HCGR: Hyperbolic Contrastive Graph Representation Learning for Session-based Recommendation

Naicheng Guo∗ Xiaolei Liu∗ Shaoshuai Li∗
Qiongxu Ma∗ Yunan Zhao∗ Bing Han∗ Lin Zheng†
Kaixin Gao‡ Xiaobo Guo∗

Abstract

Session-based recommendation (SBR) learns users’ preferences by capturing the short-term and sequential patterns from the evolution of user behaviors. Among the studies in the SBR field, graph-based approaches are a relatively powerful kind of way, which generally extract item information by message aggregation under Euclidean space. However, such methods can’t effectively extract the hierarchical information contained among consecutive items in a session, which is critical to represent users’ preferences. In this paper, we present a hyperbolic contrastive graph recommender (HCGR), a principled session-based recommendation framework involving Lorentz hyperbolic space to adequately capture the coherence and hierarchical representations of the items. Within this framework, we design a novel adaptive hyperbolic attention computation to aggregate the graph message of each user’s preference in a session-based behavior sequence. In addition, contrastive learning is leveraged to optimize the item representation by considering the geodesic distance between positive and negative samples in hyperbolic space. Extensive experiments on four real-world datasets demonstrate that HCGR consistently outperforms state-of-the-art baselines by 0.43%-28.84% in terms of HitRate, NDCG and MRR.

∗Ant Group, Beijing, China. Email:{guonaicheng.gnc, liuxiaolei.lxl, lishaoshuai.lss}@alibaba-inc.com, {qiongxu.mqx, yunan.zya, hanbing.hanbing, jefflittleguo.gxb}@antgroup.com.
†Department of Computer Science, Shantou University, Shantou, China. Email:lzheng@stu.edu.cn.
‡School of Mathematics, Tianjin University, Tianjin, China. Email: gaokaixin@tju.edu.cn.
1 Introduction

In many e-commerce online scenarios, user profiles usually cannot be obtained so that session-based recommendation has become an important solution to address anonymous recommendation. Session-based recommender (SBR) system learns users’ preferences by mining sequential patterns from users’ chronological historical behavior without user profiles to predict users’ future interests in one session. Most traditional Markov chain based SBR models, e.g., FPMC [1], FOSSIL [2], conduct sequence modeling and prediction only by considering user’s last behavior. Lately, RNN-based models treat historical behaviours of each user as a strictly-order, temporally dependent sequence like linguistic sentences. These methods like GRU4REC [3] surprisingly increase the performance in many real scenarios because of their effectiveness in storing short-term information. However, they assume that the adjacent items in a session have a fixed sequential dependence which is unable to capture the user interest variations, as a result, they are prone to introduce wrong dependencies. To address this issue, later models that fusing self-attention mechanisms like SASREC, were proposed. Guo et al. [4] further improve the attention-based approaches by introducing specialized human sentiment factors. The aforementioned attention-based methods prefer to model unidirectional message transformation between adjacent items in a sequence. Such transformation may lose the insight of the relevant information of the whole sequence. For example, in a music player APP, although a user may randomly play an album or a certain type of music, which will generate different playback records, it does not mean that the user’s interest has changed. In other words, strictly modeling the user’s local click records and ignoring the global relationship may lead to overfitting. To resolve the problem in attention-based methods, GNN-based models like SR-GNN [5] and GC-SAN [6] utilize graphs to capture the coherence of items within a session due to their powerful ability to represent structured data and adopt attention layers to learn long-term dependence. Despite the leading performance of GNN-based models compared with traditional SBR methods, great challenges remain.

Challenge1: The user’s interests are extensive and hierarchical, which can be expressed as a power-law distribution of items clicked by users. The existing session-based recommendation methods learn the representations in Euclidean space, which can’t effectively capture the information of such hierarchical, or in other words, tree-structured data.

Challenge2: Recent studies have proved that hierarchical data can be better explained under Non-Euclidean geometry of low-dimensional manifolds. But in the GNN-based methods, introducing Non-Euclidean transformation will result in the discrepancy between Euclidean and Non-Euclidean space when aggregating neighbouring messages and applying attention mechanism.

These challenges remain prevalent in real-world recommendation scenarios since it
has been demonstrated that users’ behaviors like clicking or purchasing have prototypical characteristics of complex structures, which are generally power-law distributed [7–9]. Aforementioned, data with hierarchical structure could be well represented in hyperbolic space. This has motivated representation learning in hyperbolic space to effectively capture the information of the user behaviors with hierarchical property [10]. Furthermore, the hyperbolic representations can naturally capture the similarity and hierarchy by their distance. To illustrate the difference between Euclidean and hyperbolic space, we visualized them in Figure 1 and Figure 2. In a two-dimensional Euclidean space as showed in Figure 1, the number of nodes increases polynomial in the center to the radius. By contrast, in two-dimensional hyperbolic space in Figure 2, the number of nodes increases exponentially in the center to the given radius, and the hyperbolic space has more powerful (exponential-level) representation ability than then Euclidean space [11].

In conclusion, given the same radius, hyperbolic space has larger space, thereby including more nodes. Therefore, the general representational capacity of Euclidean space can be summarized as a square level, which can cause high distortion if we model the data of hierarchical relational users’ preferences.

To overcome these session-based challenges above, in this paper we propose a novel graph neural network framework, namely Hyperbolic Contrastive Graph Recommender (HCGR), upon hyperbolic space, specifically, Lorentz hyperbolic space for its simplicity and stability, to optimize the underlying hierarchical embeddings. First, we embed items with dense and effective representations, which are predisposed to preserving their internal hierarchical properties in Lorentz hyperbolic space. To ensure the correctness of the necessary representations’ transformation, we utilize some specific operations based on the Lorentz hyperboloid model. Second, we construct an improved graph neural network framework and a novel message propagation mechanism to model the preferences in user behavior sequences. To enable the model better distinguish users’ preferences for different items, we propose an adaptive hyperbolic attention calculation method. Third, we introduce contrastive learning to optimize the model by considering the distance between positive and negative samples in hyperbolic space.

Overall, we summarize our main contributions of this work as follows:

- We exploit hyperbolic item representation for session-based recommendation, to
the best of our knowledge, our method is the first one to extract hierarchical information of user behaviors within a session under hyperbolic space for SBR tasks.

- We design a novel attention calculation approach in hyperbolic space to deal with graph information aggregating, which can’t be effectively implemented by the existing aggregation methods in Euclidean space.

- We introduce contrastive learning to optimize the item representation by considering the geodesic distance between positive and negative samples in Lorentz hyperbolic space.

- We conduct extensive experiments on three public datasets and one financial service industrial dataset, which show that our Lorentz hyperbolic session-based recommendation framework can achieve better performance compared to the state-of-the-art SBR methods in terms of \( \text{HitRate}@K \), \( \text{NDCG}@K \) and \( \text{MRR}@K \), where \( K = 10, 20 \).

The structure of this paper is as follows. Section 2 presents the related work, including session-based recommendation and hyperbolic learning. Section 3 introduces the preliminary work of this paper, including the definition of the problem and some basic knowledge of the Lorentz hyperbolic space. Section 4 introduces the implementation details of HCGR framework, and section 5 gives the experimental results. Finally, conclusions are set out in section 6.

2 Related work

In section 2.1 we review a line of representative works on session-based recommendation, including traditional MCs (markov chains) based models, RNNs (recurrent neural networks) based models, attention-based models, and GNNs (graph neural networks) based models. Then, in section 2.2 we discuss related hyperbolic representation learning methods related to our proposed HCGR.

2.1 Session-based Recommendation

2.1.1 Markov chain models

Early sequential recommendation methods mainly rely on Markov chains. For example, FPMC [1] combines MF (matrix factorization) and MC to the learn general preference and local interest of users for the next basket recommendation. HRM [12] applies nonlinear operations to extract more complex pattern of both user’s sequential behavior and interests. Fossil [2] fuses similarity-based methods with Markov chains to conduct personalized sequential recommendations. A shortcoming of MC-based models is that it is difficult to learn long-term dependencies because MC-based models assume that
the next state is only related to the previously nearest state. Although some high-order Markov models can associate the next state with several previous states, they consume high computational cost [13,14].

2.1.2 Recurrent neural networks models

In recent years, researchers adopt RNNs to capture time dependency in temporal data. The first RNN-based sequential recommendation method is GRU4REC [3], which uses GRU (gated recurrent unit) to capture long-term dependencies among sessions. Leveraging a novel pair-wise ranking loss, GRU4Rec [3] significantly outperforms MC-based approaches. Inspired by GRU4REC [3], MV-RNN [15] incorporates visual and textual information to alleviate the item cold start problem. Furthermore, ROM [16] utilizes an interactive self-attention mechanism to adaptively reorganize the entity memory and the topic memory for the rating prediction task. However, RNN-based methods assume that the adjacent items in a session have a fixed sequential dependence, which may generate wrong dependencies and introduce noises in real-world scenarios like music playing.

2.1.3 Attention mechanism

Recent models with attention mechanism [17] perform particularly well in sequential recommendation. Li et al. explores a hybrid encoder with attention mechanism to model users’ sequential behaviors and user’s interests in the same session [18]. A short-term attention priority model STAMP [13] is proposed, which can capture both personal interest from the long-term memory of the session context, and user’s current interest from the short-term behaviors. SASREC [19] effectively captures users’ long-term preferences from the sparse and dense datasets, and FDSA [20] puts the features of behaviours and items into two distinct independent blocks of self-attention to model the transition patterns of the items and the transition patterns of the items and achieve remarkable effects.

2.1.4 Graph neural networks

Most advanced sequential recommendation models apply the self-attention mechanism to capture user behavior relations in a long sequence. However, it is a challenge to find out both implicit and explicit relation between adjacent behaviors. GNNs can find out such relations effectively [14,21] and can capture complex interaction of user behaviors. For example, SR-GNN [5] instructively constructs a digraph for each sequence and CS-SAN [6] further incorporates self-attention mechanism to generate the representation of the constructed digraph. In addition, Wu et al. focuses on users in the session and models their historical sequence by dot-attention mechanism [22]. FGNN [23] proposes a weighted attention layer and a readout function to learn item embeddings and session embeddings for next item recommendation. Recently, Ma et al. utilizes memory models to capture both the long term and short-term user behaviors [24]. Wang et al. introduces
a new GNN-based method that can learn global relations between items [25]. Chen et al. [26] utilizes several ways to reduce the information within message propagation. GNN-based methods have yielded many fruitful results, but the existing methods generally model user behavior in tangent space, and the representation learned in tangent space is limited to capturing attributes of shallow properties and lack of hierarchy. In this work, we aim to learn hierarchical item representation in Lorentz hyperbolic space, and we want to find deep user behaviors patterns in session-based recommendation.

2.2 Hyperbolic Learning

Recently, many studies have shown that complex data may exhibit a highly non-Euclidean structure [27,28]. Researchers are increasingly considering building complex neural networks on Riemannian space, in which the hyperbolic space with negative constant curvature is an attractive option [29,30]. In many domains, such as sentences in natural language [31], social networks [29], biological protein graph [30], etc., data usually have a tree-like structure or can be represented hierarchically, and hyperbolic space is equipped to model hierarchical structures naturally and effectively [32]. Due to its strong representation ability, hyperbolic space has been applied in many application areas [33–37]. For instance, Liu et al. [33] proposed Hyperbolic Graph Convolutional Networks used for graph representation learning by combining the expressiveness of hyperbolic space and Graph Convolutional Networks. Chen et al. [35] proposed a hyperbolic interaction model for multi-label classification tasks. These works have shown the advantages and effectiveness of hyperbolic space in learning hierarchical structures of complex relational data.

Noticing the potential of hyperbolic space in learning complex interactions between users and items, many researchers have tried to apply hyperbolic learning to recommendation systems [38–42]. Chamberlain et al. [38] proposed a large-scale recommender system based on the hyperbolic space, which can be scaled to millions of users. [39] considered constructing multiple hyperbolic manifolds to map the representation of user and ad, and proposed a framework that can effectively learn the hierarchical structure in users and ads based on the hyperbolic space. Ma et al. [40] proposed a recommendation model in the hyperbolic space for Top-K recommendation. Li et al. [41] presented the Hyperbolic Social Recommender which utilized hyperbolic geometry to boost the performance. Wang et al. [43] proposed a novel graph neural network framework (HyperSoRec) combing hyperbolic learning for the social recommendation.

3 Preliminaries

In this section, we introduce basic knowledges about hyperbolic geometry and graph neural networks works related to our proposed HCGR.
3.1 Hyperbolic Geometry

Hyperbolic space is a Riemannian surface with negative curvature. Several hyperbolic geometric models have been widely used, including Poincare disk model [31, 42], Klein model [44] and Lorentz (hyperboloid) model [45]. All these hyperbolic models are isometrically equivalent, i.e., any point in one of these models can be transformed to another point with distance-preserving transformations [46]. In this paper, we choose the Lorentz model as the framework cornerstone, because of the numerical stability and calculation simplicity of its exponential/logarithm maps and its distance function. In hyperbolic geometry, we use the Lorentz formulation to model the network, which is found to be more stable for numeric optimization patterns [45]. We want to learn $d$-dimensional user and item embeddings.

A $d$-dimensional hyperbolic space is a Riemannian manifold $\mathcal{M}$ with a constant negative curvature, which is denoted by $c$. The negative reciprocal of the curvature is denoted by $k = -\frac{1}{c}$, where $k > 0$. The Lorentz representation is defined by the pair $\mathcal{L}^d = (\mathcal{H}^d, g_\mathcal{L})$ and

$$\mathcal{H}^d = \{ x \in \mathbb{R}^{d+1} | \langle x, x \rangle_\mathcal{L} = -k, x_0 > 0 \},$$

where $\langle x, y \rangle_\mathcal{L}$ is the Lorentz inner product given by

$$\langle x, y \rangle_\mathcal{L} = -x_0 y_0 + \sum_{i=1}^{d} x_i y_i, \quad \forall \ x, y \in \mathbb{R}^{d+1},$$

and the metric matrix $g_\mathcal{L}$ is given by

$$g_\mathcal{L} = \begin{bmatrix} -1 & 1 & \ldots & 1 \\ \vdots & \ddots & \ddots & \vdots \\ 1 & \ldots & 1 & -1 \end{bmatrix}.$$

The distance function induced by the metric $g_\mathcal{L}$ is

$$d_\mathcal{L}(x, y) = \sqrt{k} \text{arccosh} \left( \frac{-\langle x, y \rangle_\mathcal{L}}{k} \right).$$

For any pair of points $x, y \in \mathbb{R}^{d+1}$, the tangent space $T_x \mathcal{H}^d$ at point $x$ is a $d$-dimensional Euclidean space. The elements of $T_x \mathcal{H}^d$ are referred to as tangent vectors and satisfying

$$T_x \mathcal{H}^d = \{ v \in \mathbb{R}^{d+1} | \langle v, x \rangle_\mathcal{L} = 0 \}.$$  

3.2 Graph Neural Network

GNNs are neural networks that can handle graph-structured data directly. They are often applied in classification, link prediction and graph classification tasks. In this
paper, we focus on graph classification, because we formulize each user’s behavior to a graph and we want to learn a representation from it rather than a single node.

Let $G(V,E)$ denotes a given graph, where $V$ and $E$ are the set of the nodes and edges respectively, where $x_v$ represents the feature vector of $v \in V$, which is the initial embedding of node $v$. To specific, we formulize the graph classification task as follow. Our work is to learn a classifier $f$ and the graph-level representation $H$ to predict the label of the graph. Given a collection of graphs $(G_1, G_2, \ldots, G_n) \in G$ and the corresponding labels $(v_{L_1}, v_{L_2}, \ldots, v_{L_n}) \in V_L$.

GNNs use the structure of graph and the original feature of each node to learn its corresponding representation. The learning process is to take a node as the center, and iteratively aggregate the neighborhood information along edges. The information aggregation and update process can be formulated as follows:

$$t_v^{(l+1)} = f_{\text{aggregator}}(x_v^{(l)}, u \in N(v)),$$
$$x_v^{(l+1)} = f_{\text{updater}}(x_v^{(l)}, t_v^{(l+1)}).$$

where $x_v^l$ represents the embedding of node $v$ after $l$-th layer aggregator and $N(v)$ is neighborhood of node $v$. The information aggregation function $f_{\text{aggregator}}$ aggregates the information from the neighborhood information and passes it to the target $v$. The update function $f_{\text{updater}}$ calculates the new node statues from the source embedding $x_v^l$ and the aggregated information $t_v^{l+1}$.

After $l$ steps of information aggregation, the final embedding gather the $l$-hop neighborhood and the structure information. For the graph classification task, readout function $f_{\text{readout}}$ generates a graph level embedding $Z$ by gathering the embeddings of all node in the final layers:

$$Z = f_{\text{readout}}(\{x_v^{(l)}, v \in V\}).$$

4 Methodolodgy

In this section, we describe the implementation details of the HCGR framework. First, in section 4.1 we illustrate the notations used in this paper and define the session-based recommendation task. In section 4.2, we transform the user behaviors within a session into a session graph, and present the HCGR’s overall pipeline in Figure 3. In section 4.3, we introduce the embeddings in Lorentz hyperbolic space. Next, we describe the novel attention mechanism that is especially designed for hyperbolic geometry in section 4.4. After learning the embeddings, we set up the hyperbolic attention mechanism to construct representation of user behaviors (section 4.5). Finally, we describe the contrastive learning with hyperbolic space distance (section 4.6).
Figure 3: The Architecture of the HCGR Framework. It takes user’s historical sequential items as input, and then build the directed graph being projected to the hyperbolic space, finally outputs the predicted probability of the next item that the user most likely to click.

Table 1: The key mathematical notations

| Notation | Description |
|----------|-------------|
| $\mathcal{H}^d$ | a hyperbolic space of dimension $d$ |
| $\mathcal{M}$ | Riemannian manifold |
| $S$ | a given session |
| $c$ | the curvature of hyperbolic space |
| $k$ | the negative reciprocal of the curvature $c$ |
| $\mathcal{T}_x\mathcal{H}^d$ | the tangent space at point $x$ with dimension $d$ |
| $x_v$ | a item embedding in the Euclid space |
| $l$ | the layer of graph neural network |
| $x^H$ | a item embedding in the hyperbolic space |

4.1 Notation and Problem Definition

A session-based recommendation task is constructed on historical user behavior sessions, and makes predictions based on current user sessions. In this task, there is an item set $V$, where $m = |V|$ is the number of items and all items are unique. Each session $S = [v_1, v_2, \ldots, v_n]$ is composed of a series of user’s interactions, where $v_i$ represents an item clicked at the $i$-th position in $S$ and $n$ represents session’s length. Our purpose of session-based recommendation is to predict the item that the user is most likely to click on next time in a given session $S$.

For each given session $S$ in the training process, there is a label as the target. In the training process, for each item $v_i \in V$ in given session, our model learns the corresponding embedding vector $x_v \in \mathbb{R}^d$, where $d$ is the dimension of vector $x_v$. Our model outputs a probability distribution $\hat{y}$ over the given session $S$, where the item with Top-$K$ value will be regarded as the candidate for Top-$K$ recommendation.
4.2 Behaviors Graph

Because graph neural network can’t deal with session directly, the first thing we need to do is converting a given session \( S = [v_1, v_2, \ldots, v_n] \) to the session graph \( G_s \). According to the analysis of datasets, it is very common for users to click the same item multiple times within the session. Because the user’s behavior is chronological and the same item may be clicked multiple times, we choose the weighted directed graph to represent the changing process of the given session \( S \). All the sessions will be converted into session graphs. To show this process more clearly, we show the process of this session converter in Figure 4. We use \( E_s \) to denote all weighted directed edges set. Its elements are composed of \((v_t, v_{t+1}, w_{t,t+1})\), where \( v_t, v_{t+1} \) is the item clicked at timestamp \( t, t+1 \) respectively, \( w_{t,t+1} \) denotes the weighted directed edge between \( v_t \) and \( v_{t+1} \). Note that if the node does not have a self-loop, we will add a self-loop with weight 1 to it. Each node represents the unique item in the session and the features \( x_v \) are initialized in the Lorentz hyperbolic space which introduced in section 4.3.

Figure 4: An example of how to convert a session to graph

4.3 Embeddings in Lorentz Hyperbolic Space

We use the representation from Lorentz hyperbolic space for item embedding. The \( k = -\frac{1}{2} \) is the reciprocal of curve \( c \), which treated here as a trainable parameter and initials empirically. Then we fix the origin \( o = (\sqrt{k}, 0, \ldots, 0) \in \mathcal{H}^d \) and use it as a reference point. The embeddings are initialized by sampling the Gaussian distribution on the Euclidean space \( \mathcal{T}_x \mathcal{H}^d \) of the reference point \( o \).

We denote the mapping between hyperbolic spaces and tangent spaces as exponential map and logarithmic map, respectively. The exponential map is a map from subset of a tangent space \( \mathcal{T}_x \mathcal{H}^d \) to \( \mathcal{H}^d \). The logarithmic map is the reverse map that maps \( \mathcal{H}^d \) back to the tangent space \( \mathcal{T}_x \mathcal{H}^d \). For any \( x, y \in \mathcal{H}^d \) and \( v \in \mathcal{T}_x \mathcal{H}^d \) satisfying \( x \neq y \) and
\( \mathbf{v} \neq 0 \), the exponential map \( \exp^k_\mathbf{x}(\cdot) \) and logarithmic map \( \log^k_\mathbf{x}(\cdot) \) are defined as follows:

\[
\exp^k_\mathbf{x}(\mathbf{v}) = \cosh \left( \frac{\|\mathbf{v}\|_\mathcal{L}}{\sqrt{k}} \right) \mathbf{x} + \sqrt{k} \sinh \left( \frac{\|\mathbf{v}\|_\mathcal{L}}{\sqrt{k}} \right) \frac{\mathbf{v}}{\|\mathbf{v}\|_\mathcal{L}},
\]

(4.8)

\[
\log^k_\mathbf{x}(\mathbf{y}) = d_\mathcal{L}(\mathbf{x}, \mathbf{y}) \frac{\mathbf{y} + \frac{1}{k} \langle \mathbf{x}, \mathbf{y} \rangle \mathcal{L} \mathbf{x}}{\|\mathbf{y} + \frac{1}{k} \langle \mathbf{x}, \mathbf{y} \rangle \mathcal{L} \mathbf{x}\|_\mathcal{L}},
\]

(4.9)

where \( \|\mathbf{v}\|_\mathcal{L} = \sqrt{\langle \mathbf{v}, \mathbf{v} \rangle_\mathcal{L}} \) is the Lorentzian norm.

### 4.4 Hyperbolic Graph Attention Network

Following the mapping layer, how to model session graph and mine user preferences is the key issue. Users typically click on several items they like, and these items have rich hierarchical structure. As a result, we propose a novel information aggregation with attention mechanism in hyperbolic space to capture the influence of different items on user preferences during the process of information propagation. Hyperbolic space can better represent item [38, 39], but we still face a technical challenge, the traditional hyperbolic model does not define the necessary vector operation process, such as vector addition and multiplication etc. Inspired by previous works [38,39,43], we formulate the multiplication and addition operation in hyperbolic space as follow:

\[
W \otimes^k_\mathcal{H} \mathbf{x} := \exp^k_0(W \log^k_0(\mathbf{x}^\mathcal{H})),
\]

(4.10)

\[
\mathbf{x}^\mathcal{H} \oplus^k_\mathcal{H} \mathbf{b} := \exp^k_{\mathcal{H}}(P^{k}_{\mathcal{H}}(\mathbf{x}^\mathcal{H}(\mathbf{b}))),
\]

(4.11)

where \( P^{k}_{\mathcal{H}}(\mathbf{x}^\mathcal{H}(\mathbf{b})) \) is the Parallel Transport: for two point \( \mathbf{x}, \mathbf{y} \) on the Lorentz space \( \mathcal{H}^d \), the parallel transport of a tangent vector \( \mathbf{v} \in T_{\mathbf{x}}\mathcal{H}^d \) to the tangent space \( T_{\mathbf{y}}\mathcal{H}^d \) is:

\[
P_{\mathbf{x} \rightarrow \mathbf{y}}(\mathbf{v}) = \mathbf{v} - \frac{1}{d_\mathcal{L}(\mathbf{x}, \mathbf{y})} \frac{\langle \log_\mathbf{x}(\mathbf{y}), \mathbf{v} \rangle_\mathcal{L}}{\| \log_\mathbf{x}(\mathbf{y}) + \log_\mathbf{y}(\mathbf{x}) \|_\mathcal{L}}.
\]

(4.12)

Non-linear activation with different curvatures is proposed as follow:

\[
\sigma^{k_{l+1} - k_l}(\mathbf{x}^\mathcal{H}) = \exp^{k_{l+1}}_0(\sigma(\log^{k_l}_0(\mathbf{x}^\mathcal{H}))),
\]

(4.13)

where \(-\frac{1}{k_{l+1}}, -\frac{1}{k_l}\) is the hyperbolic curvature at layer \( l, l+1 \) respectively. To specific, we project the embedding from the tangent space \( T_{\mathbf{x}}\mathcal{H}^d \) to \( \mathcal{H}^d \) via exponential map \( \exp^k_\mathbf{x}(\cdot) \) according to the Eq(4.8). Then we perform the addition and multiplication operation according to the above equation.

The crucial idea of traditional GNNs is to learn representations in the given graph by iteratively aggregating and capture multi-hop neighborhood structures and features. The process of information aggregation usually consists of two parts: feature transformation and nonlinear activation. Recent studies have shown that, compared to the simple
average aggregation method, the gain with features transformation and non-linear activation is rarely, and may even bring negative gain. In addition, these two operations may lead to significant over fitting of highly sparse user behavior [47]. Based on these studies, we remove the unnecessary feature transformation and non-linear activation to accelerate the training, inference and reduce the complexity of our framework.

To make better use of the representation ability of Lorentz space, we redesigned a way of information aggregation in hyperbolic space. We refer to the ideas of GCN [48] and GAT [49] to calculate the attention weight between target node and neighbors respectively. The detailed calculation way is shown as follow:

\[ w_{ij} = \text{Softmax}_{j \in N(i)}(W_a (\log^k o(x^{H,l}_i)) \| \log^k o(x^{H,l}_j)) + b_a), \quad (4.14) \]

\[ x^{H,l+1}_i = \exp^k_{x^{H,l}_i} (\sum_{j \in N(i)} w_{ij} \log^k o(x^{H,l}_j)), \quad (4.15) \]

where \( W_a \) is \( \mathbb{R}^{1 \times (2d+2)} \) matrices, and \( b_a \) is a constant number. The learning process is to take a node \( i \) as the center, and iteratively aggregate the neighbors information along edges. For each node \( i \), in hyperbolic mechanism, all attention coefficients of their neighbors can be calculated as Eq(4.14). In order to use these attention coefficients, a linear combination for the neighbors is used for updating the embeddings of the nodes.

To take full advantage of higher-order relationships, we need to stack multiple hyperbolic attention layers together.

\[ z^H_i = \exp^k (\alpha_0 * \log^k o(x^{H,0}_i) + \alpha_1 * \log^k o(x^{H,1}_i) + \ldots + \alpha_l * \log^k o(x^{H,l}_i)). \quad (4.16) \]

### 4.5 Hyperbolic Attention Mechanism

After we obtain the graph level representation \( Z^H = [z^H_1, z^H_2, \ldots, z^H_n] \), we want to utilize self-attention mechanism to better capture user’s preference. Self-attention is an important part of attention mechanism, it has yielded many fruitful results such as: [17], [19], [20]. The self-attention mechanism can calculate the global dependence between user behavior and capture the item transformation relations of the whole session sequences. The original self-attention mechanism does not define in hyperbolic space, so we extend the self-attention mechanism to hyperbolic space and we formalize the hyperbolic self-attention mechanism as follows:

\[ F^H = \exp^k (\text{Softmax} (\frac{W^Q \log^k o(Z^H)) (W^k \log^k o(Z^H))^T}{\sqrt{d+1}} (W^V \log^k o(Z^H)))), \quad (4.17) \]

where \( W_Q, W_K \) and \( W_V \) are \( \mathbb{R}^{(d+1) \times (d+1)} \) matrices. It will receive the query \( Q \), key \( K \), and value \( V \), and calculate the similarity between each element in the session through the scaled dot-product, so as to characterize the user’s long-term preference, where \( d \) is the dimension of the input vector and \( \sqrt{d+1} \) is the scale factor, which
is used to prevent the gradient vanishing problem caused by the large value after the dot product. Correspondingly, Element-wise Feed-Forward is also extended to hyperbolic space and it is given by:

$$E_{H}^{k} = \exp^{k_{2}}(\log^{k_{2}}(W_{2} \otimes^{k_{2}} \sigma^{k_{2},k_{1}}(W_{1} \otimes^{k_{1}} F_{H}^{k_{1}} \oplus^{k_{1}} b_{1})) \oplus^{k_{2}} b_{2}) + \log^{k_{1}}(F_{H}^{k_{1}})),$$

(4.18)

where $W_{1}$ and $W_{2}$ are $\mathbb{R}^{(d+1)\times(d+1)}$ matrices, $b_{1}$ and $b_{2}$ are $(d+1)$-dimensional bias vectors. It takes full account of the interaction between the dimensions of various vectors through nonlinear activation function and linear transformation. A skip connection after the feed forward network, which makes the model reduce the loss of information and takes advantage of the low-layer information. For simplicity, we define the entire hyperbolic self-attention mechanism above as:

$$E_{H} = \text{Hyp-Self-Att}(Z_{H}).$$

(4.19)

Recent studies have shown that, different layers of self-attention mechanism may capture different types of features, so it is necessary to increase the number of layers appropriately to enhance the model expression. The multi-layer hyperbolic self-attention mechanism is define as:

$$E_{H,j}^{k} = \text{Hyp-Self-Att}(E_{H,j}^{k-1}).$$

(4.20)

Finally, the hyperbolic self-attention mechanism output is $E_{H,j}^{k}$. After $j$-th adaptive hyperbolic self-attention blocks, we obtain the long-term attentive session representation $E_{j}$. The short-term interest describes the current preferences of users. It is based on several items recently visited as the basis for prediction. The next behavior of users is often closely related to his recent interests. In order to better model the relationship in the whole session, we set up a gated mechanism to capture both long-term and short-term preference.

$$o = w \log_{k}(E_{(n)}^{H,j}) + (1 - w) \log_{k}(z_{n}^{H}),$$

(4.21)

where $E_{(n)}^{H,j}$ denotes the embedding corresponding of the last item in the given session $S$.

Finally, after we get a unified preference representation $o$, we make a recommendation score for each element in the item set $V$.

$$\hat{y}_{i} = \text{Softmax}(o^{T}v_{i}),$$

(4.22)

where $\hat{y}_{i}$ is the recommendation probability of our framework for item $v_{i}$. For the session-based recommendation task, we select the highest $K$ probabilities item from item set $V$ as final result according to $\hat{y}_{i}$.

### 4.6 Contrastive Learning

By projecting the item embedding into hyperbolic space, we empower the performance of our framework. In the recommendation scenario, there are many similar items, but users usually only choose their favorite items, so if we can let the model distinguish
this subtle distinction, it may significantly improve the recommendation ranking performance of our framework. Inspired by the successful practice of contrastive learning, we introduce contrastive learning in an innovative way into the framework in order to increase the modeling of user behavior. Compared with simple contrastive learning, the operation of our framework is calculating in hyperbolic space, which will be somewhat more complicated. Specifically, we want to make the best use of the distance between items in hyperbolic space through contrastive learning, then the recommendation model perceives more subtle distinction and improve the ranking performance.

We formulate our objective into two parts, the first part $L_e$ is cross-entropy loss function, which has been widely used in recommender system. The second part is the contrastive ranking loss $L_c$ with margin. The purpose of $L_c$ is to separate the positive and negative pairs up to a given margin. When the margin is reached, the pairs of items are considered to be properly segregated and with little loss. This enables the model keep focus on the pairs of items that are not near the margin and the margin separation is optimized in Lorentz hyperbolic space.

$$L_e = - \sum_{i=1}^{n} (y_i \log(\hat{y}_i) + (1 - y_i) \log(1 - \hat{y}_i)), \quad (4.23)$$

$$L_c = \sum_{i=1}^{n} \max((d_L(\hat{y}_i, y_P) - d_L(\hat{y}_i, y_N) + \xi, 0), \quad (4.24)$$

$$L_{total} = \gamma * L_e + \beta * L_c \quad (4.25)$$

where $\gamma$ and $\beta$ control the magnitude of the cross entropy loss and contrastive ranking loss respectively.

5 Experiment

In this section, detailed experiments will be conducted to assess the performance of the HCGR framework. We intend to answer following questions:

- **RQ1**: How does our proposed method perform comparing with the state-of-the-art methods?

- **RQ2**: How the different components (i.e., Lorentz transportation, multi-hop graph aggregation and contrastive learning) affect the performance of HCGR?

- **RQ3**: Can HCGR present reasonable explanation with regard to predicting user preference and get better recommendation results?

In particular, we first describe the datasets and experimental configuration (section 5.1). Then we compare the effectiveness of HCGR with several comparison methods (section 5.2). In section 5.3, we analyze in detail the generalization capability and the possibility of migration of the HCGR. Lastly, section 5.4, we set the case study and visualize the embedding in hyperbolic space (section 5.5).
Table 2: statistics of datasets

| Dataset   | Users   | Items   | Avg.I/user | Avg.I/item | Behaviors |
|-----------|---------|---------|------------|------------|-----------|
| Yoochoose | 136,456 | 8,827   | 4.45       | 68.86      | 0.68M     |
| Last.FM   | 1,876   | 4,614   | 41.31      | 16.79      | 0.077M    |
| Ta-Feng   | 29,131  | 18,861  | 27.67      | 42.73      | 0.81M     |
| MYbank    | 691,701 | 3,188   | 8.37       | 1818.01    | 5.79M     |

5.1 Experimental Setup

5.1.1 Datasets and Metrics

We evaluate different recommenders based on four publicly available datasets, three of which are public benchmark datasets, i.e., Yoochoose, Last.FM, and Ta-Feng. Specifically, Yoochoose\(^1\) is the competition dataset of Recsys challenge 2015. It contains e-commerce website click within six months and related information. Last.FM\(^2\) dataset contains a set of users from Last.FM online music service, which describes tagging and the music listening details of user. The Ta-Feng\(^3\) dataset is a grocery dataset published by ACM RecSys, it covers goods ranging from food, office supplies to furniture. The fourth dataset is the financial service scenarios dataset MYbank, which is an industrial online recommendation platform in the Ant Group. MYbank dataset describes users' interests and preferences in financial products such as debit, trust, accounting, which contains more than 5.6 million interactions from 691,701 users and brings more challenges compared with the three public datasets. The data statistical status after preprocessing is summarized in Table 2, where Avg.I/user and Avg.I/item denote "average interaction per user" and "average interaction per item", respectively. To filter noisy data, we filter out items that appear less than 3 times, and then remove all user's behaviors less than 3 items on four datasets. After preprocessing, we split user behaviors into three parts, i.e., we randomly pick 80% as training set, 10% as validation set for hyper-parameter tuning, and the remaining part for evaluating the performance of the model. Furthermore, to prevent overfitting, we set the patience argument to be 10 in the early stopping mechanism which denotes how many epochs we want to wait after the last time the validation metrics improved before breaking the training loop.

To fairly compare the generalization performance of each model, we evaluate for each user on his/her performance in the test set by adopting three recognized metrics: HitRate, NDCG and MRR. Here, we choose \( K = 10, 20 \) to show the different metrics for HitRate\( @K \), NDCG\( @K \) and MRR\( @K \).

- HitRate\( @K \) : If one or more element of the label \( y \) is shown in the prediction

---

\(^1\)https://reccsys.acm.org/recsys15/challenge/

\(^2\)http://millionsongdataset.com/lastfm/

\(^3\)http://reccsyswiki.com/wiki/Groceryshoppingdatasets
results $\hat{y}$, we call it a hit. The *HitRate* is calculated as follow:

$$\text{HitRate}@K = \frac{\sum_{s \in S} I(\check{y}^s \cap \hat{y}^s \neq \emptyset)}{|S|},$$

(5.26)

where $|\hat{y}^s| = K$, $I(\ast)$ denotes the indicator function and $\emptyset$ is an empty set. A larger value of *HitRate* reflects the accuracy of the recommendation results.

- **NDCG@K**: Normalized Discounted Cumulative Gain (*NDCG*) is a ranking based metric, which focuses on the order of retrieval results and is calculated in the following way:

$$\text{NDCG}@K = \frac{1}{N_k} \sum_{i=1}^{K} \frac{2^{I(\hat{y}^s \in y^s)} - 1}{\log_2(i + 1)},$$

(5.27)

where $N_k$ is a constant to denote the maximum value of *NDCG@K* given $|\hat{y}^s|$ and $I(\ast)$ denotes an indicator function. A large *NDCG* value reflects a higher the ranking position of the expected item.

- **MRR@K**: Mean Reciprocal Rank (*MRR*) when the r item is not in the higher $K$ position, the reciprocal is set to 0. It is formally given by:

$$\text{MRR}@K = \frac{1}{|S|} \sum_{i=1}^{|S|} \frac{1}{\text{rank}_i},$$

(5.28)

where $\text{rank}_i$ denotes the position of the item in $\hat{y}^s$. **MRR** is a normalize ranking take into account the order of recommendation list $y^s$. A large **MRR** value reflects a higher ranking position of the expected item.

### 5.1.2 Comparison Methods

To demonstrate the performance of HCGR, we consider the following representative methods for performance comparisons:

- **FPMC** [1] - a classical markov-based model, which considers the latest interaction.
- **FOSSIL** [2] - a classical markov-based model, which captures personalized dynamics.
- **GRU4Rec** [3] - a representative RNN-based method for session-based recommendation, it stacks multiple GRU layers for session-parallel mini-batch training.
- **NARM** [18] - a hybrid encoder with attention mechanism to model sequential behaviors in the current session.
- **SASRec** [19] - a self attention-based sequential recommender, which utilizes relatively few actions and considers long-range dependencies.
• **STAMP** [13] - a short term behavior priority attention-based method.

• **SRGNN** [5] - a graph-based recommender modeling session to learn item representations.

• **GC-SAN** [6] - an improved version of SRGNN, which uses a GNN and multilayered self-attention mechanism to compute the sequence level embeddings.

• **FGNN** [23] - a graph-based method, which uses a weight attention network to compute the graph level embeddings.

• **LESSR** [26] - a session-based recommender with GNN, which innovatively utilizes auxiliary graph to generate item representation.

• **HCGGR** - our approach with novel attentive information aggregation, which utilize contrastive loss to optimize the model by considering the distance between positive and negative samples in hyperbolic space.

In this experiment, we set the maximum length of session to be 50, and the embedding dimension to be $d=128$ for all datasets, the initial learning rate is uniformly set to 0.001, the linear schedule decay rate of 0.5 of every 3 epochs and $L_2$ penalty is $3e^{-3}$. All parameters are initialized by Gaussian distribution with mean value of 0 and standard deviation of 0.1. The model cooperates with the Adam optimizer to complete the training.

5.1.3 Data exploration

As discussed in section 2.2, complex data with tree-like structure, i.e., the data obeys a power-law distribution, is effectively explained in the hyperbolic space. Therefore, we check the data distribution used in our experiment to verify the appropriateness of dataset selection. We present the distribution of number of interactions between users and items as illustrated in Figure 5. Power-law distribution is observed in three public datasets and the MYbank dataset. In the user-item interactions, a majority of users interact with items very few times, meanwhile, most items are with few clicks. Such results demonstrate the tree-like structure of our dataset, which is supposed to have better representations in the hyperbolic space for session-based recommendation.

5.2 Overall Performance (RQ1)

The experimental results of all comparison methods in session-based recommendation are presented in Table 3. The best results of each column are highlighted in boldface. As can be observed, HCGGR outperforms the best baselines with more than 4.5% performance improvement on average on all datasets. From the results in Table 3, we can draw the following main findings:
Figure 5: Distributions of user-item interaction, where (a-d) respectively show the results on four datasets. Corresponding to each dataset, the left one illustrates the distribution of the items clicked by each user and the right one plots the distribution of the users clicking each item.

- The RNN-based approaches which capture sequential dependency in a session (i.e., GRU4REC, NARM) remarkably outperform the traditional models that rely on Markov chains (i.e., FPMC, FOSSIL). This phenomenon proves that capturing sequential effects is a key factor for session-based recommendation as user’s session-based behaviors are usually included in a short period and are likely to be temporally dependent.

- The attention-based models (i.e., NARM, SASRec, and STAMP) that involve attention mechanism get higher performance compared with that do not (i.e., GRU4REC) in all evaluation metrics. This is because NARM, STAMP, and SASRec can extract the shift of user interest within sessions and get the main purpose in the current session by incorporating an attention mechanism, which captures personal interest from the long-term memory or just models the user’s current interest from the short-term behaviors. This phenomenon indicates that RNN-based approaches with the assumption that adjacent items in a session have a fixed sequential dependence may generate wrong dependencies, which further results in recommendation bias. This could be alleviated by involving the attention mechanism.

- The GNN-based models (i.e., GC-SAN, FGNN) achieve better performance than RNNs-based models with or without attention mechanism due to the remarkable
Table 3: Performance Comparisons

| Dataset | Metric | FPAM | FOSSIL | GRU4Rec | NARM | SASRec | STAMP | SRGNN | GC-SAN | FGGN | LESSR | HCGR | Improv. |
|---------|--------|------|--------|---------|------|--------|-------|-------|--------|------|-------|------|--------|
| Last.FM | H@10   | 0.0623 | 0.0759 | 0.0749 | 0.0759 | 0.0808 | 0.0735 | 0.0802 | 0.0909 | 0.0949 | 0.0946 | 0.0916 | 0.1071 | 1.04%  |
|         | M@10   | 0.03   | 0.0278 | 0.0288 | 0.0276 | 0.0342 | 0.0343 | 0.0411 | 0.0419 | 0.0414 | 0.0417 | 0.0523 | 14.44% |
|         | N@10   | 0.0452 | 0.0365 | 0.0491 | 0.0488 | 0.0549 | 0.0378 | 0.0354 | 0.0356 | 0.0614 | 0.0669 | 0.0749 | 7.15%  |
|         | H@20   | 0.0623 | 0.0759 | 0.0749 | 0.0759 | 0.0808 | 0.0735 | 0.0802 | 0.0909 | 0.0949 | 0.0946 | 0.0916 | 0.1071 | 1.04%  |
|         | M@20   | 0.03   | 0.0278 | 0.0288 | 0.0276 | 0.0342 | 0.0343 | 0.0411 | 0.0419 | 0.0414 | 0.0417 | 0.0523 | 14.44% |
|         | N@20   | 0.0452 | 0.0365 | 0.0491 | 0.0488 | 0.0549 | 0.0378 | 0.0354 | 0.0356 | 0.0614 | 0.0669 | 0.0749 | 7.15%  |
| Yochoose | H@10   | 0.4093 | 0.4014 | 0.4524 | 0.4635 | 0.4317 | 0.3967 | 0.4341 | 0.4768 | 0.4642 | 0.4753 | 0.4798 | 0.61%  |
|         | M@10   | 0.1603 | 0.1471 | 0.2163 | 0.2207 | 0.1716 | 0.1915 | 0.2204 | 0.1880 | 0.1978 | 0.2241 | 0.2523 | 0.54%  |
|         | N@10   | 0.2424 | 0.2297 | 0.2978 | 0.3032 | 0.26   | 0.2601 | 0.2946 | 0.2844 | 0.2864 | 0.3078 | 0.3162 | 2.73%  |
|         | H@20   | 0.5013 | 0.4902 | 0.5544 | 0.5636 | 0.5391 | 0.4797 | 0.5279 | 0.5895 | 0.5697 | 0.5722 | 0.5938 | 0.73%  |
|         | M@20   | 0.1668 | 0.1533 | 0.2235 | 0.2278 | 0.1791 | 0.1973 | 0.2277 | 0.1959 | 0.2044 | 0.2310 | 0.2325 | 0.65%  |
|         | N@20   | 0.2424 | 0.2297 | 0.2978 | 0.3032 | 0.26   | 0.2601 | 0.2946 | 0.2844 | 0.2864 | 0.3078 | 0.3162 | 2.73%  |
| Ta-Feng | H@10   | 0.0853 | 0.0995 | 0.1091 | 0.1028 | 0.1091 | 0.0961 | 0.1099 | 0.1099 | 0.1099 | 0.1115 | 0.1134 | 1.79%  |
|         | M@10   | 0.04   | 0.0344 | 0.0456 | 0.0438 | 0.0447 | 0.0404 | 0.0435 | 0.0444 | 0.0396 | 0.0378 | 0.0487 | 28.84% |
|         | N@10   | 0.058  | 0.0569 | 0.0709 | 0.067  | 0.07   | 0.0592 | 0.0635 | 0.0699 | 0.0644 | 0.0673 | 0.0733 | 8.92%  |
| MVbank  | H@10   | 0.5136 | 0.4521 | 0.5647 | 0.5459 | 0.5232 | 0.5542 | 0.5522 | 0.5505 | 0.5612 | 0.5652 | 0.5773 | 2.21%  |
|         | M@10   | 0.2899 | 0.2623 | 0.3255 | 0.3185 | 0.3016 | 0.3167 | 0.3173 | 0.3164 | 0.3299 | 0.3104 | 0.3373 | 2.24%  |
|         | N@10   | 0.3429 | 0.3073 | 0.3822 | 0.3724 | 0.354  | 0.373  | 0.373  | 0.3735 | 0.3835 | 0.3841 | 0.3901 | 1.72%  |
|         | H@20   | 0.6185 | 0.5438 | 0.6647 | 0.6453 | 0.6261 | 0.6603 | 0.6544 | 0.6581 | 0.6654 | 0.6616 | 0.6713 | 0.43%  |
|         | M@20   | 0.2972 | 0.2666 | 0.3324 | 0.3254 | 0.3087 | 0.3241 | 0.3245 | 0.3417 | 0.3445 | 0.3424 | 0.3578 | 3.86%  |
|         | N@20   | 0.3694 | 0.3304 | 0.4075 | 0.3976 | 0.3799 | 0.3998 | 0.3998 | 0.4025 | 0.4103 | 0.4206 | 0.4135 | 0.78%  |

* Relative improvements are calculated by comparing with the second best performance.

capacity of graph neural networks to capture complex interaction of user behaviors and describe the coherence of items in a session, which are ignored by RNNs-based models and such ignorance leads to overfitting in RNNs-based models.

- Our proposed HCGR consistently outperforms all the comparison models on all datasets. Compared with FGNN and LESSR, our model involves an advanced hyperbolic learning component to more effectively capture the coherence and hierarchy representations of the user behaviors within the Lorentz hyperbolic space, which ensures the correctness of the necessary representations’ transformation. Furthermore, we use a novel graph message propagation mechanism with adaptive hyperbolic attention calculation to model user’s preferences in session behavior sequences. In addition, we introduce contrastive learning to optimize the model by considering the distance between positive and negative samples in hyperbolic space, which can help learn better item representations.

5.3 Ablation Study (RQ2)

5.3.1 Effect of Lorentz Transformation

To demonstrate the effectiveness of the proposed hyperbolic learning framework for session-based recommendation, we conduct the ablation experiments by combining the Lorentz transformation with several baseline Euclidean SBR models, including FPMC, GRU4Rec, SASRec, and SRGNN. Besides, we also compare the performance of HCGR with ECGR (Euclidean Contrastive Graph Representation) by removing the Lorentz
transformation from the hyperbolic contrastive graph representation learning framework shown in Figure 3. The experimental results are shown in Table 4. The postfix *lorentz* means the corresponding model is combined with hyperbolic learning to extract the hierarchy information contained within the SBR datasets. From Table 4, we can draw the following conclusions:

- The performance of all models improves significantly on the four datasets when combining the Lorentz transformation with the Euclidean SBR models, which demonstrates that the hierarchy information from the power-law like session-based recommendation data is essential for predicting the user behavior, while such information is just ignored by the traditional SBR models built upon Euclidean space. Furthermore, the improvement of Markov-based method (i.e., FPMC) and attention-based method (i.e., SASRec) is more obvious than that of RNN-based (i.e., GRU4Rec) and GNN-based (i.e., SRGNN) method.

- Our proposed hyperbolic contrastive graph representation learning method HCGR achieves the best results over all comparison models with or without Lorentz transformation, but the performance of ECGR drops evidently when replacing the Lorentz transformation with Euclidean transformation on all datasets. Besides, we find that the ECGR outperforms most baseline SBR models coupling with Lorentz transformation, which indicates the advantage of the proposed contrastive graph representation learning method.

![Figure 6: The performance with different attention mechanisms on four datasets](image)

### 5.3.2 Effect of Graph Aggregation Method

To further investigate the advantage of the proposed adaptive hyperbolic graph aggregation method that utilizes multi-hop adjacent information, we conduct an ablation
Table 4: Performance comparison of models with or without Lorentz on the Last.FM, Ta-Feng, Yoochoose and MYbank datasets.

| Dataset | Metric | FPIN | FPIN_{l} | GRU4Rec | GRU4Rec_{l} | SASRec | SASRec_{l} | SRGNN | SRGNN_{l} | ECR | HCGR |
|---------|--------|------|----------|--------|------------|--------|------------|-------|-----------|-----|------|
| Last.FM | H010  | 0.0623 | 0.0684 | 0.0759 | 0.0771 | 0.0886 | 0.0868 | 0.0902 | 0.0910 | 0.1063 | 0.1071 |
|         | N010  | 0.0376 | 0.0395 | 0.0398 | 0.0400 | 0.0452 | 0.0469 | 0.0484 | 0.0495 | 0.0612 | 0.0651 |
|         | M010  | 0.0300 | 0.0306 | 0.0288 | 0.0288 | 0.0342 | 0.0347 | 0.0411 | 0.0417 | 0.0483 | 0.0523 |
|         | H020  | 0.1149 | 0.0837 | 0.1509 | 0.1526 | 0.1494 | 0.1518 | 0.1262 | 0.1349 | 0.1294 | 0.1388 |
|         | N020  | 0.0452 | 0.0449 | 0.0491 | 0.0509 | 0.0549 | 0.0557 | 0.0354 | 0.0588 | 0.0718 | 0.0749 |
|         | M020  | 0.0321 | 0.0320 | 0.0312 | 0.0317 | 0.0370 | 0.0371 | 0.0431 | 0.0425 | 0.0516 | 0.0541 |

| Yoohoose | H010  | 0.0853 | 0.0863 | 0.1091 | 0.1101 | 0.1100 | 0.1110 | 0.0940 | 0.0966 | 0.1068 | 0.1134 |
|          | N010  | 0.0506 | 0.0555 | 0.0604 | 0.0617 | 0.0598 | 0.0603 | 0.0554 | 0.0563 | 0.0579 | 0.0539 |
|          | M010  | 0.0400 | 0.0392 | 0.0456 | 0.0470 | 0.0447 | 0.0448 | 0.0435 | 0.0440 | 0.0433 | 0.0487 |
|          | H020  | 0.5013 | 0.5110 | 0.5544 | 0.5577 | 0.5391 | 0.5618 | 0.5279 | 0.5442 | 0.1432 | 0.1507 |
|          | N020  | 0.0580 | 0.0498 | 0.0709 | 0.0724 | 0.0700 | 0.0706 | 0.0635 | 0.0659 | 0.0676 | 0.0733 |
|          | M020  | 0.0420 | 0.0403 | 0.0485 | 0.0499 | 0.0475 | 0.0477 | 0.0458 | 0.0466 | 0.0456 | 0.0512 |

| MYbank   | H010  | 0.4093 | 0.4149 | 0.4524 | 0.4566 | 0.4317 | 0.4463 | 0.4341 | 0.4469 | 0.4651 | 0.4798 |
|          | N010  | 0.2190 | 0.2243 | 0.2719 | 0.2743 | 0.2328 | 0.2359 | 0.2709 | 0.2675 | 0.2572 | 0.2898 |
|          | M010  | 0.1603 | 0.1657 | 0.2163 | 0.2181 | 0.1716 | 0.1715 | 0.2204 | 0.2221 | 0.1933 | 0.2252 |
|          | H020  | 0.6185 | 0.6410 | 0.6647 | 0.6685 | 0.6261 | 0.6236 | 0.6544 | 0.6649 | 0.5681 | 0.5938 |
|          | N020  | 0.2424 | 0.2488 | 0.2978 | 0.2999 | 0.2600 | 0.2652 | 0.2946 | 0.2952 | 0.2834 | 0.3162 |
|          | M020  | 0.1668 | 0.1724 | 0.2235 | 0.2252 | 0.1791 | 0.1796 | 0.2270 | 0.2253 | 0.2005 | 0.2325 |

Table 5: The performance with different optimization functions on Last.FM, Yoochoose datasets.

| Method   | Last.FM | Yoochoose |
|----------|---------|-----------|
|          | H010    | M010      | N010      | H020    | M020      | N020      | H010    | M010      | N010      | H020    | M020      | N020      |
| HCGR_CE  | 0.11    | 0.0508    | 0.0645    | 0.1358   | 0.052    | 0.0725    | 0.4751    | 0.2182   | 0.2861    | 0.3827    | 0.2261    | 0.3144    |
| HCGR     | 0.1071  | 0.0523    | 0.0651    | 0.1388   | 0.0541   | 0.0749    | 0.4798    | 0.2252   | 0.2988    | 0.5938    | 0.2325    | 0.3162    |
| Improv   | -2.63%  | 2.92%     | 0.93%     | 2.20%    | 4.03%    | 3.32%     | 0.99%     | 3.21%    | 1.29%     | 1.90%     | 2.83%     | 0.57%     |

Table 6: The performance with different optimization functions on Ta-Feng, MYbank datasets.

| Method   | Ta-Feng | MYbank |
|----------|---------|--------|
|          | H010    | M010   | N010   | H020   | M020   | N020   | H010    | M010   | N010   | H020   | M020   | N020   |
| HCGR_CE  | 0.1154  | 0.0453  | 0.0523  | 0.1532  | 0.048   | 0.0716  | 0.5743  | 0.3321  | 0.3876  | 0.665  | 0.3528  | 0.4123  |
| HCGR     | 0.1134  | 0.0487  | 0.0539  | 0.1507  | 0.0512  | 0.0733  | 0.5722  | 0.3373  | 0.3901  | 0.6719  | 0.3578  | 0.4135  |
| Improv   | -1.70%  | 7.5%    | 3.06%   | -0.016% | 6.67%   | 2.37%   | 0.50%   | 1.56%   | 0.64%   | 0.95%   | 1.47%   | 0.29%   |
study by comparing different graph aggregation information approach within the framework of hyperbolic contrastive representation learning on four datasets. HCGR_GCN refers to a model that the traditional spectrum-based graph convolution method is used to transport messages among adjacent neighbors, while HCGR_GAT refers to a model that the graph attention-based method is used to aggregate adjacent information. The experimental results are shown in Figure 6. we can draw the following conclusions:

- The inductive attention-based graph convolution model(i.e., HCGR_GAT) remarkably outperforms the transductive spectrum-based graph model(i.e., HCGR_GCN) on all datasets, which indicates that treating neighbours deferentially and flexibly is essential to filter noisy information during message aggregation.

- No surprisingly, HCGR with our proposed multi-hop adjacent information aggregation method achieves the best performance. Comparing to GAT, the main improvement of our proposed aggregation method relying on multi-hop aggregated message during graph node representation optimization is fully used, which indicates that low-order and high-order mutual graph information are both critical for final prediction. Such low-order mutual information is just ignored by GAT-like models.

5.3.3 Effects of Contrastive Ranking Loss

Diversity has become an important evaluation indices in recommendation scenario. In order to investigate the effectiveness of contrastive learning on the performance of our proposed hyperbolic graph representation learning framework, we conduct an experimental analysis by removing the contrastive ranking loss. To specific, HCGR_CE means that contrastive ranking loss is removed from Eq (4.25) while keeping other settings same as HCGR. The experimental resutls are shown in Table 5 and Table 6, we can draw the following observations:

- In all datasets, the contrastive ranking loss optimization model HCGR outperforms the cross-entropy loss optimization model(HCGR_CE) as regard to ranking evaluation metrics($MRR$, $NDCG$), which indicates that the contrastive ranking loss can distinguish the subtle distinction between items within sessions and improve recommendation diversity.

- With regard to the accuracy of recommendation results, there is no obvious difference between the performance of the two models, which indicates that our contrastive ranking loss can improve the ranking performances without losing recommendation accuracy.

5.3.4 Effects Of Embedding Size

We explore the impact of embedding size $d$ on several evaluation indices as such size significantly affect the representation ability. We conduct the experiment on Yoochoose
and MYbank datasets as their generalization and representation. The results are plotted in Figure 7-8. We have the following observations:

- HCGR outperforms all comparison SBR methods with most embedding sizes in all indices. Especially for small embedding size, such as 32, our model still can achieve better and robust results on these indices, which indicates introducing hyperbolic transformation can capture latent hierarchy property and boosts model performance.

- It is also observed that a proper embedding size is essential for graph node representation. When the embedding size is too small, it can’t fully express node information and result in poor performance. On the opposite, a large embedding size may induce overfitting on the dataset.

![Figure 7](image)

Figure 7: The performance of various models with different embedding size on Yoochoose datasets

5.4 Case Study (RQ3)

5.4.1 Representation Analysis

We set up the case study to explore whether our model can learn the hierarchical structure of user behavior. Whether the hierarchical structure in the data can be fully learned will affect the performance of the model, and this kind of hierarchical structure can be reflected by calculating the distance between the representation and the origin. HCGR, ECGR are calculated in two different geometries, we use gyrovector space distance and tangent distance respectively to calculate the distance from the target point to the origin. We set up three boundaries in Euclidean space and Lorentz space, and divide the
representation into four regions according to their distance from the origin. For example, the item of region 1 is the closest to the origin, whereas the item of region 4 is the farthest from the origin. To intuitively reflect the different popularity of items in different regions, we count the interaction times of nodes in all regions of the four datasets. We visualize the statistics as Figure 9. From the results, it can be seen that the average number of interactions of items from region 1 to region 4 has decreased, which shows that both of our approaches ECGR and HCGR can model the hierarchical structure of session behavior. In addition, in all datasets, the average number of interactions of items with HCGR in region 1 is higher than that with ECGR, while the average number of interactions of items with ECGR in region 3 and 4 is higher than that with HCGR. Compared with ECGR, HCGR can better distinguish the items with different popularity and learn the hierarchical structure, which indicates that hyperbolic space is more suitable for embedding hierarchical data than Euclidean space for session-based recommendation tasks.

5.4.2 Attention Analysis

Taking advantage of the attention mechanism, we visualize the attention weight between user behaviors as shown in Figure 10, which reflects the different influences within the same session on the two models (HCGR, ECGR). We randomly select three different sessions of length 10($S_{s1}$), 30($S_{m1}$) and 50($S_{l1}$) respectively from MYbank dataset(test set). For a same session in the heatmap, the above one is the attention weight between the related items and the next item user most likely to click modelled by ECGR, while the following one is the corresponding attention weight modelled by HCGR. From the heatmap, we discover that not all the behaviours in the same session equally contributing
Figure 9: Hierarchical structure analysis of four datasets in tangent space and hyperbolic space

Figure 10: Visualization of average attention weight of behavior at different locations.

during the generation of the recommendation. In addition, we also find that the attention weight of HCGGR for session behavior is higher than that of ECGR in many key positions. Specifically, HCGGR will give higher scores with the increase of the scores given by ECGR, and HCGGR can better distinguishes the item importance. It proves that hyperbolic space can better represent the hierarchical structure of data, thus making the attention mechanism capable of adaptively measuring the influence of session behavior.

5.5 Embedding Analysis and Visualization

We visualize the item embeddings in 2-dimension and 3-dimension on Last.FM, Yoochoose, Ta-Feng and MYbank datasets repectively according to Figure 11. (a) - (d). The popularity of the items is represented according to the different colors, decreasing with the color from red to green. Before training, we randomly initialize all the item embeddings. As shown in Figure 11, it is obvious that item embeddings present a hierarchical structure based on item popularity after training. On the Last.FM dataset, we can observe such a clear hierarchical representation, with the most popular items in the center and unpopular ones stay away from the center of projection space. Similar results can also be obtained on other datasets.
Figure 11: The embedding visualization, where (a-d) show the 2-d (in top row) and 3-d visualization (in bottom row) on four datasets respectively.

6 Conclusion

The GNN-based model can not capture the hierarchical information effectively, which regularly appeared in recommendation scenarios. Enlightened by the powerful representation of non-Euclidean geometry which is proved to be able to reduce the distortion of embedding data onto power-law distribution, we proposed a hyperbolic contrastive graph recommender (HCGR), utilizing Lorentz hyperbolic space for item embeddings preserving their coherent and hierarchical properties. We design a novel hyperbolic graph message propagation mechanism due to the discrepancy between Euclidean and hyperbolic space during information passing. In addition, we introduce contrastive learning to enhance model performance by optimizing the distance between positive and negative samples of hyperbolic space, considering that distance in hyperbolic space can’t be expressed well by traditional loss, such as CE, BPR loss. For future work, we will extend our method to the sequential recommendation which involves user profile and more item features. Besides, we will learn a parsimonious representation of symbolic data by embedding the dataset into spherical or product space and optimize the process of matrix multiplication in non-Euclidean geometry.

References

[1] Steffen Rendle, Christoph Freudenthaler, and Lars Schmidt-Thieme. Factorizing personalized markov chains for next-basket recommendation. In Proceedings of the 19th international conference on World wide web, pages 811–820, 2010.

[2] Ruining He and Julian McAuley. Fusing similarity models with markov chains for sparse sequential recommendation. In 2016 IEEE 16th International Conference on
[3] Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. Session-based recommendations with recurrent neural networks. *arXiv preprint arXiv:1511.06939*, 2015.

[4] Lin Zheng, Naicheng Guo, Weihao Chen, Jin Yu, and Dazhi Jiang. Sentiment-guided sequential recommendation. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1957–1960, 2020.

[5] Shu Wu, Yuyuan Tang, Yanqiao Zhu, Liang Wang, Xing Xie, and Tieniu Tan. Session-based recommendation with graph neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 346–353, 2019.

[6] Chengfeng Xu, Pengpeng Zhao, Yanchi Liu, Victor S Sheng, Jiajie Xu, Fuzhen Zhuang, Junhua Fang, and Xiaofang Zhou. Graph contextualized self-attention network for session-based recommendation. In *IJCAI*, volume 19, pages 3940–3946, 2019.

[7] Erzsébet Ravasz and Albert-László Barabási. Hierarchical organization in complex networks. *Physical review E*, 67(2):026112, 2003.

[8] Dmitri Krioukov, Fragkiskos Papadopoulos, Maksim Kitsak, Amin Vahdat, and Marián Boguná. Hyperbolic geometry of complex networks. *Physical Review E*, 82(3):036106, 2010.

[9] Fragkiskos Papadopoulos, Maksim Kitsak, M Ángeles Serrano, Marián Boguná, and Dmitri Krioukov. Popularity versus similarity in growing networks. *Nature*, 489(7417):537–540, 2012.

[10] Jianing Sun, Zhaoyue Cheng, Saba Zuberi, Felipe Pérez, and Maksims Volkovs. Hgcf: Hyperbolic graph convolution networks for collaborative filtering. In *Proceedings of the Web Conference 2021*, pages 593–601, 2021.

[11] M. Asaduzzaman, S. Catterall, J. Hubisz, Roice Nelson, and J. Unmuth-Yockey. Holography on tessellations of hyperbolic space. *Physical Review D*, 102, 2020.

[12] Pengfei Wang, Jiafeng Guo, Yanyan Lan, Jun Xu, Shengxian Wan, and Xueqi Cheng. Learning hierarchical representation model for nextbasket recommendation. In *Proceedings of the 38th International ACM SIGIR conference on Research and Development in Information Retrieval*, pages 403–412, 2015.

[13] Qiao Liu, Yifu Zeng, Refnoe Mokhosi, and Haibin Zhang. Stamp: short-term attention/memory priority model for session-based recommendation. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1831–1839, 2018.
[14] Le Wu, Peijie Sun, Yanjie Fu, Richang Hong, Xiting Wang, and Meng Wang. A neural influence diffusion model for social recommendation. In Proceedings of the 42nd international ACM SIGIR conference on research and development in information retrieval, pages 235–244, 2019.

[15] Qiang Cui, Shu Wu, Qiang Liu, Wen Zhong, and Liang Wang. Mv-rnn: A multi-view recurrent neural network for sequential recommendation. IEEE Transactions on Knowledge and Data Engineering, 32(2):317–331, 2018.

[16] Lin Zheng, Naicheng Guo, Jin Yu, and Dazhi Jiang. Memory reorganization: a symmetric memory network for reorganizing neighbors and topics to complete rating prediction. IEEE Access, 8:81876–81886, 2020.

[17] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. arXiv preprint arXiv:1706.03762, 2017.

[18] Jing Li, Pengjie Ren, Zhumin Chen, Zhaochun Ren, Tao Lian, and Jun Ma. Neural attentive session-based recommendation. In Proceedings of the 2017 ACM on Conference on Information and Knowledge Management, pages 1419–1428, 2017.

[19] Wang-Cheng Kang and Julian McAuley. Self-attentive sequential recommendation. In 2018 IEEE International Conference on Data Mining (ICDM), pages 197–206. IEEE, 2018.

[20] Tingting Zhang, Pengpeng Zhao, Yanchi Liu, Victor S Sheng, Jiajie Xu, Deqing Wang, Guanfeng Liu, and Xiaofang Zhou. Feature-level deeper self-attention network for sequential recommendation. In IJCAI, pages 4320–4326, 2019.

[21] Jin Huang, Wayne Xin Zhao, Hongjian Dou, Ji-Rong Wen, and Edward Y Chang. Improving sequential recommendation with knowledge-enhanced memory networks. In The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval, pages 505–514, 2018.

[22] Shu Wu, Mengqi Zhang, Xin Jiang, Xu Ke, and Liang Wang. Personalizing graph neural networks with attention mechanism for session-based recommendation. arXiv preprint arXiv:1910.08887, 2019.

[23] Ruihong Qiu, Jingjing Li, Zi Huang, and Hongzhi Yin. Rethinking the item order in session-based recommendation with graph neural networks. In Proceedings of the 28th ACM International Conference on Information and Knowledge Management, pages 579–588, 2019.

[24] Chen Ma, Liheng Ma, Yingxue Zhang, Jianing Sun, Xue Liu, and Mark Coates. Memory augmented graph neural networks for sequential recommendation. In Proceedings of the AAAI Conference on Artificial Intelligence, volume 34, pages 5045–5052, 2020.
[25] Wen Wang, Wei Zhang, Shukai Liu, Qi Liu, Bo Zhang, Leyu Lin, and Hongyuan Zha. Beyond clicks: Modeling multi-relational item graph for session-based target behavior prediction. In Proceedings of The Web Conference 2020, pages 3056–3062, 2020.

[26] Tianwen Chen and Raymond Chi-Wing Wong. Handling information loss of graph neural networks for session-based recommendation. In Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pages 1172–1180, 2020.

[27] Michael M. Bronstein, Joan Bruna, Yann LeCun, Arthur Szlam, and Pierre Vandergheynst. Geometric deep learning: Going beyond euclidean data. IEEE Signal Processing Magazine, 34(4):18–42, 2017.

[28] Robert P. W. Duin and Elzbieta Pekalska. Non-euclidean dissimilarities: Causes and informativeness. In Structural, Syntactic, and Statistical Pattern Recognition, pages 324–333, Berlin, Heidelberg, 2010. Springer Berlin Heidelberg.

[29] Bo Yang, Yu Lei, Jiming Liu, and Wenjie Li. Social collaborative filtering by trust. IEEE Transactions on Pattern Analysis and Machine Intelligence, 39(8):1633–1647, 2017.

[30] Brian J. McGill, Brian J. Enquist, Evan Weiher, and Mark Westoby. Rebuilding community ecology from functional traits. Trends in Ecology and Evolution, 21(4):178–185, 2006.

[31] Alexandru Tifrea, Gary Bécigneul, and Octavian-Eugen Ganea. Poincare glove: Hyperbolic word embeddings. In 7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019, 2019.

[32] Maximilian Nickel and Douwe Kiela. Poincaré embeddings for learning hierarchical representations. In Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA, pages 6338–6347, 2017.

[33] Qi Liu, Maximilian Nickel, and Douwe Kiela. Hyperbolic graph neural networks. In Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada, pages 8228–8239, 2019.

[34] Ines Chami, Zhitao Ying, Christopher Ré, and Jure Leskovec. Hyperbolic graph convolutional neural networks. In Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada, pages 4869–4880, 2019.
[35] Boli Chen, Xin Huang, Lin Xiao, Zixin Cai, and Liping Jing. Hyperbolic interaction model for hierarchical multi-label classification. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 7496–7503, 2020.

[36] Nurendra Choudhary, Nikhil Rao, Sumeet Katariya, Karthik Subbian, and Chandan K. Reddy. Self-supervised hyperboloid representations from logical queries over knowledge graphs. In *WWW ’21: The Web Conference 2021, Virtual Event / Ljubljana, Slovenia, April 19-23, 2021*, pages 1373–1384, 2021.

[37] Shanshan Feng, Lucas Vinh Tran, Gao Cong, Lisi Chen, Jing Li, and Fan Li. 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, SIGIR 2020, Virtual Event, China, July 25-30, 2020*, pages 1429–1438, 2020.

[38] Benjamin Paul Chamberlain, Stephen R. Hardwick, David R. Wardrope, Fabon Dzogang, Fabio Daolio, and Saúl Vargas. Scalable hyperbolic recommender systems. *arXiv preprint arXiv: 1902.08648*, 2019.

[39] Kyuyong Shin, Young-Jin Park, Kyung-Min Kim, and Sunyoung Kwon. Multimanifold learning for large-scale targeted advertising system. *arXiv preprint arXiv: 2007.02334*, 2020.

[40] Chen Ma, Liheng Ma, Yingxue Zhang, Haolun Wu, Xue Liu, and Mark Coates. Knowledge-enhanced top-k recommendation in poincaré ball. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*, pages 4285–4293, 2021.

[41] Anchen Li and Bo Yang. HSR: hyperbolic social recommender. *arXiv preprint arXiv: 2102.09389, abs/2102.09389*, 2021.

[42] Octavian-Eugen Ganea, Gary Bécigneul, and Thomas Hofmann. Hyperbolic neural networks. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pages 5350–5360, 2018.

[43] Hao Wang, Defu Lian, Hanghang Tong, Qi Liu, Zhenya Huang, and Enhong Chen. Hypersorec: Exploiting hyperbolic user and item representations with multiple aspects for social-aware recommendation. 2021.
[44] Çaglar 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. Hyperbolic attention networks. In 7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019.

[45] Maximilian Nickel and Douwe Kiela. Learning continuous hierarchies in the lorentz model of hyperbolic geometry. In Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018, volume 80, pages 3776–3785, 2018.

[46] J. G. Ratcliffe, A. Ramsay, and R. D. Richtmyer. Introduction to hyperbolic geometry. American Mathematical Monthly, 103(2):185, 1995.

[47] Xiangnan He, Kuan Deng, Xiang Wang, Yan Li, Yongdong Zhang, and Meng Wang. Lightgcn: Simplifying and powering graph convolution network for recommendation. In Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval, pages 639–648, 2020.

[48] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. arXiv preprint arXiv:1609.02907, 2016.

[49] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. arXiv preprint arXiv:1710.10903, 2017.