Intermediary Does Matter! Conditional Proximity Embedding for Recommender Systems

Kachun Lo
Tohoku University
Sendai, Japan
argent.lo.hk@gmail.com

Tsukasa Ishigaki
Tohoku University
Sendai, Japan
isgk@tohoku.ac.jp

***Abstract*—Recently user-item bipartite graph has been studied and exploited to learn representative embedding for personalized recommendation. Graph-based approaches endow a model with the capability of capturing complex information flowing from indirect neighbors to source node. Despite the great success achieved, several limitations still exist. Treating indirect nodes as direct nodes, regardless of the intermediate nodes, is one limitation. This introduces massive noises, since direct and indirect neighbors are intrinsically distinct for a source node. Moreover, other than the necessary embedding parameters, most works also rely heavily on "extra parameters", resulting in verbose training processes and burdensome models with proneness to overfitting. Finally, the attention mechanism is mostly used to local key neighbors, while attention on feature level is rarely utilized.**

In this work, we propose a concept of "conditional proximity", which emphasizes the decisive role of intermediate nodes playing in information propagation and helps filter or enhance signals conditionally. The conditional proximity is measured by a feature-level attention mechanism, which guides the information to propagate through pivotal feature channels of embeddings for learning meaningful latent representations. Without bringing in any "extra parameters", we summarize our ideas and develop a light graph-based framework for recommendation "*HANABI*" (HNB). Comprehensive experiments on 4 public datasets have been conducted to show HNB's superiority over state-of-the-art models.

I. Introduction

Recommender systems (RSs) has been playing a non-negligible role nowadays in surprisingly many fields [1], [2], e.g. e-commerce, social media, search engine. RSs mainly studies how to connect entities within the system effectively and efficiently. Moreover, in real applications, overwhelmingly growing amount of data also require a reliable RS to be light in weight and simple in construction, such that fast inference can be achieved and building and updating can be done on a more frequent basis.

To estimate users' preferences for items, collaborative filtering (CF) is the most popular method among the literature. CF predicts preferences based on the assumption that similar users show resemblant behaviors and, as a result, would interact similar items. In order to capture latent user-item relations, most models embed both users and items as embedding vectors of same dimensions in a common space. Among many classical methods, matrix factorization (MF) utilizes inner products of users and items embeddings to infer

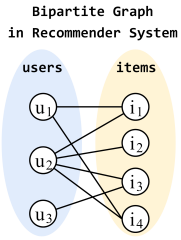


Fig. 1: Illustration of Bipartite Graph.

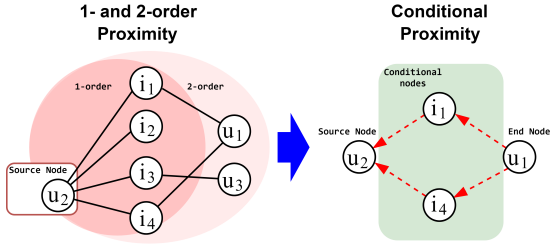


Fig. 2: Illustration of how indirect 2-order information flow through intermediate nodes to source node.

observed interactions. Bayesian Personalized Ranking (BPR) frames the regression problem in MF as a binary classification problem for implicit interactions, where users' preferences are measured by a probabilistic pairwise ranking function. Since deep learning showed its superior capability of capturing complex patterns [3], researchers have also adopted the idea in RSs and made CF models deeper. Neural Collaborative Filtering (NCF) extends CF by stacking multiple layers with nonlinear activation functions. Collaborative Memory Network (CMN) combines CF with memory network, which helps memorize users' subtle preferences.

These models are then further improved by graph-based frameworks, which enrich the sparse RSs with indirect information. DeepWalk, LINE and Node2Vec have successfully caught high-order indirect relations and generated representative nodes' embeddings through random surfer [4], [5], [6]. In the domain of RSs, user-item interactions are typically constructed as a bipartite graph first, where users and items have no direct edge with their own kind, as shown in Figure 1. Then, pairs of neighbor nodes would be collected by random walk sampling as training examples. An example of 1 and

2-order proximity can be found in Figure 2 (left).

Bipartite Network Embedding (BINE) is a graph embedding model for general purposes, featured by emphasizing homogeneous relations between user-user and item-item [7]. HOP-Rec is designed for RSs, which improves a basic BPR model by adding indirect higher-order neighbors as training examples [8]. CSE, inspired by HOP-Rec and LINE, adopts the idea of auxiliary context embedding which helps to simultaneously model higher-order user-user, item-item, and user-item similarity [9]. More recently, NGCF takes into account the node-to-node information propagation and explicitly model this process in a recursive style over layers [10]. Multi-GCCF takes a step forward by not only modelling the information propagation but also maintaining the intrinsic difference between users and items [11].

Graph-based methods empower CF to capture indirect high-order relations. There, however, still exist several **limitations** in most current works (details in section II-B):

1) Treating "indirect high-order proximity" in the same way as "direct proximity" and ignoring the intermediate nodes.
2) Weighty "extra parameters" are introduced in training or inference in most graph-based CF models.
3) Compared to the popular node-level attention, feature-level attention has not been utilized in graph embedding studys.

To alleviate the aforementioned problems, we propose our light graph embedding model for RSs, "***HANABI***" (HNB) [1]. HNB not only gets rid of additional parameters both in training and inference, but also achieves superior performance by explicitly modelling intermediate nodes and conditionally measuring high-order proximity with feature-level attention.

Specifically, HNB first samples a pair of direct neighbors and treats them as two source nodes. Then, starting from source nodes, multiple indirect neighbors are sampled by an extended random walk for training. Next, especially different from other works, HNB models high-order information propagation **conditionally**. The process that information has to flow through intermediate node(s) before arriving to source nodes is modelled as "conditional proximity". We design a feature-level attention mechanism to capture these conditional dependencies.

To sum up, the main contributions of our work are as follows:

- We propose HNB, a light graph-based CF model for recommendation, which gets rid of additional parameters, achieving fast inference, allowing easy construction and being less prone to overfitting.
- We design "conditional proximity" to model the decisive process that, before arriving to source nodes, information signals are filtered or enhanced by intermediate nodes. Feature-level attention is used to capture the conditional dependencies.
- Extensive experiments conducted on 4 datasets examine the effectiveness of the conditional proximity and reveal the superiority of HNB over state-of-the-art models.

[1]The model is named after the graph structure which looks like a firework and "HANABI" (HNB) is "firework" in Japanese.

TABLE I: List of Notations.

Symbol	Definition
U, I	User, Item set
u, i	A specific user, item
N_j	Set of direct neighbors of node j
θ_j, Θ	Embedding of node j; Entire embedding matrix
d	The predefined embedding size
k	The walk length of the random walk sampling
$s, \mathbf{m}, e$	Source, intermediate node(s) and end node of an information flowing path
e^+, e^-	Positive, negative end node
η	Number of high-order neighbors to sample
w_k	Length-aware decay factor
w_{od_j}	Outdegree-aware factor
λ	L_2 regularization hyperparameter

II. Preliminary

In this section, we first summarize notation and introduce the concept of "conditional proximity". Then, the detailed discussion of limitations of current works will also be covered.

Notation we use in this paper is summarized in Table I.

A. Information Flowing in Graph

Graph-based method captures similarity between nodes by measuring the quality and quantity of information flows (or edges). For example, as shown in Figure 2(left), for a source node u_2, all items $i_{1\sim4}$ are equally important, because they are all direct neighbors and the edge weights, by default in most RSs, are the same. The 2-order neighbors, u_1 and u_3, are not as close to u_2 as $i_{1\sim4}$, since they take two steps to reach u_2. Furthermore, given that u_1 has more paths to reach u_2 than u_3, u_1 is more similar to u_2 than u_3.

Conditional Proximity. Although proximity can be partially obtained by measuring the length and number of paths, such simple approach is not enough to capture high-order complex relations, because indirect neighbors have intrinsically different way of propagating information from direct (1-order) neighbors. We argue that, intermediate node(s), which connect source and end nodes, play a decisive role in high-order proximity. Illustrated in Figure 2(right), a 2-order neighbor u_1 has 2 paths to the source node u_2, where each path carries fundamentally different information depending on the intermediate node, i_1 or i_4.

An **intuitive example** of how information propagates distinctly through 2 paths, $[e \to m_1 \to s]$ and $[e \to m_2 \to s]$, is as follows. Suppose a source user (node s) has two friends, a scientist (m_1) and an artist (m_2), and they both recommend a book (e) to s. User s would probably be very interested because s gets multiple suggestions of the same thing e from two 1-order close friends. However, more interestingly, user s might perceive the two suggestion messages very ***distinctively***. While the scientist m_1 might give suggestion for educational purpose, the artist m_2 might recommend e because of its

aesthetic value. We, therefore, refer to the information propagation being conditional on intermediate node as **"conditional proximity"** and intermediate node as **"conditional node"**.

B. Limitations in Current Works

The limitations of current graph-based CF models are discussed in this subsection.

Limitation 1. Most graph-based models treat "high-order proximity" in the same way as "direct proximity" by maximizing inner products of pairs of neighbors' embeddings, regardless of their different orders. Such approach completely ignores intermediate node(s). Since direct and indirect interactions are intrinsically different, treating them the same way would introduce massive noises, resulting in sub-optimal solution. We argue that, especially for indirect interactions, the decisive role of intermediate node(s) should be modelled explicitly.

Limitation 2. To improve performance, most graph-based models employ "extra parameters/weights" in either training or both training and inference. To clarify, for graph embedding model, we refer to any parameters other than the "embedding matrix" as "extra weights". "Extra weights" include "context embedding", "weights for linear transformation", etc.

First, "extra weights" are rather burdensome and often imply complex computations. A robust RSs should be light in weight for fast inference and fast updating.

Second, heavier and deeper models are prone to overfitting, especially in extremely sparse settings such as RSs. Consequently, deep models exhibiting powerful performance in experimental setting might not generalize well in real applications.

Finally, it's truth that training techniques like drop-out and early-stopping can effectively alleviate overfitting. These, however, require more laboring efforts on tuning hyperparameters, which again is at cost of human efforts and computer resources [12], [13].

Limitation 3. Feature-level attentive mechanism has not been utilized in graph-based RSs studies. Although many works [14], [15], [16], [17] have employed "node-level" attention to locate key neighbors for source nodes, the attention on latent feature channels of embedding has not been explicitly exploited. We argue that, due to the extreme sparsity and the heterogeneity of nodes in RSs, a more fine-grained "feature-level" attention can better handle propagation of information signals than "node-level" attention.

III. Our Model

In this section, we introduce our proposed HNB in detail.

A. Node Embedding Layer

We first model both users and items embeddings in a common space as an embedding matrix, whose parameters are all learnable, as other works [8], [9], [10], [11]. The overall embedding matrix $\mathbf{\Theta}$ takes the following form:

$$\mathbf{\Theta} = [\theta_{u_1}, \cdots, \theta_{u_{|U|}}, \ \ \theta_{i_1}, \cdots, \theta_{i_{|I|}}], \quad \mathbf{\Theta} \in R^{(|U|+|I|)\times d} \tag{1}$$

Then, for a node j, its embedding vector θ_j can be looked up from Θ (where $\theta_j \in R^d$ and d is the predefined embedding size). This embedding matrix is randomly initialized and the objective of our framework is to learn from the observed data and optimize $\mathbf{\Theta}$, such that the embeddings are representative of the graph structure and capture complex user-item relations.

It is also worth noting that to keep the model light, we refrain from using any additional parameters. The single embedding matrix $\mathbf{\Theta}$ contains all learnable parameters that are necessary for the proposed HNB model.

B. Feature-Level Attention for Conditional Proximity

Next, we describe the equations of computing various proximity. Especially, for high-order proximity, the key concept of "conditional proximity" will be conditional on intermediate nodes and be measured by feature-level attention.

For 1-order proximity. Following prior works [5], [7], the proximity score between u and its direct neighbor i is measured by:

$$\mathbf{Prox}(u, i) = \theta_u^\top \theta_i.$$

Since direct user-item interaction is the strongest signal when measuring proximity, we use the inner product to ensure that the two embeddings should be close across all dimensions.

For 2-order proximity. Unlike direct neighbors, we model high-order proximity as conditional proximity, where the intermediate node is taken into account as condition. We denote the conditional proximity between a source node u and its 2-order neighbor a_{i3} as:

$$\mathbf{Prox}(u, a_{i3}|i),$$

where i is the intermediate node connecting u and a_{i3}. Illustration can be found in Figure 3.

To model $Prox(u, a_{i3}|i)$ and to emphasize the effect of intermediate node, we propose to employ feature-level attentive mechanism. Ideally, such mechanism should first aggregate features' information from both source node and intermediate node and then guide the information flows to focus on only important features.

$$\mathbf{Prox}(u, a_{i3}|i) = [\mathbf{Atten}(\theta_u, \theta_i) \odot \theta_u]^\top \theta_{a_{i3}}, \tag{2}$$

$$\text{where } \mathbf{Atten}(\theta_u, \theta_i) = softmax(\theta_u \oplus \theta_i). \tag{3}$$

The $\oplus$ and $\odot$ are element-wise addition and product operation, the $\mathbf{Atten}(\cdot)$ is the feature-level attention and the $softmax(\cdot)$ is:

$$softmax(\theta)_l = \frac{e^{\vartheta_l}}{\sum_{j=1}^{d} e^{\vartheta_j}},$$
$$\text{for } l = 1, \ldots, d \text{ and } \theta = (\vartheta_1, \cdots, \vartheta_d) \in R^d.$$

We utilize the $\oplus$ to model feature-level attention $\mathbf{Atten}(\cdot)$, because when considering further neighbors (e.g.

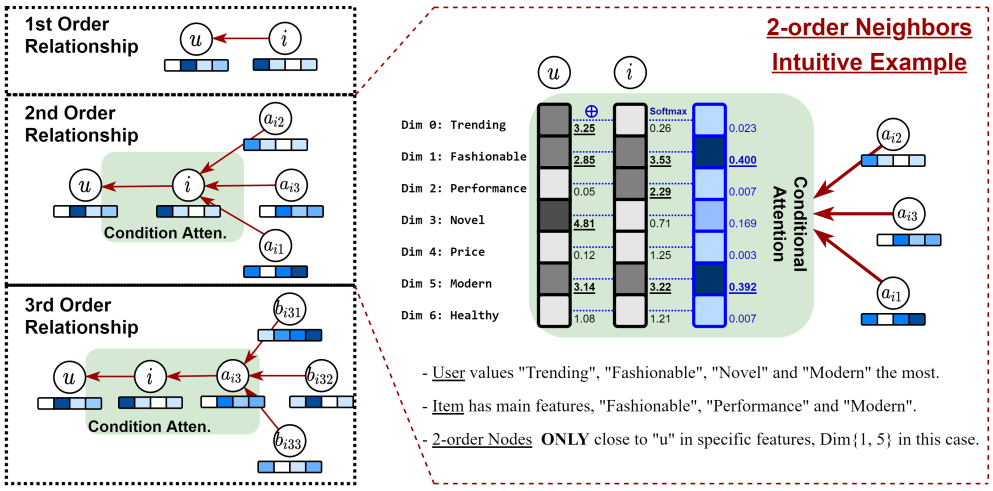


Fig. 3: Intuitive Example of how Feature-level Attention models Conditional Information Flowing. Best view in color.

$\mathbf{Atten}(\theta_0, \theta_1, \theta_2, \theta_3)$), the product $\odot$ will produce very small numbers, while the simpler adding is stabler.

The mechanism is intuitively explained in Figure 3. Supposed each dimension of embedding encodes an interpretable feature of preference, e.g. dimension 0 for the feature "Trending", then the source node u has higher preference for dimensions $\{0, 1, 3, 5\}$, while the intermediate node i is featured by dimensions $\{1, 2, 5\}$.

When modelling the 2-order proximity between $\{a_{i\cdot}\}$ and u, typical graph models [8], [9] would treat it like the 1-order proximity and ignore the intermediate node i, which leads to massive noises being brought in to u along the information flows. As discussed in section II-A, a suggestion of an item could be perceived differently depending on the intermediate node. A feature-level attention can effectively fix this. It introduces the intermediate node i and constrains the information flows to pass through only significant features of both u and i. Eventually, noisy features are filtered and key features are enhanced. In the example, u finally receives information mainly about "dim 1: Fashionable" and "dim 5: Modern" from the 2-order neighbors $a_{i\cdot}$ through i.

For even higher-order proximity. The same logic generalizes to higher order. Take a 3-order proximity example from Figure 3. Consider a path of information flow $[b_{i32} \rightarrow a_{i3} \rightarrow i \rightarrow u]$, the 3-order proximity is computed:

$$\mathbf{Prox}(u, b_{i32}|i, a_{i3}) = [\mathbf{Atten}(\theta_u, \theta_i, \theta_{a_{i3}}) \odot \theta_u]^\top \theta_{b_{i32}},$$
$$\text{where} \quad \mathbf{Atten}(\theta_u, \theta_i, \theta_{a_{i3}}) = softmax(\theta_u \oplus \theta_i \oplus \theta_{a_{i3}}).$$

General Proximity Notation. To simplify the notations, we use a general notation as follows for both conditional (high-order) and unconditional (1-order) proximity:

$$\mathbf{Prox}(s, e|\mathbf{m}) = \begin{cases} \mathbf{Prox}(s, e) & \text{if } k = 1, \\ \mathbf{Prox}(s, e|\text{ intermediate node(s)}) & \text{if } k > 1, \end{cases} \tag{4}$$

In Equation 4, $\mathbf{m}$ is a general symbol for intermediate node(s). In case of 1-order proximity ($k = 1$), $\mathbf{m}$ is $\emptyset$, indicating an unconditional proximity; for $k = 2$, $\mathbf{m}$ is the node between source s and end node e; for $k = 3$, $\mathbf{m}$ is the set of intermediate nodes; and so on.

C. Reinforce Conditional Proximity

We propose three components to emphasize the role of intermediate nodes in high-order propagation and to reinforce the conditional proximity.

Length-Aware Decay Factor. As studied in [10], [8], [9], the strength of proximity decays as the order gets higher. We take into account such effect by integrating a length-aware decay factor:

$$w_k = (\frac{1}{k})^\tau, \tag{5}$$

where τ is a hyper-parameter and can be chosen in range $(1, 8)$. The purpose of w_k is to emphasize close relations and down-weight faraway pairs. w_k is introduced in the loss function in section III-D.

Outdegree-Aware Factor. Moreover, node's outdegree also plays an important role in measuring user-item relations [10], [18]. Given a node j, its outdegree-aware factor w_{od_j} is obtained by:

$$w_{od_j} = \frac{1}{\log(|N_j| + 2)}, \tag{6}$$

where $|N_j|$ is the number of direct neighbors of j. The logarithm serves as a smoothing function and the $|N_j| + 2$ is to avoid situation where user or item has only 0 or 1 historical interaction.

We design the w_{od_j} to be inversely proportional to outdegree, because information coming from popular neighbor nodes (those having high outdegree) are less important than information from unpopular nodes (those having low outdegree) [10], [18].

In RSs, this is particularly important. For instance, when a user interacts with a popular item, this item actually does not provide much useful knowledge about this user, since almost all users are interested in it. Conversely, unpopular items (with low outdegree) carry much precious knowledge about users,

because users must have specific and strong preferences to buy something others would ignore or dislike. Moreover, two users are likely to be especially similar (in some aspects), if they both have interaction with a very unpopular item.

Extended Random Walks. There are 2 sampling methods often used to augment training data in recent graph-based CF models. One is random walk, which starts from a source node and collects one sample per step until reaching walk length k [4], [8], [9]. The other one [10], [11], gathers all possible neighbors on each order, similar to breadth-first search (BFS), generating exponentially great number of training samples. The simple random walk sampling is fast but is likely to miss meaningful neighbors, while the second sampling method maintains perfect neighborhood information but is very expensive to optimize in training.

To balance the two methods and to reinforce the conditional proximity, we use a straightforward "extended random walk". The random walk is extended by collecting not only 1, but $\eta > 1$ "indirect neighbors" in each step. Note that, since we aim at collecting training samples to reinforce the conditional proximity, the random walk is only extended when sampling high-order neighbors. For example, in Figure 3, given an edge $[i \rightarrow u]$ in dataset, the source node u will eventually have one direct neighbor i, three 2-order neighbors and nine 3-order neighbors when length $k = 3$ and $\eta = 3$. This simple sampling can collect enough information without forcing too heavy optimization in training. In experiments, we sample training data for HNB this way in preprocessing.

D. Optimization

Our model is trained by optimizing the Bayesian Pairwise Ranking loss function (BPR) [19], which has also been widely used in RSs [9], [8], [10], [11], [20]. BPR takes into account both positive and negative examples during training, where the positive and negative examples are observed and unobserved user-item pairs, respectively. The loss function of the proposed HNB framework is:

$$\begin{aligned}\textbf{Loss} = -\sum_{(s,e^+,e^-)} & \sqrt[3]{w_{od_s} w_{od_{e^+}} w_{od_{e^-}}} \\ & \cdot \log \sigma(w_k \cdot [\mathbf{Prox}(s, e^+|\mathbf{m}) - \mathbf{Prox}(s, e^-|\mathbf{m})]) \\ & + \lambda \|\Theta\|_2^2,\end{aligned}$$

where the set (s, e^+, e^-) is the collection of sampled source node s, positive end node e^+ and negative end node e^-; the first term is the geometric mean of the "w_{od}" of s, e^+ and e^-; the $\sigma(\cdot)$ is the *sigmoid function*; Θ is the embedding matrix; to prevent overfitting, we use L_2 regularization $\|\Theta\|_2^2$ with strength λ; *Adam Optimizer* is used to update model parameters Θ until convergence [21]. Through minimizing the BPR loss, HNB gathers conditional information filtered or enhanced by intermediate nodes and learns to distinguish positive items from negative items.

TABLE II: Statistics of Datasets.

Dataset	Sparsity	Users	Items	Interactions
Amazon-Book	99.94%	52,643	91,599	2,984,108
Gowalla	99.92%	29,858	40,981	1,027,370
ML-1M	95.53%	6,040	3,706	1,000,209
Citeulike-a	99.78%	5,551	16,980	204,987

E. Inference for Top-N Recommendations

The inference phase of HNB is as streamlined as the training phase. To generate a top-N recommendation list for a particular user u, HNB only needs to compute the inner products of embeddings and keep the top-N items as the suggestion list:

$$\theta_u^T \theta_i, \quad \text{for } i \notin N_u.$$

F. Complexity Analysis

Depending on the implementation, the complexity may vary with $\{d, |E|\}$, where E are the edges of a graph. During training, HNB performs vector operations $O(|E|dk)$, softmax $O(d)$ and SGD optimization $O(d)$. As for the space of model storage, since the embedding matrix contains all parameters of HNB, the space is $O((|U| + |I|) \cdot d)$.

The empirical computation time on Gowalla dataset are summerized as well. In training phase, BPR, CSE, NGCF and HNB take about $\{21s, 50s, 75s, 53s\}$ per epoch. In inference, BPR, CSE, NGCF and HNB take about $\{17s, 17s, 86s, 17s\}$. HNB has the same inference time as CSE and the simplest BPR, since they have the same size of embedding matrix for inference. Detailed parameter settings are in Section IV-B.

IV. Experiments

Comprehensive experiments are described in this section.

A. Datasets

Each dataset, summarized in Table II, is representative of a particular scenario in RSs (varies in sparsity, size). For all datasets, 72% of a user's historical items are randomly sampled as training set, 8% as validation set for hyper-parameters tuning and 20% as test set.

Amazon-Book (large; sparse). Amazon-Book is the largest and most sparse dataset in our experiments [22], reflecting real situations in e-commence and music store. As in [10], [20], [11], we keep nodes with at least ten data and binarize the ratings to {0, 1}.

Gowalla (mid; sparse). Gowalla is collected by Gowalla [23], where users interact with locations by checking-in. As in [10], [20], [11], we only keep users with at least ten locations.

ML-1M (mid; dense). ML-1M are constructed from users' ratings toward movies from MovieLen [24]. We binarize the original rating scores by setting observed entries as 1 and unobserved as 0.

Citeulike-a (small; sparse). Citeulike-a consists of implicit feedback data from CiteULike [25]. When a user shares a paper, the interaction is established with value 1; otherwise, 0.

B. Baseline Models and Settings

The proposed approach is compared with 7 baseline models including state-of-the-art models.

Model-base models:

- **BPR** [19] is a MF model specialized for implicit feedback data, which replaces the MSE with the BPR loss function.
- **NMF** [26] is a deep-learning CF model incorporating the features of MF and multilayer perceptron (MLP) with nonlinear activation. We use the default settings from the authors [2].
- **CMN** [27] is a state-of-the-art model that simulates the memory network [28] by integrating 2 "memories components" which enables reading and writing latent features flexibly.

Graph-base models:

- **RANK-CSE** [9] is a state-of-the-art graph-based model. Inspired by LINE [5], CSE employs context embedding to improve training quality. We implement the "ranking" variant of CSE in our experiments, since it uses BPR loss too.
- **PinSage** [29] PinSage adopts the convolutional neural network (CNN) in GraphSage [30] for RSs. The CNN is notably used for discerning key neighbors for a source node. As other works [10], [11], we also apply 2 convolutional layers.
- **HOP-Rec** [8] is a graph-based CF model which successfully merges the graph information with collaborative relations. The high-order proximity is obtained via data sampling.
- **NGCF** [10] is a state-of-the-art graph-based model. NGCF explicitly models the information propagation by stacking "propagation layer" recursively. The implementation is publicly available [3].

Parameter Settings.

All models are trained by optimizing the BPR loss function. For fair comparison, the embedding size d is set to 100 for all baselines, except for NGCF, which requires $64 \times 3(194)$ [10]. Other hyper-parameters of baselines are set as the same as their original works.

As for HNB, HNB_2 and HNB_3 are HNB with length $k = \{2, 3\}$ respectively. Note that the length in CSE and HOP-Rec are $k = 2$, as their default [9], [8]. The number of neighbors $\eta = 3$; embedding size $d = \{100, 150\}$; $\tau = 3$; the negative ratio is set as 5 for all models using BPR loss function.

C. Evaluation Metrics

Models are evaluated by comparing the predicted items R_u and the ground-true items $Test_u$ for each user u. All the results are evaluated based on the top-20 list. Metrics we used are:

- **Recall** is the ratio of the hold-out item being successfully included in the recommended top-N list.
  $\text{Recall}\ @N = \frac{1}{|U|} \sum_{u \in U} \frac{\sum_{i \in R_u} \mathbb{1}(i \in Test_u)}{|Test_u|}$.
- **Precision** is the fraction of the top-N suggested items that are actually hold-out items.
  $\text{Precision}\ @N = \frac{1}{|U|} \sum_{u \in U} \frac{\sum_{i \in R_u} \mathbb{1}(i \in Test_u)}{N}$.
- **Normalized Discounted Cumulative Gain (NDCG)** is order-sensitive measure, which computes accuracy like recall and penalizes greater when the positive item is ranked lower in the list [31].

D. Comparative Experiments

Comparative experimental results are summerized in Table III. Given that each dataset is representative of a particular scenario in RSs (vary in sparsity, size), we observe and discuss the results by dataset.

- **Amazon-Book (large; sparse).** A large and sparse dataset is representative of most e-commerce and music RS. Among baseline models, RANK-CSE outperforms all others, attributed to its ability in modelling user-user, item-item and user-item relations simultaneously. Compared with HOP-Rec and PinSage, RANK-CSE takes into account the effect of outdegree and also utilizes high-order neighbors to augment training set. RANK-CSE also has an extra "context embedding" when computing high-order proximity. The purposes of the context embedding are to avoid directly optimizing the vertex embedding because of the noisy signals from high-order neighbors, and to help put attentions on key neighbors, resembling a node-level attention. Based on the experimental results, we argue that these purposes can be better realized by the feature-level attention in HNB.
  HNB makes greatest improvements over baselines on Amazon-Book dataset than on other datasets. Specifically, HNB_2 (d=150) improves 12.5%, 12.4% and 17.9% on $Recall$, $Precision$ and $NDCG$. Compared to other models, HNB considers the influence of intermediate node in the process of information propagation and models it through feature-level attention. These turn out to help capture more meaningful user-item interactions and boost performance.
- **Gowalla (mid; sparse).** The experimental results are generally as expected. NGCF is the best baseline model showing the benefits of adding high-order neighbors and using training techniques like "message and node dropout". HNB outperforms baselines thanks to the feature-level attention, which allows the information to propagate conditionally only through important feature channels and filters noisy signals flowing from high-order neighbors to source nodes.
- **ML-1M (mid; dense).** Dense dataset is sometimes challenging to models specialized for extreme sparsity. In our experiment, NMF of simpler architecture, provides the best results in 4 model-based baselines. Graph-based models outperform model-based models again in ML-1M. Especially NGCF, thanks to its delicate propagation

[2] https://github.com/hexiangnan/neural_collaborative_filtering
[3] https://github.com/xiangwang1223/neural_graph_collaborative_filtering

layer, yields the best result on $Precison$ among most models. However, NGCF is relatively too heavy. With all neighbors included in every order and $d = 194$, NGCF requires considerable computation time on such dense dataset. HNB maintains its high performance and, by increasing the d to 150, it outperforms all baselines. We believe the reason is that dense dataset provides more information, which requires larger dimension size for sufficient information encoding.

- **Citeulike-a (small; sparse).** CMN yields the best results among model-based models thanks to its "neighbor-wise attention", which assigns various weights to different neighbors based on similarities. The superiority of HNB over CMN derives from the extra high-order information and properly filtering these information with feature-level attention. NGCF, however, underperforms a simpler BPR. The reasons might be complicated, but we believe NGCF trying to make use of all possible neighbors (not sub-sampling them) in such small dataset may cause popular nodes appearing too often in optimization and jeopardise performance. While in larger dataset, node's importance is diluted and this problem is alleviated. Compared to NGCF, HNB sub-samples high-order neighbors and is able to maintain its ability in small dataset.

E. Analysis of HNB

In this part, we further analyze HNB from various aspects.

*1) **Embedding Size** d:* In Section IV-D, we find that HNB is able to learn more meaningful embedding when size d increases. We here further examine HNB with a larger range of embedding size, $d = \{50, 75, 100, 125, 150, 175, 200\}$. As shown in Figure 4 (Amazon-Book; HNB ($k = 2, \eta = 3$)), HNB shows better performances with greater embedding size. And with the least size setting, $d = 50$, HNB already outperforms all model-based baseline models and HOP-Rec. In principle, HNB consistently yields better results with greater d, but the increment drops. Due to limited resources, HNB of $d > 200$ hasn't been examined in this work, but the results show that if more memory is allowed, HNB is capable of learning more meaningful and representative embedding.

*2) **Number of Neighbors** η:* Since η directly affects the sampling size and the optimization quality, we study the effects of η on HNB's performance in this part. Results are plotted in Figure 5 (Citeulike-a; HNB ($d = 100$)). Generally HNB's performances peak when η is around 3. This observation is a great support to our assumption stated in Section III-C that reinforcing the conditional flows with only few random neighbors, other than with all neighbors, contributes to a more robust and stable embedding. Moreover, though $\eta > 5$ might result in inferior performance, the number of epochs needed till convergence declines dramatically, implying a trade-off between performance and training time.

*3) **Embedding Visualization**:* To further investigate how the conditional proximity helps learn user-item relations, we visualize the t-SNE transformed embeddings of the top 2 models on Amazon-Book dataset, CSE and HNB_2 (d=100). Figure 6(a) and 6 (b) plot embeddings of 8 randomly sampled users and their related test items. Two interesting observations can be noticed:

- Tighter clusters. Generally, nodes of same colors tend to be much closer together (noticeably tighter clusters) in Figure 6 (b) than in Figure 6(a), indicating the embedding of HNB encodes more meaningful user-item relations and achieves better clustering effect among user and related items.
- Effectiveness of the feature-level attention. We also notice a more subtle inference. Note that one key difference between HNB and CSE is that CSE utilizes an extra "context embedding matrix" during training [9], which helps the model focus on important neighbors, resembling a "node-level" attention mechanism, while the HNB organizes information flows by "feature-level" attention. As a result, we argue that the better clustering effect is achieved thanks to the feature-level attention which helps filter noises and enhance key signals on latent features for high-order proximity.

*4) **Ablation Study**:* To evaluate the effectiveness of each component, an ablation study is conducted. As shown in Table IV, experiments are on Amazon-Book and the base model is BPR ($d = 100$).

First, it's interesting to notice that, without considering the effects of path length and outdegree, the k-order proximity alone actually jeopardizes the performance. Second, the designed feature-level attention significantly boosts the performance, indicating the effectiveness of the conditional flow mechanism and the intermediate nodes are indeed critical when optimizing with indirect neighbors. Finally, the full HNB model is completed with training examples collected by "extended random walks" in preprossing. The further improvement shows that the conditional proximity is reinforced by adding more indirect samples, which helps encode meaningful features in embedding.

V. Related Work

Model-based CF Approach. Model-based CF approach makes assumption of collaborative filtering that similar users tend to behave in the same way, and improves the recommendations by modifying on a model level, for example adding various specialized layers. 3 baseline models we used in the experiments, BPR, NMF and CMN, are model-based. The structure details for these models are summarized in Section IV-B.

Similar to model-based methods, HNB also aims at improving the capability of capturing meaningful signals from neighbors. However, rather than node-level attention, HNB uses a more fine-grained feature-level attention. HNB also differs from these models in exploiting not only direct but indirect high-order neighbors.

Graph-based CF Approach. Graph-based CF approach learns user-item relations by exploring the graph structure. Typically, user-item interactions would be first constructed as a bipartite graph. Then, for a randomly chosen source node, its

TABLE III: Comparative experiments for HNB. Best results are in bold. The symbol ‡ denotes the best baseline results. The improvements (%) of HNB to the best baseline are summarized in the parentheses. HNB_2 and HNB_3 are HNB with length $k = \{2, 3\}$. Generally, HNB outperforms all baselines, especially on Amazon-Book dataset.

	@20	BPR	NMF	CMN	RANK-CSE	PinSage	HOP-Rec	NGCF	HNB_2 (d=100)	HNB_2 (d=150)	HNB_3 (d=150)
Amazon-Book (large; sparse)	**Rec.**	0.0256	0.0261	0.0284	‡0.0351	0.0288	0.0309	0.0344	**0.0372** (+6.0%)	**0.0395** (+12.5%)	**0.0397** (+12.7%)
	Prec.	0.0121	0.0122	0.0130	‡0.0145	0.0128	0.0124	0.0138	**0.0153** (+5.5%)	**0.0163** (+12.4%)	**0.0164** (+12.9%)
	NDCG	0.0520	0.0526	0.0597	‡0.0638	0.0558	0.0606	0.0630	**0.0700** (+9.7%)	**0.0746** (+17.9%)	**0.0760** (+19.0%)
Gowalla (mid; sparse)	**Rec.**	0.1359	0.1388	0.1404	0.1049	0.1382	0.1399	‡0.1547	**0.1600** (+3.4%)	**0.1646** (+6.4%)	**0.1654** (+6.9%)
	Prec.	0.0417	0.0430	0.0445	0.0324	0.0451	0.0456	‡0.0470	**0.0487** (+3.6%)	**0.0500** (+6.4%)	**0.0502** (+7.0%)
	NDCG	0.1890	0.2023	0.2129	0.1712	0.1944	0.2128	‡0.2237	**0.2315** (+3.5%)	**0.2382** (+6.6%)	**0.2391** (+6.9%)
ML-1M (mid; dense)	**Rec.**	0.1761	0.1834	0.1733	0.1917	0.1902	0.1900	‡0.2447	**0.2463** (+0.7%)	**0.2513** (+2.7%)	**0.2516** (+2.8%)
	Prec.	0.2147	0.2221	0.2132	0.2175	0.2164	0.2178	‡0.2810	0.2747 (-2.2%)	**0.2810** (0.0%)	**0.2818** (+0.3%)
	NDCG	0.5590	0.5607	0.5416	0.5780	0.5709	0.5629	‡0.6524	0.6354 (-2.6%)	**0.6877** (+5.4%)	**0.6909** (+5.5%)
Citeulike-a (small; sparse)	**Rec.**	0.2200	0.2224	0.2375	0.2503	0.2418	‡0.2549	0.2138	**0.2601** (+2.1%)	**0.2636** (+3.4%)	**0.2645** (+3.8%)
	Prec.	0.0742	0.0797	0.0827	0.0852	0.0845	‡0.0878	0.0802	**0.0886** (+0.9%)	**0.0911** (+3.8%)	**0.0913** (+4.0%)
	NDCG	0.3288	0.3200	0.3330	0.3486	0.3381	‡0.3489	0.3424	**0.3599** (+3.1%)	**0.3762** (+7.8%)	**0.3775** (+8.2%)

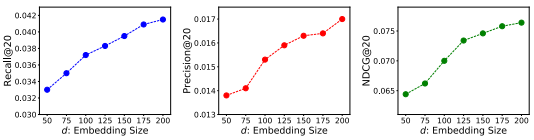


Fig. 4: Experiments on Embedding Size d.

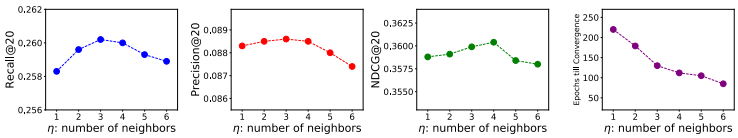
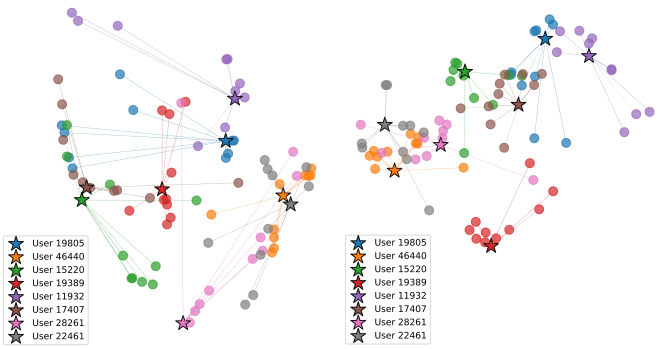


Fig. 5: Experiments on Number of High-order Neighbors.



(a) CSE (b) HNB

Fig. 6: Visualization of t-SNE transformed embeddings of the top 2 models on Amazon-Book dataset. The stars are 8 sampled users and the circles of same color are their test items. Generally, HNB can achieve much tighter clusters.

TABLE IV: Ablation Study.

Amazon-Book (@20)	Recall	Precision	NDCG
BPR ($d = 100$)	0.0256	0.0121	0.0520
+ k-order Proximity ($k = 3$)	0.0263	0.0119	0.0517
+ Length-Aware Decay Factor	0.0298	0.0124	0.0576
+ Outdegree-Aware Factor	0.0315	0.0126	0.0603
+ Feature-level Attention	0.0367	0.0151	0.0702
+ Full HNB ($\eta = 3$)	**0.0379**	**0.0155**	**0.0718**

graph structural properties, for example, neighbors, position and community, would be collected and used to optimize the embedding. Some graph-based CF models are described in Section IV-B. In general, graph-based CF models learn better embedding vectors than its model-based counterparties. We argue that the superiority should attribute to the fact that the lack of data due to the extreme sparsity of RSs, is appreciably compensated by the information signals from high-order neighbors.

HNB, as a graph-based approach, also enriches training data by making use of high-order information. Compared to other graph-based models, HNB explicitly model the effect of intermediate nodes to filter noises from indirect neighbors.

Graph Attention Model. Attention mechanism has also been widely applied in graph embedding for helping base models to focus on decisive parts. Attention Walk [17] integrates the attention mechanism in the sampling phase, where the random surfer is endowed with the ability of sampling close neighbors for a source node. Graph Attenion Network [14] improves the graph embedding model with a new convolution-style attention layer which assigns different importance weights to different nodes of a community. KGAT [16] successfully employs the attentive embedding propagation

in knowledge graph to adaptively and attentively propagate only important relation information of a knowledge graph. In social recommendation, DGRec [32] learns dynamically users' embeddings for online suggestion, where the attention is used to distinguish key direct neighbors.

Nevertheless, as most works focus on distinguishing important neighbors, few previous works actually utilize attention mechanism on a feature level. HNB differs from these models in using attention to capture pivotal feature channels within embeddings when learning from high-order information flows.

VI. Conclusion

In the work, we explore the importance of intermediate nodes and introduce the concept of "conditional proximity" for high-order information propagation. The process of conditional flowing is captured by a feature-level attention mechanism, which helps model to focus on decisive feature channels when measuring high-order proximity. We summarize the ideas and propose our HNB framework for RS. HNB is light in weight, with no extra parameters, and superior in performance, supported by the comprehensive experiments against state-of-the-art baselines.

As future work, we plan to extend HNB to a general graph embedding model not just for RSs, as "conditional proximity" is a general concept for graph analysis.

REFERENCES

[1] Z. Zolaktaf, R. Babanezhad, and R. Pottinger, "A generic top-n recommendation framework for trading-off accuracy, novelty, and coverage," in *2018 IEEE 34th International Conference on Data Engineering (ICDE)*. IEEE, 2018, pp. 149–160.

[2] ——, "A generic top-n recommendation framework for trading-off accuracy, novelty, and coverage," in *2018 IEEE 34th International Conference on Data Engineering (ICDE)*. IEEE, 2018, pp. 149–160.

[3] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[4] B. Perozzi, R. Al-Rfou, and S. Skiena, "Deepwalk: Online learning of social representations," in *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2014, pp. 701–710.

[5] J. Tang, M. Qu, M. Wang, M. Zhang, J. Yan, and Q. Mei, "Line: Large-scale information network embedding," in *Proceedings of the 24th international conference on world wide web*. International World Wide Web Conferences Steering Committee, 2015, pp. 1067–1077.

[6] A. Grover and J. Leskovec, "node2vec: Scalable feature learning for networks," in *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2016, pp. 855–864.

[7] M. Gao, L. Chen, X. He, and A. Zhou, "Bine: Bipartite network embedding," in *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*. ACM, 2018, pp. 715–724.

[8] J.-H. Yang, C.-M. Chen, C.-J. Wang, and M.-F. Tsai, "Hop-rec: high-order proximity for implicit recommendation," in *Proceedings of the 12th ACM Conference on Recommender Systems*. ACM, 2018, pp. 140–144.

[9] C.-M. Chen, C.-J. Wang, M.-F. Tsai, and Y.-H. Yang, "Collaborative similarity embedding for recommender systems," in *The World Wide Web Conference*. ACM, 2019, pp. 2637–2643.

[10] X. Wang, X. He, M. Wang, F. Feng, and T.-S. Chua, "Neural graph collaborative filtering," *arXiv preprint arXiv:1905.08108*, 2019.

[11] J. Sun, Y. Zhang, C. Ma, M. Coates, H. Guo, R. Tang, and X. He, "Multi-graph convolution collaborative filtering," in *ICDM 2019, The 19th IEEE International Conference on Data Mining, Beijing, China, 8-11 November 2019*. ACM, 2019.

[12] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *The journal of machine learning research*, 2014.

[13] W. Zaremba, I. Sutskever, and O. Vinyals, "Recurrent neural network regularization," *arXiv preprint arXiv:1409.2329*, 2014.

[14] P. Velickovic, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, "Graph attention networks," in *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*, 2018.

[15] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in neural information processing systems*, 2017, pp. 5998–6008.

[16] X. Wang, X. He, Y. Cao, M. Liu, and T.-S. Chua, "Kgat: Knowledge graph attention network for recommendation," *arXiv preprint arXiv:1905.07854*, 2019.

[17] S. Abu-El-Haija, B. Perozzi, R. Al-Rfou, and A. A. Alemi, "Watch your step: Learning node embeddings via graph attention," in *Advances in Neural Information Processing Systems*, 2018, pp. 9180–9190.

[18] K. Lo and T. Ishigaki, "Matching novelty while training: Novel recommendation based on personalized pairwise loss weighting," in *ICDM 2019, The 19th IEEE International Conference on Data Mining, Beijing, China, 8-11 November 2019*, 2019.

[19] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme, "Bpr: Bayesian personalized ranking from implicit feedback," in *Proceedings of the twenty-fifth conference on uncertainty in artificial intelligence*. AUAI Press, 2009, pp. 452–461.

[20] R. He and J. McAuley, "Vbpr: visual bayesian personalized ranking from implicit feedback," in *Thirtieth AAAI Conference on Artificial Intelligence*, 2016.

[21] S. Abu-El-Haija, B. Perozzi, and R. Al-Rfou, "Learning edge representations via low-rank asymmetric projections," in *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*. ACM, 2017, pp. 1787–1796.

[22] R. He and J. McAuley, "Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering," in *proceedings of the 25th international conference on world wide web*, 2016.

[23] D. Liang, L. Charlin, J. McInerney, and D. M. Blei, "Modeling user exposure in recommendation," in *Proceedings of the 25th International Conference on World Wide Web*, 2016.

[24] F. M. Harper and J. A. Konstan, "The movielens datasets: History and context," *TiiS*, vol. 5, no. 4, pp. 19:1–19:19, 2016. [Online]. Available: https://doi.org/10.1145/2827872

[25] C. Wang and D. M. Blei, "Collaborative topic modeling for recommending scientific articles," in *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Diego, CA, USA, August 21-24, 2011*, 2011, pp. 448–456.

[26] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, and T.-S. Chua, "Neural collaborative filtering," in *Proceedings of the 26th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 2017, pp. 173–182.

[27] T. Ebesu, B. Shen, and Y. Fang, "Collaborative memory network for recommendation systems," in *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*. ACM, 2018, pp. 515–524.

[28] S. Sukhbaatar, A. Szlam, J. Weston, and R. Fergus, "End-to-end memory networks," in *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, 2015, pp. 2440–2448.

[29] R. Ying, R. He, K. Chen, P. Eksombatchai, W. L. Hamilton, and J. Leskovec, "Graph convolutional neural networks for web-scale recommender systems," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2018.

[30] W. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in *Advances in neural information processing systems*, 2017, pp. 1024–1034.

[31] F. Ricci, L. Rokach, and B. Shapira, "Introduction to recommender systems handbook," in *Recommender Systems Handbook*, 2011, pp. 1–35.

[32] W. Song, Z. Xiao, Y. Wang, L. Charlin, M. Zhang, and J. Tang, "Session-based social recommendation via dynamic graph attention networks," in *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*. ACM, 2019, pp. 555–563.