *Proceedings of the 32nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. 203–210.
- [27] Hao Ma, Haixuan Yang, Michael R. Lyu, and Irwin King. 2008. SoRec: social recommendation using probabilistic matrix factorization. In *Proceedings of the 17th ACM Conference on Information and Knowledge Management*. 931–940.
- [28] Hao Ma, Dengyong Zhou, Chao Liu, Michael R. Lyu, and Irwin King. 2011. Recommender systems with social regularization. In *Proceedings of the 4th International Conference on Web Search and Web Data Mining*. 287–296.
- [29] Maas, Andrew L., Awni Y. Hannun, and Andrew Y. Ng. 2013. Rectifier nonlinearities improve neural network acoustic models. In *ICML*.
- [30] Leyla Mirvakhabova, Evgeny Frolov, Valentin Khrukov, Ivan V. Oseledets, and Alexander Tuzhilin. 2020. Performance of Hyperbolic Geometry Models on Top-N Recommendation Tasks. In *Fourteenth ACM Conference on Recommender Systems*. 527–532.
- [31] Maximilian Nickel and Douwe Kiela. 2017. Poincaré Embeddings for Learning Hierarchical Representations. In *Advances in Neural Information Processing Systems* 30. 6338–6347.
- [32] Maximilian Nickel and Douwe Kiela. 2018. Learning Continuous Hierarchies in the Lorentz Model of Hyperbolic Geometry. In *Proceedings of the 35th International Conference on Machine Learning*. 3776–3785.
- [33] Steffen Rendle. 2010. Factorization Machines. In *Proceedings of the 10th IEEE International Conference on Data Mining*. 995–1000.
- [34] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian Personalized Ranking from Implicit Feedback. In *Proceedings of the 25th Conference on Uncertainty in Artificial Intelligence*. 452–461.
- [35] Frederic Sala, Christopher De Sa, Albert Gu, and Christopher Ré. 2018. Representation Tradeoffs for Hyperbolic Embeddings. In *Proceedings of the 35th International Conference on Machine Learning*. 4457–4466.
- [36] Ruslan Salakhutdinov and Andriy Mnih. 2007. Probabilistic Matrix Factorization. In *Proceedings of the 21st Annual Conference on Neural Information Processing Systems*. 1257–1264.
- [37] Badrul Munir Sarwar, George Karypis, Joseph A. Konstan, and John Riedl. 2001. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th International World Wide Web Conference*. 285–295.
- [38] Timothy Schmeier, Joseph Chisari, Sam Garrett, and Brett Vintch. 2019. Music recommendations in hyperbolic space: an application of empirical bayes and hierarchical poincaré embeddings. In *Proceedings of the 13th ACM Conference on Recommender Systems*. 437–441.
- [39] Jiliang Tang, Suhang Wang, Xia Hu, Dawei Yin, Yingzhou Bi, Yi Chang, and Huan Liu. 2016. Recommendation with Social Dimensions. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*. 251–257.
- [40] Alexandru Tifrea, Gary Bécigneul, and Octavian-Eugen Ganea. 2019. Poincaré Glove: Hyperbolic Word Embeddings. In *7th International Conference on Learning Representations*.
- [41] Lucas Vinh Tran, Yi Tay, Shuai Zhang, Gao Cong, and Xiaoli Li. 2020. HyperML: A Boosting Metric Learning Approach in Hyperbolic Space for Recommender Systems. In *The Thirteenth ACM International Conference on Web Search and Data Mining*. 609–617.
- [42] Tran Dang Quang Vinh, Yi Tay, Shuai Zhang, Gao Cong, and Xiao-Li Li. 2018. Hyperbolic Recommender Systems. In *CoRR abs/1809.01703*.
- [43] Xiao Wang, Yiding Zhang, and Chuan Shi. 2019. Hyperbolic Heterogeneous Information Network Embedding. In *The Thirty-Third AAAI Conference on Artificial Intelligence*. 5337–5344.
- [44] Le Wu, Peijie Sun, Yanjie Fu, Richang Hong, Xiting Wang, and Meng Wang. 2019. A Neural Influence Diffusion Model for Social Recommendation. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*. 235–244.
- [45] Hanwang Zhang, Fumin Shen, Wei Liu, Xiangnan He, Huanbo Luan, and Tat-Seng Chua. 2016. Discrete Collaborative Filtering. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*. 325–334.
- [46] Yiding Zhang, Xiao Wang, Xunqiang Jiang, Chuan Shi, and Yanfang Ye. 2019. Hyperbolic Graph Attention Network. In *CoRR abs/1912.03046*.
- [47] Çağlar Gülçehre, Misha Denil, Mateusz Malinowski, Ali Razavi, Razvan Pascanu, Karl Moritz Hermann, Peter W. Battaglia, Victor Bapst, David Raposo, Adam Santoro, and Nando de Freitas. 2019. Hyperbolic Attention Networks. In *ICLR*.