Generalized Personalized PageRank with Graph Convolutional Networks in Recommender Systems

Hiroshi Wayama
hiroshi.wayama@gmail.com
Kyoto University
Kyoto, Japan

Kazunari Sugiyama
sugiyama-k@g.osaka-seikei.ac.jp
Osaka Seikei University
Osaka, Japan

Abstract

Applying graph convolutional networks (GCNs) to recommender systems can provide more relevant items to each user. However, it is important to provide not only items that are relevant to each user's preferences but also items that are diverse and novel on e-commerce platforms. Users often prefer a variety of recommendations as well as ordinary recommendations obtained from their own search behaviors. This is achieved by LightGCN, a GCN-based recommender system that utilizes a multi-layer aggregation function and adjacency matrix to learn the latent vectors of users and items. However, LightGCN often provides recommendations without diversity when the number of layers is insufficient. On the other hand, when it is excessive, the accuracy is not acceptable, which is often referred to as the over-smoothing problem. To address this problem, we propose Generalized Personalized PageRank (GPPR) and Adjacency Personalized PageRank (APPR). GPPR is a generalized model of Personalized PageRank that can be applied to any graphs. In addition, APPR enhances Personalized PageRank, which is a specialized version of GPPR. We incorporate our approach into LightGCN and demonstrate that it provides more relevant and diverse recommendations, outperforming LightGCN. We have released our code[1] for this work.

CCS Concepts

• **Information systems → Recommender systems**.

Keywords

Recommender Systems, Graph Convolutional Network, Personalized PageRank

ACM Reference Format:
Hiroshi Wayama and Kazunari Sugiyama. 2025. Generalized Personalized PageRank with Graph Convolutional Networks in Recommender Systems. In *Proceedings of the 2025 International ACM SIGIR Conference on Innovative Concepts and Theories in Information Retrieval (ICTIR) (ICTIR '25), July 18, 2025, Padua, Italy.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3731120.3744607

[1]https://github.com/hiroshi0530/ICTIR2025_GPPR

1 Introduction

From Neural Graph Collaborative Filtering (NGCF) [34] to LightGCN [13], Graph Convolutional Networks (GCNs) have attracted much attention as they can enhance collaborative filtering (CF) by exploiting proximity information among users and items.

GCNs first set the embedding vector and then propagate it to the next layer based on the adjacency matrix. LightGCN removes feature transformation and nonlinear activation, and linearly aggregates normalized neighbor embeddings, thereby reducing computational complexity relative to NGCF and improving relevance. Subsequent to LightGCN, a lot of recommender systems have been proposed so far, such as UltraGCN [24], IMP-GCN [22], SGL [38], RecDCL [41], and LightGCL [4]. However, graph-based recommender systems have some critical limitations. For example, the difficulty in capturing distant relationships between users and items is one of them. While GCNs can capture distant relationships by increasing layers, they only average the embedding vectors and cannot sufficiently learn the graph structures. This is referred to as the over-smoothing problem. To address this, Klicpera et al. [19] proposed Personalized Propagation of Neural Predictions (PPNP) and Approximate Personalized Propagation of Neural Predictions (APPNP), which integrate GCNs with Personalized PageRank [28]. Furthermore, in our previous work, we proposed Quantum-GCN, which is a recommender system that integrates the theory of quantum random walks with GCNs [36].

Previous works on recommender systems mostly predicted user ratings for unknown items. However, recently, researchers' attention has shifted to item ranking, diversity, and novelty [15, 35]. Diversity is important to satisfy each user's various preferences. Generally, most users prefer recommender systems that provide a wide variety of items, not those that just match their preferences.

We aim to develop a recommender system that can provide better accuracy and diversity while addressing the over-smoothing problem. In addition, we theoretically analyze the mathematical structures of our proposed models. Our main contributions can be summarized as follows:

(1) We propose Generalized Personalized PageRank (GPPR) and Adjacency Personalized PageRank (APPR), which utilize the teleportation probability from Personalized PageRank.
(2) We incorporate our approach into LightGCN and show that it can achieve better relevance and diversity.
(3) We mathematically analyze the characteristics of the adjacency matrices defined in GPPR and APPR and show that they work as low-pass filters.
(4) We develop an efficient recommender system that addresses the over-smoothing problem.

2 Related Work

2.1 Graph Convolutional Networks

Collaborative Filtering (CF) is a canonical approach in recommender systems that models the user-item rating matrix built from explicit feedback (*e.g.*, ratings) and implicit feedback (*e.g.*, purchases or clicks). Matrix Factorization (MF) [20, 32] is also widely used to transform high-dimensional data into a low-dimensional representation to recommend more relevant items for each user. Users and items are first represented as trainable embedding vectors, and then the predicted score is computed as the inner product.

A number of recommender systems using graph structure and deep learning have been proposed and have achieved great success [2, 11, 30]. These methods are also used for graph classification [27] and semi-supervised node classification [17]. Some nice surveys on recommender systems using graph neural networks have been published [6, 9, 39]. Hereafter, we outline the neural network and graph-based models closely related to our work.

Wang et al. [34] proposed Neural Graph Collaborative Filtering (NGCF), which uses a graph convolution layer to aggregate information from neighboring nodes. This approach can capture high-order interactions between users and items. He et al. [13] developed LightGCN, which removes the nonlinear activation function and feature transformation of NGCF and employs a simple convolutional structure. Mao et al. [24] developed UltraGCN, which directly approximates the constraints of infinite-layer graph convolution and reduces computational costs. He et al. [12] proposed initialization with network embedding on compact graphs to enhance GCN-based CF models, eliminate nonlinearities, improve recommendation accuracy, and reduce smoothing problems.

2.2 Personalized PageRank and PPNP

Brin and Page [3] introduced artificial irreducibility and acyclicity in the adjacency matrix to ensure ergodicity, showing the existence of an eigenvector with eigenvalue 1, interpreted as a random walker's stationary probability at each node. Page et al. [28] then added a teleportation vector to return to the starting node, yielding Personalized PageRank which effectively incorporates information on distant and local nodes. Paudel et al. [29] proposed a graph vertex ranking method based on a 3-step random walk (RP^3_β). However, it does not address teleportation probability.

Klicpera et al. [19] proposed Personalized Propagation of Neural Predictions (PPNP) and Approximate Personalized Propagation of Neural Predictions (APPNP). PPNP combines GCNs with PageRank, whereas APPNP offers a faster approximation with comparable accuracy. Compared with conventional models, PPNP achieves efficient training with the same or fewer parameters.

2.3 Evaluation of Diversity

Previous works on recommender systems focused on improving accuracy. However, many researchers have aimed at "beyond accuracy" such as diversity, surprise discovery, novelty, and comprehensiveness [15]. Kaminskas et al. [15] surveyed these goals, reviewed relevant metrics and optimization strategies, and showed that evaluation-based diversity and novelty are positively correlated, with novelty broadening the scope of recommendations. Ziegler et al. [43] introduced an intra-list similarity (ILS) metric to diversify recommended items using a probabilistic topic model. ILS is also used in [40, 42].

3 PRELIMINARIES

3.1 Overview of LightGCN

We first overview LightGCN [13], a state-of-the-art GCN-based recommender system.

GCNs learn embedding vectors by convolving and smoothing node features [18, 37]. NGCF [34] aggregates neighbor features and uses non-linear activation, whereas LightGCN [13] skips these steps and simply sums neighbor features. Figure 1 shows the aggregation function in LightGCN. Despite this simplification, LightGCN outperforms previous GCN-based recommender systems.

Collaborative filtering uses a preference matrix (user-item interaction matrix) with 1 if a user gives implicit feedback (*e.g.*, clicks, purchases, and so on) and 0 otherwise. The preference matrix is defined as follows:

$$\mathbf{R_{u,i}} = \begin{cases} 1 & \text{if } (u,i) \text{ interaction is observed} \\ 0 & \text{otherwise.} \end{cases}$$

The adjacency matrix is defined using the preference matrix as follows:

$$\mathbf{A} = \begin{pmatrix} \mathbf{0} & \mathbf{R} \\ \mathbf{R}^T & \mathbf{0} \end{pmatrix}.$$

The dimensions of the matrix are given by $(m+n) \times (m+n)$, where m and n denote the number of users and items, respectively. The diagonal elements of the square matrix $\mathbf{D_U} = \text{Diag}(\mathbf{R} \cdot \mathbf{1})$ constitute the aggregate number of interactions for each user. Diag is a function that converts a vector into a matrix with diagonal elements. $\mathbf{1}$ denotes a vector where all elements are 1. By using the square matrix $\mathbf{D_I} = \text{Diag}\left(\mathbf{1}^T \cdot \mathbf{R}\right)$ and $\mathbf{D_U}$, the normalized preference matrix $\hat{\mathbf{R}} = \mathbf{D_U}^{-\frac{1}{2}}\mathbf{R}\mathbf{D_I}^{-\frac{1}{2}}$ is established. The normalized adjacency matrix is defined by utilizing the normalized preference matrix as follows:

$$\hat{\mathbf{A}} = \begin{pmatrix} \mathbf{0} & \hat{\mathbf{R}} \\ \hat{\mathbf{R}}^\mathbf{T} & \mathbf{0} \end{pmatrix}. \tag{1}$$

In LightGCN [13], information at the nodes of the graph is propagated to the next layer by the normalized adjacency matrix. In the first layer, user and item information is represented by an embedding vector. The adjacency matrix then aggregates the features and propagates the information to the next layer.

Let $\mathbf{e}_{u_s}$ and $\mathbf{e}_{i_t}$ be the latent vectors of user u_s and item i_t, respectively. The initial state is defined as $\mathbf{E}^{(0)} = \left(\mathbf{e}_{u_1}, \ldots, \mathbf{e}_{u_m}, \mathbf{e}_{i_1}, \ldots, \mathbf{e}_{i_n}\right)^T$. Then $\mathbf{E}^{(0)}$ is propagated to adjacent nodes in the graph through the normalized adjacency matrix $\hat{\mathbf{A}}$ as follows:

$$\mathbf{E}^{(1)} = \hat{\mathbf{A}}\mathbf{E}^{(0)}. \tag{2}$$

LightGCN [13] employs the normalized sum of neighborhood representations as an aggregation function, which allows the embedding vector to be propagated by multiplying the adjacency matrix from the left, as shown in Equation (2). The final user and item embedding vectors are obtained by averaging the representations

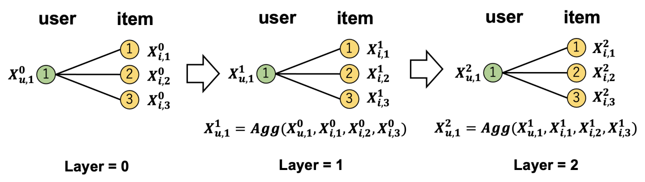

Figure 1: An example of smoothing the features of a node on a graph in GCN using an aggregation function. At layer 0, each user and item has an embedding vector as its initial state. At layer 1, each user and item vector is propagated to the next layer as this aggregate function. $\mathrm{X}_u^{(k+1)} = \mathrm{AGG}\left(\mathrm{X}_u^{(k)}, \left\{\mathrm{X}_i^{(k)} : i \in \mathcal{N}_u\right\}\right)$. In LightGCN [13], the aggregate function is defined as a function that simply takes the normalized sum.

from all $L+1$ layers as follows:

$$\mathbf{E} = \frac{1}{L+1}\left(\mathbf{E}^{(0)} + \hat{\mathbf{A}}\mathbf{E}^{(0)} + \cdots + \hat{\mathbf{A}}^L\mathbf{E}^{(0)}\right). \tag{3}$$

By computing the inner product of the final user vector $\mathbf{E}_u$ and the item vector $\mathbf{E}_i$, derived from Equation (3), our approach ranks the items for each user.

3.2 Personalized PageRank

Our method builds on Personalized PageRank. First, we define the column-normalized matrix $\tilde{\mathbf{A}}$ such that $\sum_j \tilde{\mathbf{A}}_{ij} = 1$, where $\tilde{\mathbf{A}}_{ij}$ is the transition probability from node i to j.

Page et al. [3] showed that making the adjacency matrix irreducible and acyclic yields an eigenvector (eigenvalue 1) representing the stationary probability of a random walker on each node. If $\boldsymbol{\pi}_{\mathrm{pr}}^{(t)}$ denotes the probability of a random walker being at each node at time t, then it evolves as $\boldsymbol{\pi}_{\mathrm{pr}}^{(t+1)} = \tilde{\mathbf{A}}^T\boldsymbol{\pi}_{\mathrm{pr}}^{(t)}$. If $t \to \infty$, $\boldsymbol{\pi}_{\mathrm{pr}}^{(t)}$ converges to a certain vector, which is the probability that the random walker exists at each node. This vector is referred to as PageRank, which is widely applied to a lot of works in information retrieval.

Page et al. [28] introduced the probability of returning to the first node (*i.e.*, teleportation vector) in the random walker and proposed Personalized PageRank. Let $\boldsymbol{\pi}_{\mathrm{ppr}}^{(t)}$ be the probability of a random walker staying at each node at time t in Personalized PageRank model and then $\boldsymbol{\pi}_{\mathrm{ppr}}^{(t)}$ evolves over time as follows:

$$\boldsymbol{\pi}_{\mathrm{ppr}}^{(t+1)} = (1-\alpha_0)\tilde{\mathbf{A}}^T\boldsymbol{\pi}_{\mathrm{ppr}}^{(t)} + \alpha_0\boldsymbol{\pi}_{\mathrm{ppr}}^{(0)}, \tag{4}$$

where α_0 is the probability of returning to the first node and satisfies $0 < \alpha_0 < 1$. This teleportation probability allows a random walker to stay close to the first node. Personalized PageRank effectively combines information about distant nodes with information about the neighborhood of the node. They define the adjacency matrix for Personalized PageRank as follows:

$$\hat{\mathbf{A}}_{\mathrm{PPR}} = \alpha_0\left(\mathbf{I} - (1-\alpha_0)\tilde{\mathbf{A}}\right)^{-1}. \tag{5}$$

The component $\left(\hat{\mathbf{A}}_{\mathrm{PPR}}\right)_{ij}$ of the adjacency matrix defined by Equation (5) is interpreted as the transition probability from node i to node j. This represents a latent connection between node i and node j. Furthermore, Equation (5) represents a dense matrix and contains a greater number of non-zero components than Equation (1), which can represent adjacencies between nodes that cannot be represented by Equation (1).

$\hat{\mathbf{A}}_{\mathrm{PPR}}$ has a larger value for the diagonal component as the probability of returning to itself increases for larger α_0. On the other hand, the smaller α_0, the higher the probability that a random walker is at a distant node and the less important the information at neighboring nodes. α_0 should not be too small or too large. It is a challenging task to set this value in Personalized PageRank.

3.3 PPNP

Klicpera et al. [19] proposed Personalized Propagation of Neural Predictions (PPNP) and Approximate Personalized Propagation of Neural Predictions (APPNP) by integrating GCNs with Personalized PageRank. They proposed the following simple model:

$$\mathbf{E}^{(k+1)} = \mathrm{softmax}\left[\alpha_0\left(\mathbf{I} - (1-\alpha_0)\tilde{\mathbf{A}}\right)^{-1} f_\theta(\mathbf{E}^{(k)})\right],$$

where $\mathbf{E}^{(k)}$ is the user–item matrix at layer k and f_θ is a neural network.

3.4 Definition of Diversity

Generally, recommender systems rank relevant items at higher rankings. On the other hand, it is also important to appropriately rank items with diversity and novelty. We explain how to quantitatively define diversity, which is commonly used in [1, 25, 33]. In this work, we define diversity using the average Euclidean distance between all pairs in a particular set, following Kaminskas and Bridge's work [15].

Let $\mathcal{U}$ and $\mathcal{I}$ be the sets of all users and items, respectively. Let $\mathcal{I}_u$ be the set of items recommended to user u. For items $i, j \in \mathcal{I}_u$ that belong to the set $\mathcal{I}$, we define a function $d(i,j) : \mathcal{I} \times \mathcal{I} \to \mathbb{R}$ which computes the Euclidean distance. Using the $d(i,j)$, we define Diversity as follows:

$$\mathrm{Diversity} = \frac{1}{|\mathcal{U}|}\sum_{u\in\mathcal{U}} \frac{\sum_{i\in\mathcal{I}_u}\sum_{j\neq i\in\mathcal{I}_u} d(i,j)}{|\mathcal{I}_u|(|\mathcal{I}_u|-1)}. \tag{6}$$

4 Proposed Method

We propose Generalized Personalized PageRank (GPPR), which generalizes and extends the framework of Personalized PageRank. We also propose Adjacency Personalized PageRank (APPR), which is a special case of GPPR. Personalized PageRank (PPR) considers teleportation probabilities for random walkers to a starting point of $t = 0$. GPPR, on the other hand, relies on teleportation probabilities to all nodes that random walkers can reach. APPR considers the probability of teleportation to nodes that random walkers can reach at $t = 0$ and $t = 1$. By using the teleportation probability to all nodes that random walkers can visit, we aim to more accurately address the potential relationships between users and items compared with Personalized PageRank. As a result, we can solve the over-smoothing problem and build an efficient recommender system with high accuracy and diversity without increasing the number of layers.

In this section, we first formulate APPR and derive the new adjacency matrix ($\hat{\mathbf{A}}_{\mathrm{APPR}}$) from APPR. Then we generalize the concepts of PPR and APPR, formulate GPPR, and derive the new adjacency matrix ($\hat{\mathbf{A}}_{\text{GPPR}}$) from GPPR. The properties of our newly defined adjacency matrices are discussed mainly in terms of eigenvalues. We show that the new adjacency matrices work better as a low-pass filter than the adjacency matrix used by LightGCN [13]. Finally, we explain that APPR is a special case of GPPR and uses its adjacency matrix to construct a graph neural network for a recommender system.

4.1 Adjacency Personalized PageRank (APPR)

4.1.1 New Adjacency Matrix as Stochastic Matrix. First, we define a new normalized adjacency matrix $\tilde{\mathbf{A}}$ as follows:

$$\tilde{\mathbf{A}} \overset{\text{def}}{=} \mathbf{D}^{-1}\mathbf{A}. \tag{7}$$

$\mathbf{D}$ is a degree matrix, defined as $\text{Diag}(\mathbf{A} \cdot \mathbf{1})$. See Section 3.1 for the definition of Diag and $\mathbf{1}$. $\tilde{\mathbf{A}}$ satisfies $\sum_j \tilde{\mathbf{A}}_{ij} = 1$ and can be defined as the probability matrix containing transition probabilities from node i to node j on the graph. $\tilde{\mathbf{A}}$ is the same as the matrix defined in Section 3.2.

As with the framework of PageRank, we consider random walkers starting from each node. Let $\boldsymbol{\pi}_s^{(t)}$ be the probability that a random walker starting from node s stays at each node at time t. $\boldsymbol{\pi}_s^{(0)}$ is a one-hot vector where s*-th* element is 1 and the others are 0. Let $\boldsymbol{\Pi}^{(t)}$ be probabilities that all random walkers stay at each node and defined as follows:

$$\boldsymbol{\Pi}^{(t)} = \left(\boldsymbol{\pi}_1^{(t)}, \boldsymbol{\pi}_2^{(t)}, \boldsymbol{\pi}_3^{(t)}, \ldots, \boldsymbol{\pi}_{m+n}^{(t)}\right). \tag{8}$$

4.1.2 Definition of $\hat{\mathbf{A}}_{APPR}$. In APPR, all random walkers transition to the next nodes according to the following formula:

$$\boldsymbol{\Pi}^{(t+1)^T} = \beta \boldsymbol{\Pi}^{(t)^T}\tilde{\mathbf{A}} + \alpha_0 \mathbf{I} + \alpha_1 \tilde{\mathbf{A}}, \tag{9}$$

where β denotes the probability of transitioning to the next node according to $\tilde{\mathbf{A}}$, α_0 and α_1 denote the probability of returning to its own first node and its own neighbor nodes, respectively. β, α_0, and α_1 satisfy $\beta + \alpha_0 + \alpha_1 = 1$.

In Equation (9), if $t \longrightarrow \infty$, $\boldsymbol{\Pi}^{(t)^T}$ converges to a stationary matrix, which is the probability matrix that all random walkers are present at each node. We define this matrix as $\hat{\mathbf{A}}_{\text{APPR}}$ as follows:

$$\boldsymbol{\Pi}^T = (\mathbf{I} - \beta\tilde{\mathbf{A}})^{-1}\left(\alpha_0 \mathbf{I} + \alpha_1 \tilde{\mathbf{A}}\right) \overset{\text{def}}{=} \hat{\mathbf{A}}_{\text{APPR}}. \tag{10}$$

4.1.3 Comparison between Personalized PageRank and Adjacency Personalized PageRank. Figure 2 shows the difference between PPR and APPR. PPR defines α_0 as the probability of teleporting to its own node. On the other hand, our proposed APPR defines α_1 as the probability of teleporting to a node adjacent to itself. By increasing α_1, the random walker can stay closer to its neighbors rather than its own node. In addition, the elements around the neighbor node also have non-zero values. This non-zero factor represents latent user-user, item-item, and user-item relationships that $\hat{\mathbf{A}}$ cannot represent. In addition, it allows the information of distant nodes to be aggregated in a very small number of layers, thus avoiding the over-smoothing problem.

The teleportation probability α_0 in Personalized PageRank [28] is defined as the probability of teleporting to its own node. However, if α_0 is large, the diagonal component of $\hat{\mathbf{A}}_{\text{PPR}}$ becomes larger, and

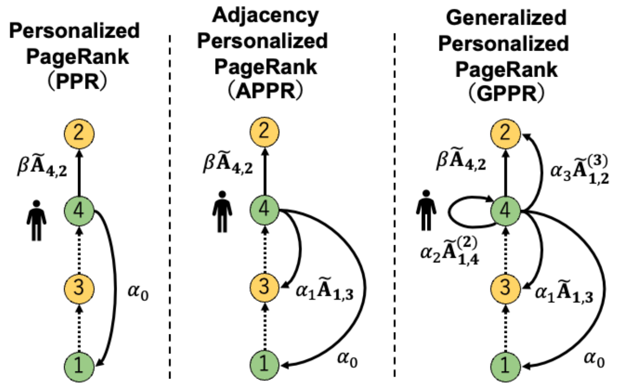
Personalized PageRank (PPR)
Adjacency Personalized PageRank (APPR)
Generalized Personalized PageRank (GPPR)
$\beta\tilde{\mathbf{A}}_{4,2}$
α_0
$\alpha_1\tilde{\mathbf{A}}_{1,3}$
$\alpha_2\tilde{\mathbf{A}}^{(2)}_{1,4}$
$\alpha_3\tilde{\mathbf{A}}^{(3)}_{1,2}$
1
2
3
4

Figure 2: Comparison between Personalized PageRank (PPR), Adjacency Personalized PageRank (APPR), and Generalized Personalized PageRank (GPPR). $\tilde{\mathbf{A}}_{i,j}$ denotes the transition probability from node i to node j. PPR defines α_0 as the probability of teleporting to the root node. On the other hand, our APPR defines α_0 and α_1 as the probabilities of teleporting to the root and adjacent nodes. Our GPPR defines $\alpha_0, \alpha_1, \cdots, \alpha_k$ as the probabilities of teleporting to the nodes that random walkers could reach. In all three models, $\beta + \sum_k \alpha_k = 1$.

the user-item component, which has an inherently large value, becomes relatively smaller. On the other hand, when α_0 is small, the probability of transition to a distant node increases, and the noise component of $\hat{\mathbf{A}}_{\text{PPR}}$ becomes larger.

To solve these problems, we introduce the teleportation probability α_1 as the probability of teleporting to a node adjacent to itself.

4.2 Generalized Personalized PageRank (GPPR)

4.2.1 Definition of $\hat{\mathbf{A}}_{GPPR}$. We generalize the concept of APPR and propose Generalized Personalized PageRank (GPPR). As shown on the right in Figure 2, in GPPR, all random walkers transition to the next node is defined as follows:

$$\boldsymbol{\Pi}^{(t+1)^T} = \beta \boldsymbol{\Pi}^{(t)^T}\tilde{\mathbf{A}} + \sum_{k=0}^{\infty} \alpha_k \tilde{\mathbf{A}}^k, \tag{11}$$

where β is the probability of transitioning to the next node based on the adjacency matrix $\tilde{\mathbf{A}}$ defined by Equation (7). α_k is the teleportation probability to reachable nodes at time $t = k$.

The probability that a random walker leaving node s at $t = 0$ can stay at time $t = k$ is indicated by the s*-th* row values of $\tilde{\mathbf{A}}^k$. β and α_k satisfy $\beta + \sum_{k=0}^{\infty} \alpha_k = 1$.

In Equation (11), if $t \longrightarrow \infty$, $\boldsymbol{\Pi}^{(t)^T}$ converges to a stationary matrix, which is the probability matrix that all random walkers present at each node in GPPR. Let $\hat{\mathbf{A}}_{\text{GPPR}}$ be the stationary matrix, then it is defined as follows:

$$\boldsymbol{\Pi}^T = (\mathbf{I} - \beta\tilde{\mathbf{A}})^{-1}\left(\sum_{k=0}^{\infty} \alpha_k \tilde{\mathbf{A}}^k\right) \overset{\text{def}}{=} \hat{\mathbf{A}}_{\text{GPPR}}. \tag{12}$$

The (i, j) components of $\hat{\mathbf{A}}_{\text{GPPR}}$ can represent the potential relationships between nodes i and j, which is impossible to use just matrix $\hat{\mathbf{A}}$. $\hat{\mathbf{A}}_{\text{GPPR}}$ is a dense matrix, and unlike that in LightGCN, where the zero components of $\hat{\mathbf{A}}$ contain no significant information, these components contain non-zero values in $\hat{\mathbf{A}}_{\text{GPPR}}$.

By using this matrix $\hat{\mathbf{A}}_{\text{GPPR}}$ in a graph convolutional network, we can minimize the number of layers to avoid the over-smoothing problem. We also believe that it can quantify the potential relationships between nodes, thereby avoiding the limitations of locality in convolution and enhancing diversity.

4.2.2 Formulation of three models. Table 1 shows $\mathbf{\Pi}^{(t+1)^T}$ and the newly defined adjacency matrices for the three models: PPR, APPR, and GPPR. APPR includes features of PPR, and GPPR includes features of both APPR and PPR. PPR and APPR are special cases of GPPR.

4.2.3 Theorems. $\hat{\mathbf{A}}_{\text{GPPR}}$ has the following mathematical properties. We prove the following equations in Section 7.

Theorem 4.2.1. *$\hat{\mathbf{A}}_{GPPR}$ can be represented using a Neumann series as follows:*

$$\hat{\mathbf{A}}_{GPPR} = \left(\sum_{k=0}^{\infty} \beta^k \tilde{\mathbf{A}}^k\right)\left(\sum_{k=0}^{\infty} \alpha_k \tilde{\mathbf{A}}^k\right) = \sum_{l=0}^{\infty}\sum_{k=0}^{l} \alpha_k \beta^{l-k} \tilde{\mathbf{A}}^l. \tag{13}$$

Theorem 4.2.2. *The sum of the coefficients of $\tilde{\mathbf{A}}^l$ in Equation (13) is 1. This indicates that $\hat{\mathbf{A}}_{GPPR}$ is a weighted average over the powers of $\tilde{\mathbf{A}}$.*

Theorem 4.2.3. *Let $\eta_1 \leqq \eta_2 \leqq \cdots \leqq \eta_{m+n}$ be the eigenvalues of $\hat{\mathbf{A}}_{GPPR}$, then these values are within the following bounds:*

$$-1 < \eta_1 \leqq \eta_2 \leqq \cdots \leqq \eta_{m+n} = 1. \tag{14}$$

4.2.4 $\hat{\mathbf{A}}_{APPR}$ and $\hat{\mathbf{A}}_{GPPR}$ as a lower-pass filter. In the signal processing domain, it is well-known that GCNs work as a frequency filter that cuts certain frequencies. GCNs are applicable when a given signal contains a lot of useful and useless information in the low- and high-frequency components, respectively.

As discussed in Simplified GCN [37], frequency components with eigenvalues close to 1 are removed. However, high-frequency components with eigenvalues close to 2 cannot be removed, and the filter does not satisfy the characteristics of a low-pass filter. Simplified GCN [37] solved this problem by integrating self-connection.

Table 2 shows the eigenvalues of the Laplacian matrices of $\hat{\mathbf{A}}$ and $\hat{\mathbf{A}}_{\text{GPPR}}$. As widely known, the eigenvalues of $\hat{\mathbf{A}}$ are between -1 and 1, inclusive. Thus, the eigenvalues of the Laplacian Matrix, which is $\mathbf{I} - \hat{\mathbf{A}}$, are greater than or equal to 0 and less than or equal to 2. On the other hand, in GPPR, we can define $\hat{\mathbf{L}}_{\text{GPPR}}$ as $\mathbf{I} - \hat{\mathbf{A}}_{\text{GPPR}}$. The maximum eigenvalue of $\hat{\mathbf{L}}_{\text{GPPR}}$ is less than 2 from Theorem 4.2.3. Similar to GPPR, $\hat{\mathbf{L}}_{\text{APPR}}$ can be defined as $\mathbf{I} - \hat{\mathbf{A}}_{\text{APPR}}$. The maximum eigenvalue of $\hat{\mathbf{L}}_{\text{APPR}}$ is less than 2.

For example, the maximum eigenvalue of $\hat{\mathbf{L}}_{\text{APPR}}$ with the hyperparameters in our experiments in Section 5 is 1.54. We set the hyperparameters of Equation (10) to $\beta = 0.3$, $\alpha_0 = 0$, and $\alpha_1 = 0.7$.

As shown in Simplified GCN [37], by keeping the maximum eigenvalue below 2, $\hat{\mathbf{A}}_{\text{APPR}}$ and $\hat{\mathbf{A}}_{\text{GPPR}}$ can work as a better low-pass filter than $\hat{\mathbf{A}}$.

4.2.5 Advantages of GPPR. The adjacency matrix $\hat{\mathbf{A}}_{\text{GPPR}}$, derived from Generalized Personalized PageRank, has a non-zero component that is absent in the adjacency matrix $\hat{\mathbf{A}}$ utilized in LightGCN.

Table 1: $\mathbf{\Pi}^{(t+1)^T}$ and newly defined adjacency matrices for the models: PPR, APPR, and GPPR.

Model	$\mathbf{\Pi}^{(t+1)^T}$	$\mathbf{A}_{\{\text{PPR, APPR, GPPR}\}}$
PPR	$\beta\mathbf{\Pi}^{(t)^T}\tilde{\mathbf{A}} + \alpha_0\mathbf{I}$	$\alpha_0(\mathbf{I} - \beta\tilde{\mathbf{A}})^{-1}$
APPR	$\beta\mathbf{\Pi}^{(t)^T}\tilde{\mathbf{A}} + \alpha_0\mathbf{I} + \alpha_1\tilde{\mathbf{A}}$	$(\mathbf{I} - \beta\tilde{\mathbf{A}})^{-1}\left(\alpha_0\mathbf{I} + \alpha_1\tilde{\mathbf{A}}\right)$
GPPR	$\beta\mathbf{\Pi}^{(t)^T}\tilde{\mathbf{A}} + \sum_{k=0}^{\infty}\alpha_k\tilde{\mathbf{A}}^k$	$(\mathbf{I} - \beta\tilde{\mathbf{A}})^{-1}\left(\sum_{k=0}^{\infty}\alpha_k\tilde{\mathbf{A}}^k\right)$

Table 2: The eigenvalues of the Laplacian matrices of $\hat{\mathbf{A}}$ and $\hat{\mathbf{A}}_{\text{GPPR}}$ are presented. The terms "Adj Mat" and "Lap Mat" in the table denote "Adjacency Matrix" and "Laplacian Matrix", respectively.

Adj Mat	Lap Mat	Eigenvalue of Lap Mat
$\hat{\mathbf{A}}$	$\hat{\mathbf{L}} = \mathbf{I} - \hat{\mathbf{A}}$	$0 = \lambda_1 \leqq \lambda_2 \leqq \cdots \leqq \lambda_n = 2$
$\hat{\mathbf{A}}_{\text{GPPR}}$	$\hat{\mathbf{L}}_{\text{GPPR}} \overset{\text{def}}{=} \mathbf{I} - \hat{\mathbf{A}}_{\text{GPPR}}$	$0 = \lambda_1 \leqq \lambda_2 \leqq \cdots \leqq \lambda_n < 2$

This feature allows $\hat{\mathbf{A}}_{\text{GPPR}}$ to aggregate information from distant nodes with only one graph convolution layer. Additionally, as outlined in Section 4.2.4, $\hat{\mathbf{A}}_{\text{GPPR}}$ also works as a low-pass filter. As a result, we can solve the over-smoothing problem and construct a recommender system with higher recommendation accuracy and diversity.

4.2.6 Relationship with LightGCN. β ranges from 0 to 1, and β^k rapidly decreases as k increases. For example, if k is greater than or equal to 2, we can assume that β^k is negligible. In this case, $\hat{\mathbf{A}}_{\text{PPR}}$ and $\hat{\mathbf{A}}_{\text{APPR}}$ can be approximated as follows:

$$\begin{cases} \hat{\mathbf{A}}_{\text{PPR}} = \displaystyle\sum_{l=0}^{\infty}\sum_{k=0}^{\min\{0,l\}} \alpha_k\beta^{l-k}\tilde{\mathbf{A}}^l \approx \alpha_0\mathbf{I} + \alpha_0\beta\tilde{\mathbf{A}}, \\ \hat{\mathbf{A}}_{\text{APPR}} = \displaystyle\sum_{l=0}^{\infty}\sum_{k=0}^{\min\{1,l\}} \alpha_k\beta^{l-k}\tilde{\mathbf{A}}^l \approx \alpha_0\mathbf{I} + (\alpha_1 + \alpha_0\beta)\tilde{\mathbf{A}} + \beta\alpha_1\tilde{\mathbf{A}}^2. \end{cases}$$

As indicated by Equation (3), in LightGCN, the embedding vector is a weighted average of the vectors for each layer specified by the hyperparameters. Compared with Equation (3), we see that they have the same formula except for coefficients for weights. $\hat{\mathbf{A}}_{\text{PPR}}$ is the same as LightGCN with one layer and $\hat{\mathbf{A}}_{\text{APPR}}$ is the same as LightGCN with two layers, respectively. We observe that taking the average of multiple layers in LightGCN is equivalent to considering the teleportation probability in GPPR. This indicates that GPPR is an extension of LightGCN.

4.3 APPR-GCN

We propose a new model integrating APPR and GCNs. GPPR has an infinite number of parameter combinations. In this paper, we conduct experiments on APPR and GCNs.

Figure 3 shows the overall architecture diagram of our proposed model. The initial matrix $\mathbf{E}^{(0)} = \left(\mathbf{e}_{u_1}, \ldots, \mathbf{e}_{u_m}, \mathbf{e}_{i_1}, \ldots, \mathbf{e}_{i_n}\right)^T$ is propagated to the next layer using the adjacency matrix $\hat{\mathbf{A}}_{\text{APPR}}$.

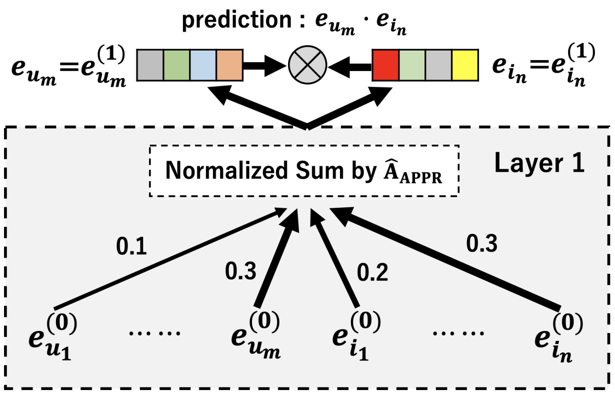

Figure 3: The overall architecture diagram of our proposed model. The user and item initial vectors are propagated to the next layer using the adjacency matrix $\hat{\mathbf{A}}_{\mathrm{APPR}}$. As in LightGCN [13], our model computes a simple summation using this adjacency matrix. The thickness of the arrows from the embedding vectors to the aggregation function is different, showing that $\hat{\mathbf{A}}_{\mathrm{APPR}}$ is a dense matrix and has various values from 0 to 1. Note that our model requires one layer only.

$\hat{\mathbf{A}}_{\mathrm{APPR}}$ defined by Equation (10) is a dense matrix, and non-zero values are stored in the elements for which $\hat{\mathbf{A}}$ is zero. These values can represent latent user-user, item-item, and user-item relationships that $\hat{\mathbf{A}}$ cannot represent.

As in LightGCN [13], it is composed of a simple summation by the adjacency matrix, eliminating the transformation layer of activation functions and features. However, unlike LightGCN, only one layer is used. This simple architecture can make a recommender system faster than LightGCN.

While LightGCN [13] averages multiple layers in order to obtain information on distant nodes and increase diversity, $\hat{\mathbf{A}}_{\mathrm{APPR}}$ can include potential user-user, user-item, and item-item relationships in the adjacency matrix. Thus, we do not need to increase the number of layers, resulting in the fact that one layer is sufficient. We define our new model as Adjacency Personalized PageRank GCN (hereafter, "APPR-GCN").

4.4 Optimization

In APPR-GCN, the only parameter to optimize is the initial embedding vector. Our loss function is the Bayesian Personalized Ranking (BPR) loss, which is widely used in graph convolutional networks [32]. Let m and $\mathcal{I}_u$ be the number of users and the set of items that interact with user u, respectively. We define BPR loss (L_{BPR}) as follows:

$$L_{BPR} = -\sum_{u=1}^{m} \sum_{i \in \mathcal{I}_u} \sum_{j \notin \mathcal{I}_u} \ln \sigma \left(\hat{r}_{ui} - \hat{r}_{uj} \right) + \lambda \left\| \mathbf{E}^{(0)} \right\|^2, \qquad (15)$$

where $\hat{r}_{ui}$ and $\hat{r}_{uj}$ are recommendation scores, σ is the sigmoid function, and λ is the L_2 regularization parameter used to prevent overfitting. As we detail in Section 5, we sample negative items for each user to compute L_{BPR}.

Table 3: Some statistics on the datasets.

Dataset	#Users	#Items	#Interactions	Density
MovieLens-1M	6,022	3,043	995,154	0.05431
Gowalla	4,381	4,017	216,284	0.01229
Yelp	4,756	4,041	346,011	0.01800
Amazon-Books	5,094	5,279	654,496	0.02434

5 EXPERIMENTS

In this section, we first detail some datasets for our experiments and experimental settings, and then show our experimental results.

5.1 Dataset

To verify the effectiveness of APPR-GCN, we conduct experiments on the following four datasets: MovieLens-1M,[2] Gowalla [5], Yelp,[3] and Amazon-Books,[4] which are widely used in many recommendation works such as NGCF [34], LightGCN [13], and so on.

MovieLens includes movie reviews provided by users. Gowalla is a dataset constructed from a social network service that can share each user's check-in locations. Yelp is a business dataset from Yelp Challenge data. The items are point of interests (POIs), where users can leave reviews and ratings. Amazon-Books is the largest genre in Amazon Review Data [26]. Each dataset has its own statistics such as the number of users, items, interactions, and sparsity. We addressed data with duplicate users and items as the same one. To ensure acceptable computation time, we set lower bounds on the number of interactions for each user and item in the datasets. More specifically, we set these limits to 20, 22, 37, and 58 for MovieLens-1M, Gowalla, Yelp, and Amazon-Books, respectively. We did not set the upper bounds of interactions (*i.e.*, infinite) for all datasets. Given the large number of users and items in Amazon-Books, we implemented this filtering after extracting the top 50,000 users with the highest number of interactions with items and the top 150,000 items with the most interactions with users. Table 3 shows some statistics on each dataset after filtering.

5.2 Experimental Settings

This section explains the evaluation metrics, training settings, and baselines in our comparative experiments. We also optimize the hyperparameters for our experiments.

5.2.1 *Evaluation of Relevance and Diversity.*

We evaluate our APPR-GCN using recall and normalized discounted cumulative gain (nDCG) [14], both of which are widely used to evaluate recommender systems. Specifically, we employ Recall@10 and nDCG@10 to evaluate the relevance of items ranked higher.

Our work aims at developing a recommender system that not only provides higher relevance but also enhances diversity. To achieve this, we also evaluate our method in terms of diversity based on Equation (6). Generally, in an ideal recommender system, items more relevant to each user are ranked higher, while items with diversity are ranked lower.

[2]https://grouplens.org/datasets/MovieLens/1m/
[3]https://www.yelp.com/dataset
[4]https://nijianmo.github.io/amazon/index.html

Table 4: Recommendation accuracy for the four datasets (see Table 3 for details). The top 3 results are shown in bold. "RI" denotes the relative improvement of the score obtained by APPR-GCN. "†" denotes the statistical significance for $p < 0.05$ compared with LightGCN^{L1}. "$L1$" and "$L2$" denote the number of layers in LightGCN (*i.e.*, L in Equation (3)) is "1" and "2", respectively.

Dataset	MovieLens-1M		Gowalla		Yelp		Amazon-Books	
Metrics	Recall@10	nDCG@10	Recall@10	nDCG@10	Recall@10	nDCG@10	Recall@10	nDCG@10
BPR	0.1597	0.2498	0.1663	0.1498	0.0665	0.0663	0.0889	0.1228
NGCF	0.1584	0.2433	0.1614	0.1441	0.0691	0.0685	0.0860	0.1183
LightGCN^{L1}	0.1696	0.2589	0.1784	**0.1618 (2)**	0.0754	0.0755	0.1029	0.1424
LightGCN^{L2}	**0.1736 (2)**	**0.2668 (2)**	0.1754	0.1597	0.0776	**0.0781 (2)**	0.1064	0.1478
UltraGCN	0.1440	0.2093	0.1702	0.1538	**0.0793 (2)**	0.0733	**0.1359 (1)**	**0.1747 (2)**
SimpleX	0.1678	**0.2618 (3)**	0.1772	0.1579	**0.0780 (3)**	0.0768	**0.1229 (3)**	**0.1729 (3)**
IMP-GCN	**0.1727 (3)**	0.2344	**0.1810 (3)**	0.1585	0.0774	**0.0769 (3)**	0.1097	0.1453
SGL	0.1519	0.2020	**0.1842 (2)**	**0.1598 (3)**	0.0734	0.0730	0.1055	0.1395
LightGCL	0.1624	0.2579	0.1505	0.1310	0.0776	0.0762	0.1188	01634
PPR-GCN	0.1622	0.2485	0.1750	0.1587	0.0708	0.0711	0.0934	0.1281
APPR-GCN	**0.1836 (1)**	**0.2817 (1)**	**0.1865 (1)**	**0.1707 (1)**	**0.0808 (1)**	**0.0821 (1)**	**0.1248 (2)**	**0.1768 (1)**
RI over LightGCN^{L1}	8.25% (†)	8.83% (†)	4.54% (†)	5.53% (†)	7.17% (†)	8.68% (†)	21.30% (†)	24.16% (†)

Table 5: The diversity for the four datasets (see Table 3 for details). The top 3 results are shown in bold. APPR-GCN outperforms all baselines on all datasets. "$L1$" and "$L2$" are the same as in Table 4.

Dataset	MovieLens-1M		Gowalla		Yelp		Amazon-Books	
Ranking	11th-15th	16th-20th	11th-15th	16th-20th	11th-15th	16th-20th	11th-15th	16th-20th
BPR	3.68	3.78	2.98	3.00	2.97	2.99	4.45	4.49
NGCF	3.78	3.83	3.01	3.02	2.60	2.60	4.38	4.41
LightGCN^{L1}	5.54	5.63	3.98	4.05	4.23	4.27	6.42	6.48
LightGCN^{L2}	7.33	7.44	4.24	4.29	4.97	5.03	8.87	8.95
UltraGCN	2.02	2.09	4.50	4.46	2.69	2.75	4.21	4.25
SimpleX	2.11	2.13	1.05	1.08	1.32	1.33	2.37	2.42
IMP-GCN	**17.57 (2)**	**17.59 (2)**	**8.64 (3)**	**8.71 (3)**	**11.12 (2)**	**11.17 (2)**	**17.32 (2)**	**17.35 (2)**
SGL	**11.99 (3)**	**11.71 (3)**	**8.98 (2)**	**8.97 (2)**	**9.35 (3)**	**9.30 (3)**	**11.25 (3)**	**11.22 (3)**
LightGCL	2.38	2.50	2.14	2.21	2.12	2.16	2.19	2.25
PPR-GCN	4.72	4.79	3.81	3.84	3.56	3.58	5.84	5.89
APPR-GCN	**41.48 (1)**	**41.04 (1)**	**14.82 (1)**	**14.60 (1)**	**16.18 (1)**	**16.10 (1)**	**32.48 (1)**	**32.56 (1)**

5.2.2 Training Settings. We randomly split the four datasets into 8:1:1 ratio for training, validation, and testing, respectively. We employ mini-batch gradient descent to learn the initial user and item embedding vectors from the training dataset. Then we initialize the embedding vectors using Xavier's method [10]. For each epoch, we computed Recall@10 using the validation data. To avoid overfitting, we set an early stopping criterion to 10 and terminated the training process if we did not observe any improvement in Recall@10. Finally, we measured Recall@10 and nDCG@10 using test data.

To ensure the accuracy of our experiments, we conducted our experiments five times, each with a different random seed for data splitting and optimization. We show the averaged scores from our experiments as the final Recall@10 and nDCG@10. We evaluated diversity with Equation (6) and averaged the results over the five iterations in the same way.

Negative sampling was employed to compute the loss function defined by Equation (15), with samples selected randomly at every epoch. While numerous negative sampling methods have been proposed [7, 21, 31], we performed sampling according to a uniform distribution of items without a user-interaction relationship [8].

We conducted a grid search to optimize hyperparameters and obtained a learning rate of 10^{-3} and an L_2 normalization factor of 10^{-5}. We fixed the embedding size at 64. We employed the Adam optimizer [16] for the mini-batch gradient descent method and set the batch size to 512. The number of graph convolution layers in APPR-GCN is 1. The teleportation probabilities β, α_0, and α_1 in Equations (10) are set to 0.3, 0, and 0.7, respectively.

5.2.3 Baselines. We compare our APPR-GCN with the following state-of-the-arts:

- **BPR** [32]: BPR proposes a generic optimization criterion for personalized ranking, termed BPR-Opt, which is the maximum posterior estimator obtained from a Bayesian analysis of the problem.

- **NGCF** [34]: NGCF is an innovative recommendation framework that explicitly incorporates collaborative signals into model-based CF. NGCF effectively introduces collaborative signals during the embedding process by leveraging the higher-order connections of the user–item interaction graph.
- **LightGCN** [13]: LightGCN is a simplified framework of NGCF, a model that includes solely neighborhood aggregation and consists of two elements: graph convolution and layer merging. This model is easy to learn and demonstrates robust generalization ability and effectiveness.
- **UltraGCN** [24]: Rather than using explicit message passing, UltraGCN directly approximates the limit of infinite-layer graph convolution. This approach allows for the assignment of relevant edge weights and the flexible modification of relative importance between various types of relations.
- **SimpleX** [23]: SimpleX proposes a cosine contrastive loss (CCL) to provide better recommendations. CCL optimizes to maximize the similarity between positive pairs and minimize the similarity between negative pairs.
- **IMP-GCN** [22]: IMP-GCN is a novel recommender model that operates within a subgraph consisting of users with similar interests and their items.
- **SGL** [38]: SGL is a new recommender system that improves the accuracy and robustness of GCNs through self-supervised learning in user and item graphs.
- **LightGCL** [4]: LightGCL is designed to more accurately capture the relationships between users and items from two perspectives: the features obtained through graph propagation via GCN and the features obtained through SVD.

As with APPR-GCN, we fixed the embedding size to 64 and set the batch size to 512 for all of the baselines. We also used the Adam optimizer [16] for the mini-batch gradient descent method. Furthermore, we performed a grid search to optimize the hyperparameters of each model.

In addition to the above state-of-the-arts, we also compare our APPR-GCN with the following variant:

- **PPR-GCN**: PPR-GCN is a combination of LightGCN and Personalized PageRank, using $\hat{\mathbf{A}}_{\text{PPR}}$ as the adjacency matrix for LightGCN. We set α_0 to 0.3.

For all of the baselines and PPR-GCN, we used the same training, validation, and test data as we did for APPR-GCN. For NGCF[5], LightGCN[6], UltraGCN[7], SimpleX[8], IMP-GCN[9], SGL[10], and LightGCL[11], we utilized the source codes released by the authors of these works.

5.3 Experimental Results

5.3.1 Relevance of Recommendation. Table 4 shows the experimental results on the four different datasets.

We compare the performance of our proposed APPR-GCN with the baselines BPR [32], NGCF [34], LightGCN [13], UltraGCN [24], SimpleX [23], IMP-GCN [22], SGL [38], and LightGCL [4]. According to Table 4, APPR-GCN outperforms LightGCN in terms of Recall@10 and nDCG@10 on all datasets. We conducted the experiment five times using different random seeds and performed t-test to verify the effectiveness of our approach. The p-values for all datasets are smaller than 0.05, indicating that our APPR-GCN outperforms LightGCN with statistical significance.

Furthermore, APPR-GCN also achieves the best Recall@10 and nDCG@10 on MovieLens-1M, Gowalla, and Yelp datasets. In Amazon-Books dataset, APPR-GCN is the best in nDCG@10 and the second best in Recall@10. APPR-GCN outperforms the latest baselines, SimpleX [23], IMP-GCN [22], and SGL [38]. We additionally compare our APPR-GCN with the baselines in terms of Recall@20 and nDCG@20. Similar to the results evaluated with Recall@10 and nDCG@10, our APPR-GCN outperforms the baselines.

The adjacency matrix $\hat{\mathbf{A}}_{\text{APPR}}$ in APPR-GCN can represent potential interrelationships among user-user, user-item, and item-item, unlike the adjacency matrix used in LightGCN. Despite having only one layer, this property of the adjacency matrix gives more accurate results than LightGCN. These results indicate that our APPR-GCN outperforms LightGCN in the top-ranked item evaluations.

PPR-GCN, a model that uses $\hat{\mathbf{A}}_{\text{PPR}}$ as the adjacency matrix, is less accurate than LightGCN [13] and APPR-GCN. Both PPR-GCN and APPR-GCN have teleportation probabilities built into their models. However, APPR-GCN, which returns to neighboring nodes, provides better recommendation accuracy than PPR-GCN, which returns to its own node.

5.3.2 Diversity. Table 5 shows the experimental results on the diversity defined by Equation (6) for lower-ranked items.

We recognize the importance of having diversity in the subitems. Thus, we examined items ranked 11th to 15th and 16th to 20th. Our APPR-GCN outperforms all baselines on all datasets, indicating that items with high relevance to users are ranked higher while items with high diversity are ranked lower.

5.4 Training Performance

This section explores the training performance of our APPR-GCN and LightGCN by running our method on a GPU-equipped PC.

To obtain $\hat{\mathbf{A}}_{\text{APPR}}$ more accurately, the inverse matrix needs to be computed, but it only needs to be done once throughout the entire training, which takes very little time compared with the overall training time. However, computing the inverse matrix requires a larger amount of memory.

Table 6 shows the training time (*i.e.*, [seconds/epoch]). Generally, LightGCN can address the adjacency matrix as a sparse one, resulting in fast computation. It seems that the training time of our APPR-GCN is slower than that of LightGCN as the adjacency matrix used in APPR-GCN is dense. However, our APPR-GCN is faster on all four datasets even when L, the number of layers in LightGCN, is at its minimum (*i.e.*, 1). This is because the number of layers in our APPR-GCN is 1, while LightGCN needs to take the average of all layers. Our APPR-GCN has an advantage in that it does not need to address information from distant nodes by stacking layers and it works well using one layer only.

[5]https://github.com/huangtinglin/NGCF-PyTorch
[6]https://github.com/gusye1234/LightGCN-PyTorch
[7]https://github.com/xue-pai/UltraGCN
[8]https://github.com/xue-pai/SimpleX
[9]https://github.com/liufancs/IMP_GCN
[10]https://github.com/wujcan/SGL-Torch
[11]https://github.com/HKUDS/LightGCL

Table 6: Training time [seconds/epoch]. "M-1M" and "A-Books" denote "MovieLens-1M" and "Amazon-Books", respectively. "$L1$" and "$L2$" are the same as in Table 4.

Dataset	M-1M	Gowalla	Yelp	A-Books
LightGCN^{L1}	19.9	2.49	3.29	9.24
LightGCN^{L2}	37.8	3.99	5.55	17.07
APPR-GCN	6.73	1.30	2.24	5.59

6 Conclusion and Future Work

In this work, we proposed Generalized Personalized PageRank (GPPR) and Adjacency Personalized PageRank (APPR) with Graph Convolutional Networks, which extend the framework of Personalized PageRank [19, 28].

Based on this approach, we derived new adjacency matrices, $\hat{\mathbf{A}}_{\text{APPR}}$ and $\hat{\mathbf{A}}_{\text{GPPR}}$. We incorporated $\hat{\mathbf{A}}_{\text{APPR}}$ into LightGCN framework to construct our recommender system, APPR-GCN.

Most of the previous works generate the adjacency matrix by utilizing the users' implicit feedback. However, not all feedback is closely related to the user and item, and observable feedback is limited. Hence, we utilized random walks and GCNs to redefine the potential relevance between users and items. Our APPR-GCN can quantify potential user-item, user-user, and item-item relations that are difficult to identify with LightGCN alone.

GPPR is a model that extends the framework of personalized PageRank, with APPR being a special case of GPPR that retains all its characteristics. We analyzed the features of GPPR from a mathematical perspective and found that the adjacency matrix of GPPR has a minimum eigenvalue greater than -1 and works as a low-pass filter compared with the adjacency matrix used in LightGCN. This property significantly enhances recommendation accuracy. Additionally, we discussed the similarities between GPPR and LightGCN, discovering that averaging multiple layers in LightGCN is equivalent to considering the teleportation probability in GPPR. This indicates that GPPR encompasses the principles of LightGCN.

Experimental results demonstrated that our APPR-GCN improves the recommendation accuracy of top-ranked items while also increasing diversity. As a result, APPR-GCN outperforms LightGCN in terms of Recall and nDCG for top-ranked items, and for lower-ranked items, resulting in more diverse recommendations.

APPR-GCN is faster than LightGCN on all datasets, even though the denser adjacency matrix could potentially slow down computation. This speed is achieved as APPR-GCN uses only one layer to address the over-smoothing problem, eliminating the need to compute averages between layers of vectors. The adjacency matrices defined in GPPR and APPR can capture potential relationships between users and items, as well as user-user and item-item relationships. This capability allows them to achieve high recommendation accuracy without the need to deepen the layers.

In this work, we successfully addressed the over-smoothing problem and developed the fast recommender system with high accuracy and diversity. In future work, we plan to enhance GPPR-GCN and APPR-GCN to further improve recommendation accuracy relevant to each user's preferences.

7 PROOFS

In this section, we prove the theorems in Section 4.2.3. In the following equations, $\mathbf{v}_i$ and μ_i denote an eigenvector and an eigenvalue of $\hat{\mathbf{A}}$, respectively. μ_i is between -1 and 1, inclusive. ξ_i denotes an eigenvalue of $\tilde{\mathbf{A}}$. η_i denotes an eigenvalue of $\hat{\mathbf{A}}_{\text{GPPR}}$. Each $\mathbf{v}_i$ is normalized and orthogonal, thus $\mathbf{v}_i^T\mathbf{v}_j = \delta_{ij}$ holds.

7.1 Proof of Theorem 4.2.1

Proof. First, we show that the eigenvalues of $\tilde{\mathbf{A}}$ are between -1 and 1, inclusive. We consider multiplying $\tilde{\mathbf{A}}$ by $\mathbf{D}^{-\frac{1}{2}}\mathbf{v}_i$ as follows:

$$\tilde{\mathbf{A}}\mathbf{D}^{-\frac{1}{2}}\mathbf{v}_i = \mathbf{D}^{-1}\mathbf{A}\mathbf{D}^{-\frac{1}{2}}\mathbf{v}_i = \mathbf{D}^{-\frac{1}{2}}\hat{\mathbf{A}}\mathbf{v}_i = \mu_i\mathbf{D}^{-\frac{1}{2}}\mathbf{v}_i. \tag{16}$$

Equation (16) indicates the eigenvalues ξ_i are the same as μ_i, so they are between -1 to 1, inclusive. Second, we show that $\beta\rho(\tilde{\mathbf{A}})$ is less than 1. $\rho(\tilde{\mathbf{A}})$ denotes the spectral radius of $\tilde{\mathbf{A}}$ and is defined as follows: $\rho(\tilde{\mathbf{A}}) = \max\{|\xi_1|, |\xi_2|, \cdots, |\xi_{m+n}|\}$. $\rho(\tilde{\mathbf{A}})$ is equal to 1 and $\beta < 1$, so $\beta\rho(\tilde{\mathbf{A}}) < 1$. This indicates that $(\mathbf{I}-\beta\tilde{\mathbf{A}})^{-1}$ can be expanded in a Neumann series as follows: $(\mathbf{I}-\beta\tilde{\mathbf{A}})^{-1} = \sum_{k=0}^{\infty}(\beta\tilde{\mathbf{A}})^k$. By substituting this series into Equation (12) and rearranging by powers of $\tilde{\mathbf{A}}$, we obtain Equation (13). □

7.2 Proof of Theorem 4.2.2

Proof. The sum of the coefficients of $\tilde{\mathbf{A}}^l$ in Equation (13) is the value obtained by substituting 1 (a scalar) for $\tilde{\mathbf{A}}$, which can be calculated as follows:

$$\left(\sum_{k=0}^{\infty}\beta^k\right)\left(\sum_{l=0}^{\infty}\alpha_l\right) = \left(\frac{1}{1-\beta}\right)\cdot(1-\beta) = 1. \qquad \square$$

7.3 Proof of Theorem 4.2.3

Proof. By multiplying $\hat{\mathbf{A}}_{\text{GPPR}}$ by $\mathbf{v}_i$, we see that the eigenvectors of $\hat{\mathbf{A}}_{\text{GPPR}}$ are the same as those of $\hat{\mathbf{A}}$, and the eigenvalues are shown as follows:

$$\hat{\mathbf{A}}_{\text{GPPR}}\mathbf{v}_i = \left(\sum_{k=0}^{\infty}\beta^k\tilde{\mathbf{A}}^k\right)\left(\sum_{l=0}^{\infty}\alpha_l\tilde{\mathbf{A}}^l\right)\mathbf{v}_i = \left(\sum_{k=0}^{\infty}\beta^k\mu_i^k\right)\left(\sum_{l=0}^{\infty}\alpha_l\mu_i^l\right)\mathbf{v}_i.$$

Maximum Eigenvalue. The maximum eigenvalue occurs when $\mu_i = 1$. By substituting 1 for μ_i, we show that the maximum eigenvalue is 1 as follows: $\left(\sum_{k=0}^{\infty}\beta^k\right)\left(\sum_{l=0}^{\infty}\alpha_l\right) = \left(\frac{1}{1-\beta}\right)\cdot(1-\beta) = 1$.

Minimum Eigenvalue. We show that the minimum eigenvalue is greater than -1. The minimum eigenvalue is obtained when μ_i is in the range $-1 \leqq \mu_i < 0$. We represent the eigenvalues as $\sum_{k=0}^{\infty}\gamma_k\mu_i^k$. γ_k satisfies $\sum_{k=0}^{\infty}\gamma_k = 1$ and $0 < \gamma_k < 1$. We transform the series as follows:

$$\sum_{k=0}^{\infty}\gamma_k\mu_i^k = \sum_{k=0}^{\infty}\gamma_{2k}\mu_i^{2k} + \sum_{k=0}^{\infty}\gamma_{2k+1}\mu_i^{2k+1} > -\sum_{k=0}^{\infty}\gamma_{2k}\mu_i^{2k} + \sum_{k=0}^{\infty}\gamma_{2k+1}\mu_i^{2k+1}$$
$$= -\left(\sum_{k=0}^{\infty}\gamma_{2k}(-\mu_i)^{2k} + \sum_{k=0}^{\infty}\gamma_{2k+1}(-\mu_i)^{2k+1}\right) = -\left(\sum_{k=0}^{\infty}\gamma_k(-\mu_i)^k\right)$$

The range of μ_i is $-1 \leqq \mu_i < 0$, so we can show that the minimum eigenvalue is greater than -1 because $-\left(\sum_{k=0}^{\infty}\gamma_k(-\mu_i)^k\right) > -1$ holds. □

References

[1] Upasna Bhandari, Kazunari Sugiyama, Anindya Datta, and Rajni Jindal. 2013. Serendipitous Recommendation for Mobile Apps Using Item-Item Similarity Graph. In *Proceedings of the 9th Asia Information Retrieval Societies Conference (AIRS '13)*. 440–451.

[2] Aleksandar Bojchevski, Oleksandr Shchur, Daniel Zügner, and Stephan Günnemann. 2018. NetGAN: Generating Graphs via Random Walks. In *Proceedings of the 35th International Conference on Machine Learning, (ICML '18)*. 609–618.

[3] Sergey Brin and Lawrence Page. 1998. The Anatomy of a Large-Scale Hypertextual Web Search Engine. *Computer Networks* 30, 1–7 (1998), 107–117.

[4] Xuheng Cai, Chao Huang, Lianghao Xia, and Xubin Ren. 2023. LightGCL: Simple Yet Effective Graph Contrastive Learning for Recommendation. In *The Eleventh International Conference on Learning Representations, (ICLR '23)*.

[5] Eunjoon Cho, Seth A. Myers, and Jure Leskovec. 2011. Friendship and Mobility: User Movement in Location-based Social Networks. In *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '11)*. 1082–1090.

[6] Yue Deng. 2022. Recommender Systems Based on Graph Embedding Techniques: A Review. *IEEE Access* 10 (2022), 51587–51633.

[7] Jingtao Ding, Yuhan Quan, Xiangnan He, Yong Li, and Depeng Jin. 2019. Reinforced Negative Sampling for Recommendation with Exposure Data. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence (IJCAI-19)*. 2230–2236.

[8] Jingtao Ding, Yuhan Quan, Quanming Yao, Yong Li, and Depeng Jin. 2020. Simplify and Robustify Negative Sampling for Implicit Collaborative Filtering. In *Proceedings of the 34th International Conference on Neural Information Processing Systems (NeurIPS '20)*.

[9] Chen Gao, Yu Zheng, Nian Li, Yinfeng Li, Yingrong Qin, Jinghua Piao, Yuhan Quan, Jianxin Chang, Depeng Jin, Xiangnan He, and Yong Li. 2023. A Survey of Graph Neural Networks for Recommender Systems: Challenges, Methods, and Directions. *ACM Transactions on Recommender Systems (TORS)* 1, 1 (2023), 3:1–3:51.

[10] Xavier Glorot and Yoshua Bengio. 2010. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the 13th International Conference on Artificial Intelligence and Statistics (AISTATS '10)*, Vol. 9. 249–256.

[11] Aditya Grover and Jure Leskovec. 2016. Node2vec: Scalable Feature Learning for Networks. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '16)*. 855–864.

[12] Li He, Xianzhi Wang, Dingxian Wang, Haoyuan Zou, Hongzhi Yin, and Guandong Xu. 2023. Simplifying Graph-Based Collaborative Filtering for Recommendation. In *Proceedings of the 16th ACM International Conference on Web Search and Data Mining (WSDM '23)*. 60–68.

[13] Xiangnan He, Kuan Deng, Xiang Wang, Yan Li, YongDong Zhang, and Meng Wang. 2020. 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 (SIGIR '20)*. 639–648.

[14] Kalervo Järvelin and Jaana Kekäläinen. 2002. Cumulated Gain-Based Evaluation of IR Techniques. *ACM Transactions on Information Systems (TOIS)* 20, 4 (2002), 422–446.

[15] Marius Kaminskas and Derek Bridge. 2016. Diversity, Serendipity, Novelty, and Coverage: A Survey and Empirical Analysis of Beyond-Accuracy Objectives in Recommender Systems. *ACM Transactions on Interactive Intelligent System (TiiS)* 7, 1 (2016), 2:1–2:42.

[16] Diederik P. Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *Proceedings of the 3rd International Conference on Learning Representations (ICLR '15)*.

[17] Thomas N. Kipf and Max Welling. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *Proceedings of the 5th International Conference on Learning Representations, (ICLR '17)*.

[18] Thomas N. Kipf and Max Welling. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *Proceedings of the 5th International Conference on Learning Representations (ICLR '17)*.

[19] Johannes Klicpera, Aleksandar Bojchevski, and Stephan Günnemann. 2019. Predict then Propagate: Graph Neural Networks meet Personalized PageRank. In *Proceedings of the 7th International Conference on Learning Representations (ICLR '19)*.

[20] Yehuda Koren. 2008. Factorization Meets the Neighborhood: A Multifaceted Collaborative Filtering Model. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '08)*. 426–434.

[21] Defu Lian, Qi Liu, and Enhong Chen. 2020. Personalized Ranking with Importance Sampling. In *Proceedings of the Web Conference (WWW '20)*. 1093–1103.

[22] Fan Liu, Zhiyong Cheng, Lei Zhu, Zan Gao, and Liqiang Nie. 2021. Interest-aware Message-Passing GCN for Recommendation. In *Proceedings of the Web Conference 2021 (WWW '21)*. 1296–1305.

[23] Kelong Mao, Jieming Zhu, Jinpeng Wang, Quanyu Dai, Zhenhua Dong, Xi Xiao, and Xiuqiang He. 2021. SimpleX: A Simple and Strong Baseline for Collaborative Filtering. In *Proceedings of the 30th ACM International Conference on Information and Knowledge Management (CIKM '21)*. 1243–1252.

[24] Kelong Mao, Jieming Zhu, Xi Xiao, Biao Lu, Zhaowei Wang, and Xiuqiang He. 2021. UltraGCN: Ultra Simplification of Graph Convolutional Networks for Recommendation. In *Proceedings of the 30th ACM International Conference on Information and Knowledge Management (CIKM '21)*. 1253–1262.

[25] Klaus Nehring and Clemens Puppe. 1999. A Theory of Diversity. *Electronic Notes in Discrete Mathematics* 2 (1999), 180–181.

[26] Jianmo Ni, Jiacheng Li, and Julian J. McAuley. 2019. Justifying Recommendations using Distantly-Labeled Reviews and Fine-Grained Aspects. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP '19)*. 188–197.

[27] Mathias Niepert, Mohamed Ahmed, and Konstantin Kutzkov. 2016. Learning Convolutional Neural Networks for Graphs. In *Proceedings of the 33rd International Conference on Machine Learning, (ICML '16)*, Vol. 48. 2014–2023.

[28] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. 1999. *The PageRank Citation Ranking: Bringing Order to the Web.* Technical Report SIDL-WP-1999-0120. Stanford Digital Library Technologies Project.

[29] Bibek Paudel, Fabian Christoffel, Chris Newell, and Abraham Bernstein. 2016. Updatable, Accurate, Diverse, and Scalable Recommendations for Interactive Applications. *ACM Transactions on Interactive Intelligent Systems (TiiS)* 7, 1 (2016), 1:1–1:34.

[30] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 2014. DeepWalk: Online Learning of Social Representations. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '14)*. 701–710.

[31] Steffen Rendle and Christoph Freudenthaler. 2014. Improving Pairwise Learning for Item Recommendation from Implicit Feedback. In *Proceedings of the 7th ACM International Conference on Web Search and Data Mining (WSDM '14)*. 273–282.

[32] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian Personalized Ranking from Implicit Feedback. In *Proceedings of the 25th Conference on Uncertainty in Artificial Intelligence (UAI '09)*. 452–461.

[33] Barry Smyth and Paul McClave. 2001. Similarity vs. Diversity. In *Proceedings of the 4th International Conference on Case-Based Reasoning (ICCBR '01)*. 347–361.

[34] Xiang Wang, Xiangnan He, Meng Wang, Fuli Feng, and Tat-Seng Chua. 2019. Neural Graph Collaborative Filtering. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '19)*. 165–174.

[35] Zihong Wang, Yingxia Shao, Jiyuan He, Jinbao Liu, Shitao Xiao, Tao Feng, and Ming Liu. 2023. Diversity-aware Deep Ranking Network for Recommendation. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management (CIKM '23)*. 2564–2573.

[36] Hiroshi Wayama and Kazunari Sugiyama. 2023. The Effectiveness of Quantum Random Walk Model in Recommender Systems. In *Proceedings of the 2023 ACM SIGIR International Conference on the Theory of Information Retrieval (ICTIR '23)*. 225–234.

[37] Felix Wu, Amauri H. Souza Jr., Tianyi Zhang, Christopher Fifty, Tao Yu, and Kilian Q. Weinberger. 2019. Simplifying Graph Convolutional Networks. In *Proceedings of the 36th International Conference on Machine Learning (ICML '19)*. 6861–6871.

[38] Jiancan Wu, Xiang Wang, Fuli Feng, Xiangnan He, Liang Chen, Jianxun Lian, and Xing Xie. 2021. Self-supervised Graph Learning for Recommendation. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '21)*. 726–735.

[39] Shiwen Wu, Fei Sun, Wentao Zhang, Xu Xie, and Bin Cui. 2022. Graph Neural Networks in Recommender Systems: A Survey. *ACM Computing Surveys (CSUR)* 55, 5, Article 97:1-97:37 (2022).

[40] Eva Zangerle and Christine Bauer. 2022. Evaluating Recommender Systems: Survey and Framework. *ACM Computing Surveys (CSUR)* 55, 8 (2022), 170:1–170:38.

[41] Dan Zhang, Yangliao Geng, Wenwen Gong, Zhongang Qi, Zhiyu Chen, Xing Tang, Ying Shan, Yuxiao Dong, and Jie Tang. 2024. RecDCL: Dual Contrastive Learning for Recommendation. In *Proceedings of the Web Conference 2024 (WWW '24)*. 3655–3666.

[42] Guanjie Zheng, Fuzheng Zhang, Zihan Zheng, Yang Xiang, Nicholas Jing Yuan, Xing Xie, and Zhenhui Li. 2018. DRN: A Deep Reinforcement Learning Framework for News Recommendation. In *Proceedings of the 2018 World Wide Web Conference (WWW '18)*. 167–176.

[43] Cai-Nicolas Ziegler, Sean M. McNee, Joseph A. Konstan, and Georg Lausen. 2005. Improving Recommendation Lists through Topic Diversification. In *Proceedings of the 14th International Conference on World Wide Web (WWW '05)*. 22–32.