Improving Recommendation Diversity with Architectural Modifications to DGRec

João Pedro Oliveira Batisteli
IMScience Lab
Pontifícia Universidade Católica
de Minas Gerais (PUC Minas)
Belo Horizonte, Brazil
joao.batisteli@sga.pucminas.br

***Abstract*—Graph Neural Networks (GNNs) have achieved significant success in recommendation systems by effectively modeling complex user-item relationships. A common limitation of many GNN-based recommenders, however, is their tendency to recommend a narrow set of popular items, resulting in limited diversity. The Diversified GNN-based Recommender System (DGRec) seeks to mitigate this issue. In this work, we propose novel modifications to the DGRec architecture to further promote recommendation diversity, specifically introducing a gated attention mechanism and additional transformations within the graph processing modules. We evaluate these modifications on the large-scale TaoBao e-commerce dataset, demonstrating improvements in Coverage, a key metric for diversity, and analyze the trade-off with accuracy. Our findings highlight the critical balance between diversity and accuracy in recommender system design.**

***Index Terms*—Recommendation Systems; Graph Neural Networks;**

I. Introduction

Recommender systems are essential components of online platforms, empowering users to efficiently navigate vast amounts of data and discover items tailored to their individual needs and preferences. Given their critical role in user experience, it is essential to improve recommender systems, minimizing the presentation of irrelevant recommendations and maximizing the delivery of valuable and personalized content.

Although the main focus of recommenders is maximizing accuracy, studies have increasingly emphasized the importance of recommendation diversity [1], [2]. Providing diverse recommendations is crucial for enhancing user experience by fostering exploration of new items, mitigating the formation of filter bubbles that limit exposure to diverse perspectives, and ultimately increasing user satisfaction. However, achieving high levels of diversity often necessitates a trade-off with accuracy, as recommending less predictable items can decrease the likelihood of recommending items perfectly aligned with immediate user preferences. This inherent tension between diversity and accuracy poses a significant challenge for recommender system design.

Graphs are versatile structures that offer a powerful way to represent diverse forms of data and information across numerous domains. Their flexibility stems from their ability to capture relationships and interactions between entities. This makes them applicable to a wide range of problems, including: representing mathematical equations and their underlying relationships [3], modeling the steps and dependencies within algorithms, enabling analysis and optimization [4], capturing the complex connections and dynamics of social networks, facilitating understanding of community structures [5], representing user-item interactions in recommender systems, allowing for personalized recommendations based on user preferences and item similarities [1], and even encoding the spatial relationships and hierarchical structures within images, enabling tasks like image segmentation and object recognition [6]–[8].

Traditional recommendation methods often struggle to effectively capture user-item interaction data's complex relationships and high-order connectivity. Graph Neural Networks (GNNs) offer a compelling solution to this challenge, demonstrating remarkable success in modeling complex interactions within graph-structured data [9]. GNNs improve recommendation performance by learning nuanced representations through information propagation across the graph, which has garnered attention for GNN-based recommenders.

To tackle the issue of limited recommendation diversity in current methods, the Diversified GNN-based Recommender System (DGRec) [1] was introduced, utilizing graph representations to model the recommendation problem. The DGRec comprises three key modules designed to address distinct aspects of graph-based recommendation. First, submodular neighbor selection identifies a diverse subset of neighboring nodes for aggregation, thereby enhancing the diversity of learned representations. Second, layer attention applies attention weights to the outputs of each GNN layer, mitigating the over-smoothing problem and effectively leveraging high-order graph connections. Finally, loss reweighting adjusts the training process to emphasize items from long-tail categories, aiming to achieve a balance between recommendation accuracy and diversity.

In this work, we propose two key modifications to the DGRec architecture: enhancing node feature transformations and refining the model's attention mechanism. These modifications directly impact the recommendation process by improving both information aggregation and feature representation. Specifically, they enable the model to better differentiate between various types of user-item interactions and learn more nuanced representations, thereby facilitating the recommendation of a more diverse set of items and promoting user exploration and discovery.

This work is organized as follows: Section III describes the most important concepts needed for understanding the proposed method. Section IV presents our proposed graph creation pipeline and our novel readout method. Section V details the experiments conducted and offers a comparative analysis of the proposed approach with the existing state-of-the-art method. Finally, Section VI concludes the paper and suggests potential areas for future work.

II. Related Works

This section provides an overview of related work that employs Graph Neural Networks for recommendation tasks, highlighting key approaches

[10] extend collaborative filtering to the graph domain by introducing Neural Graph Collaborative Filtering (NGCF). This method directly aggregates information from neighbors in the bipartite graph while explicitly embedding collaborative signals into the learned representations.

LightGCN [11] builds on NGCF by simplifying its architecture, removing computational overheads such as linear transformations and non-linear activations. This streamlined design not only improves performance but also significantly reduces training time.

[5] presents a comprehensive survey on the application of Graph Neural Networks in social network recommendation, highlighting their ability to model complex relationships, capture high-order connectivity, and improve recommendation quality through innovative graph-based techniques.

[12] introduce Dual Graph Neural Network (DualGNN), a novel framework designed for micro-video recommendation. This approach operates on two graph structures: the user-microvideo bipartite graph and the user co-occurrence graph. By leveraging user correlations, DualGNN collaboratively uncovers personalized fusion patterns, enhancing the ability to capture nuanced user preferences and improving recommendation accuracy in micro-video platforms.

[13] proposes an approach that integrates item category and interaction time information with a multi-layer Graph Convolution Network (GCN) to create multi-dimensional, fine-grained item representations. Additionally, the authors design a temporal self-attention network to effectively capture the dynamic nature of user preferences over time, enabling more accurate next-item recommendations. This method enhances the modeling of temporal and categorical factors, improving the system's ability to predict user behavior.

III. Background

A. Graph Neural Networks

Graph Neural Networks (GNNs) can be broadly classified into two categories: (i) spatial and (ii) spectral. Spatial GNNs operate directly on the graph structure, deriving node (and potentially edge) representations from their neighborhood information. In contrast, spectral GNNs work in the graph's spectral domain, leveraging the eigenvectors of the graph Laplacian matrix as a basis for graph data representation.

Each GNN layer takes as input the node feature vectors $\{h_i \in \mathbb{R}^d \mid i \in \mathcal{V}\}$ for the node set $\mathcal{V}$ and edge feature vectors $\{w_{ij} \in \mathbb{R}^d \mid (i,j) \in \mathcal{E}\}$ for the set of edges $\mathcal{E}$. Each layer produces a new node representation $\{h_i^{'} \in \mathbb{R}^{d^{'}} \mid i \in \mathcal{V}\}$, applying the same parametric function to each node. This is based on its neighbors $\mathcal{N}_i = \{j \in \mathcal{V} \mid (i,j) \in \mathcal{E})\}$ and the edges incident to it. The generic form is given by Equation 1,

$$h_i' = f_\theta\left(h_i, aggregate(h_j, w_{ij}' \mid j \in \mathcal{N}_i)\right) \tag{1}$$

in which $aggregate$ is a permutation invariant function, and f_θ is a parametric function. The updated edge feature, denoted as w_{ij}', is defined by the Equation 2,

$$w_{ij}' = g_\theta(h_i, h_j, w_{ij}) \tag{2}$$

in which $g_{\theta'}$ is another parametric function. Each step of node updating is also referred to as the '*message passing step*', representing the process in which vertices transmit information to their neighboring vertices.

IV. Diversified GNN-based Recommender System

This section presents the Diversified GNN-based Recommender System (DGRec) [1] model. The DGRec framework incorporates three key components: submodular neighbor selection, layer attention, and loss reweighting.

A. Training Framework

The model input is a bipartite graph $G = (U, I, E)$, where U and I are the disjoint sets of user and item nodes, respectively, and E represents the set of edges encoding user-item interactions.

Each user and item ID is passed through an embedding layer, generating a corresponding vector that serves as the node's feature representation. This vector is shown in Equation 3.

$$\mathbf{E}^{(0)} = \left(\mathbf{e}_1^0, \mathbf{e}_2^0, ..., \mathbf{e}_{|U|+|I|}^0\right) \tag{3}$$

where $\mathbf{e}^0 \in \mathbb{R}^d$ denotes the d-dimensional vector space of initial node embeddings. These initial embeddings ($\mathbf{e}_i^0$) serve as the input to the GNN for subsequent information aggregation.

The convolution operation used in this work is Light Graph Convolution (LGC) [11]. LGC simplifies the standard graph convolution operation by discarding feature transformations and nonlinear activations, performing direct aggregation of neighbor features, as shown in Equation 8.

$$\mathbf{e}_{\mathcal{T}}^{\ell+1} = \sum_{i \in \mathcal{N}_{\mathcal{T}}} \frac{1}{\sqrt{|\mathcal{N}_I|}\sqrt{|\mathcal{N}_U|}} \mathbf{e}_{\mathcal{T}}^{\ell} \tag{4}$$

where $\mathbf{e}_{\mathcal{T}}^{\ell}$ are the embedding at layer ℓ for nodes of type $\mathcal{T}$, where $\mathcal{T}$ can be the set of users (U) or the set of items (I). The term $\frac{1}{\sqrt{|\mathcal{N}_I|}\sqrt{|\mathcal{N}_U|}}$ is the normalization factor. $\mathcal{N}_I$ and $\mathcal{N}_U$ represent the neighborhoods of item and user nodes, respectively. The user neighborhood set ($\mathcal{N}_U$) is determined via a submodular neighbor selection function [1].

Each LGC layer produces an embedding vector for each user and item node, capturing information from a different receptive field. Layer attention is then applied to these embeddings to obtain the final user and item representations, as shown in Equation 5.

$$\mathbf{e}_{\mathcal{T}} = \texttt{Layer\_Attention}(e_{\mathcal{T}}^{(0)}, e_{\mathcal{T}}^{(1)}, ..., e_{\mathcal{T}}^{(L)}) \tag{5}$$

where $\mathcal{T}$ is the node type and L is the number of model layers.

After obtaining the final user and item embeddings, e_u and e_i, the interaction score between user u and item i is determined by the dot product of the two vectors. We employ the Bayesian Personalized Ranking (BPR) loss [14] to train the model. For each observed positive interaction (u, i), a negative item j is randomly sampled, and the BPR loss is computed to encourage higher scores for positive items compared to negative ones. The loss function is defined in Equation 6.

$$\mathcal{L} = \sum_{(u,i)\in E} w_{C(i)} \mathcal{L}_{bpr}(u, i, j) + \lambda ||\Theta||_2^2 \tag{6}$$

where $w_{C(i)}$ is the weight for each sample based on its category, λ is the regularization factor and j is a randomly sampled negative item.

B. Submodular Neighbor Selection

While GNN message passing offers powerful aggregation capabilities, it can also suffer from aggregating information from irrelevant neighbors, potentially leading to oversmoothing and loss of distinct node representations. To address this, the submodular neighbor selection module proposed in [1] aims to select a subset of neighbors that maximizes diversity and information content for aggregation.

Facility location function [15] is used to select a set of diverse neighbors for aggregation. Given a user node u with neighbor set $\mathcal{N}_u$, the objective is to find a subset $\mathcal{S}_u \subseteq \mathcal{N}_u$ that maximizes the sum of the maximum similarities between each item $i \in \mathcal{N}_u \setminus \mathcal{S}_u$ and the items in $\mathcal{S}_u$, formally expressed as $max_{i' \in \mathcal{S}_u} \operatorname{sim}(i, i') \, \forall \; i \in \mathcal{N}_u \setminus \mathcal{S}_u$. The facility location function is defined by Equation 7.

$$f(\mathcal{S}_u) = \sum_{i \in \mathcal{N}_u \setminus \mathcal{S}_u} max_{i' \in \mathcal{S}_u} \operatorname{sim}(i, i') \tag{7}$$

where $\mathcal{S}_u$ denotes the subset of neighbors selected for user u. The similarity between items i and i', denoted as $\operatorname{sim}(i, i')$, is calculated using a Gaussian kernel with kernel width σ^2 (Equation 8). The Gaussian kernel is a common choice for similarity measures due to its ability to capture local relationships between items.

$$\operatorname{sim}(i, i') = exp(-\frac{\|e_i - e_{i'}\|^2}{\sigma^2}) \tag{8}$$

A widely used method for maximizing submodular functions is the greedy algorithm. Starting with an empty set $\mathcal{S}_u := \emptyset$, this algorithm iteratively adds the item $i \in I \setminus \mathcal{S}_u$ that provides the largest marginal gain. While not guaranteed to find the optimal solution, this greedy approach provides a $(1 - 1/e)$ approximation guarantee for monotone submodular functions. This process is defined by the Equation 9.

$$\mathcal{S}_u \leftarrow \mathcal{S}_u \cup i^*,$$
$$i^* = \arg\max_{\mathrm{i} \in \mathcal{N}_\mathrm{u} \setminus \mathcal{S}_\mathrm{u}} [f(\mathcal{S}_u \cup i) - f(\mathcal{S}_u)] \tag{9}$$

After k steps, the subset obtained is then used for aggregation.

C. Loss Reweighting

The number of items within each category is highly imbalanced and follows the power-law distribution. Training the model by directly optimizing the mean loss over all samples would leave the training of long-tail categories imperceptible. To address this issue of class imbalance, the authors in [1] proposed a category-aware loss reweighting scheme during the training process.

Each sample (u, i) is reweighted based on the category number of items. This is achieved by incorporating a weight $w_{C(i)}$ (Equation 10) into the loss function. This reweighting gives more importance to samples from under-represented categories.

$$w_{C(i)} = \frac{1 - \beta}{1 - \beta^{|C(i)|}} \tag{10}$$

where β is the hyperparameter that decides the weight.

D. Proposed Changes

1) Additional Transformations: In tasks involving graphs and GNNs, the diversity of node representations plays a crucial role in determining the success of the model. A richer variety of node representations, capturing different aspects of each node's role and connectivity within the graph, can lead to improvements in downstream tasks.

Building upon the insights from [16] and [17], we also employ a Graph Processing Module (GP Module) [17] designed to enhance the diversity of node representations. This module incorporates linear and normalization layers both before and after each graph convolution. Pre-convolution linear and normalization layers project the node features into a shared domain, while post-convolution layers refine the learned representations. This sequence of layers (linear-norm-convolution-norm-linear-norm) promotes a richer variety of features, contributing to improved performance. The original Equation 4 is consequently modified as shown in Equation 11

$$\mathbf{e}_{\mathcal{T}}^{\ell+1} = BN(W_2(BN(\sum_{i \in \mathcal{N}_{\mathcal{T}}} \frac{1}{\sqrt{|\mathcal{N}_I|}\sqrt{|\mathcal{N}_U|}} BN(W_1 \mathbf{e}_{\mathcal{T}}^{\ell})))) \tag{11}$$

where W_1 and W_2 are linear transformations, and BN stands for Batch Normalization. To reduce the number of parameters, the same W_1, W_2, and BN layers are shared before and after each convolution.

2) Gated Attention: A key challenge is to aggregate information in a way that preserves feature diversity while also ensuring robust representations for items with low prevalence in the dataset. The initial DGRec implementation using Layer Attention [1] exhibited limitations due to its simplicity. To address this, we adopted a gated attention mechanism [18].

The gated attention (GA) mechanism presents a computationally efficient approach to weighting and combining layer-wise representations. Unlike traditional multi-head attention mechanisms [19] that require substantial computational resources, this approach introduces a lightweight alternative that maintains model expressiveness while significantly reducing memory requirements.

GA combines two key components: a traditional attention mechanism and a gating function. The attention component learns to weigh different layers based on their relevance, while the gating component controls information flow by learning to selectively pass or block features from each layer. This dual-control system allows the model to be both selective and expressive in how it combines information across layers.

Let $E = [e^1, e^2, ..., e^L]$ be the set of layer representations of each user/item, where L is the number of layers and each $e^i \in \mathbb{R}^d$ represents the embedding at layer i with dimension d. The Attention Scoring is obtained by applying the Equation 12.

$$s^i = \mathbf{a}^T (W\mathbf{e}^i) \tag{12}$$

where $W \in \mathbb{R}^{d\times d}$ is a learnable projection matrix and $\mathbf{a} \in \mathbb{R}^d$ is the attention vector. The gate computation is done by following the Equation 13.

$$g^i = \sigma(W_g\, e^i) \tag{13}$$

where $W_g \in \mathbb{R}^{d\times 1}$ is the gating weight matrix and σ represents the sigmoid function. The Equation 14 describes how the weights are combined.

$$w^i = \text{softmax}(s^i) \cdot g_i \tag{14}$$

Finally, the final representation is obtained by applying the readout function in Equation 15.

$$\mathbf{y} = \sum_{i=1}^{L} w^i \mathbf{e}^i \tag{15}$$

V. Experiments

A. Dataset and Graph Construction

The TaoBao dataset [2] contains user behavior on TaoBao platform. The dataset contains multiple relationships between users and items, like clicking, purchasing, adding items to carts, and item favoring.

Following the approach of [1], we represent the dataset as a bipartite graph with user and item nodes. This bipartite structure provides a clear and intuitive representation of the user-item relationships. All observed interactions are treated as positive samples, reflecting the direct connection between users and the items they interact with. These connections are represented by edges of type *"rated"* (user to item) and *"rated by"* (item to user). This explicit modeling of user-item interactions through distinct edge types allows the graph neural network to learn patterns and relationships within the data. Figure 1 illustrates an example of the constructed graph.

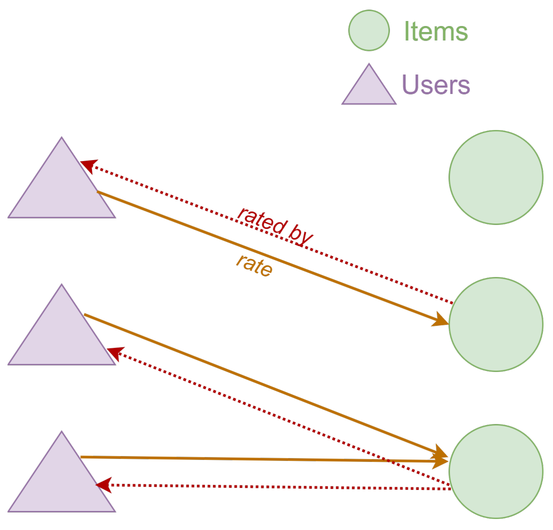

Fig. 1. Graph Example

B. Metrics

The metrics adopted followed those used in [1], which are:

- **Hit Ratio (HR)**: Measures the proportion of relevant items successfully recommended to users within a specified top-N list. A higher HR reflects greater recommendation accuracy, indicating the model's ability to prioritize items of interest effectively.
- **Recall**: Recall quantifies the fraction of relevant items retrieved out of the total number of relevant items in the dataset. It highlights the model's capability to identify as many relevant items as possible, thereby assessing its comprehensiveness in recommendations.
- **Coverage**: Evaluates the diversity of recommendations by determining the fraction of unique items recommended across all users. Higher coverage demonstrates the system's ability to address varied user preferences and expand the range of items presented.

To provide a comprehensive analysis, the Top-100 and Top-300 retrieval results are reported for all three metrics.

C. Quantitative analysis

Table I presents a comparative analysis of the original DGRec model [1] and our two proposed variants. DGRec$_T$ incorporates additional transformations within the graph processing modules, while DGRec$_A$ features a modified attention mechanism, replacing the standard attention layer with a gated attention mechanism. This comparison allows us to evaluate the impact of these modifications on performance. To ensure reproducibility and fair comparison, the DGRec results presented in Table I were obtained by re-running the original experiments under identical settings

TABLE I
OVERALL COMPARISON ON TAOBAO DATASET.

Method	Recall@100	Recall@300	HR@100	HR@300	Coverage@100	Coverage@300
DGRec [1]	**0.0455**	**0.0916**	**0.2946**	**0.4714**	39.5123	90.2758
DGRec$_T$	0.0204	0.0401	0.1569	0.2727	52.3410	**126.1431**
DGRec$_A$	0.0037	0.0116	0,0398	0.1075	**54.3406**	114.1998

Both proposed models achieved higher Coverage scores at both top-100 (@100) and top-300 (@300) recommendations compared to the original DGRec, indicating a greater ability to recommend a wider range of items and thus demonstrating improved recommendation diversity. This suggests that our modifications effectively address the issue of recommending only a small subset of popular items.

Interestingly, the two variants exhibit different strengths in terms of Coverage. DGRec$_T$ achieves higher Coverage@100, indicating a better ability to recommend a diverse set of the top 100 items. However, DGRec$_A$ surpasses DGRec$_T$ in Coverage@300, suggesting it more effectively recommends a broader range of items within the top 300.

While both proposed models demonstrated gains in Coverage, indicating improved recommendation diversity, they exhibited a trade-off in accuracy, as measured by Hit Ratio and Recall. These metrics showed a decrease compared to the original DGRec. This suggests that by broadening the range of recommended items, the proposed models also began recommending items that users had not interacted with or purchased, thus impacting accuracy.

VI. CONCLUSION

This work introduces architectural modifications to the DGRec model [1] with the goal of enhancing the diversity of recommended items. Specifically, we incorporate a gated attention mechanism and additional transformations to augment the model's capacity to recommend a broader range of items, thereby mitigating potential biases towards popular or frequently interacted-with items.

Evaluation on the TaoBao dataset demonstrated that the proposed architectural changes effectively enhanced recommendation diversity, as measured by Coverage. This increase in diversity, however, was accompanied by a decrease in accuracy metrics, suggesting that the model began recommending a wider range of items, including some less relevant to individual user preferences. This observation emphasizes the need to carefully navigate the trade-off between recommending a diverse set of items and maintaining high accuracy.

Future work will focus on exploring architectural changes to achieve a better balance between recommendation diversity and accuracy. Furthermore, we will investigate the use of multigraph representations to explicitly model different types of user-item relationships, such as clicks, purchases, adds to cart, and ratings, treating each interaction type independently. This granular approach promises to provide a richer understanding of user behavior and lead to more effective recommendations.

REFERENCES

[1] L. Yang, S. Wang, Y. Tao, J. Sun, X. Liu, P. S. Yu, and T. Wang, "Dgrec: Graph neural network for recommendation with diversified embedding generation," in *Proceedings of the Sixteenth ACM International Conference on Web Search and Data Mining*, 2023, pp. 661–669.

[2] Y. Zheng, C. Gao, L. Chen, D. Jin, and Y. Li, "Dgcn: Diversified recommendation with graph convolutional networks," in *Proceedings of the Web Conference 2021*, 2021, pp. 401–412.

[3] V. P. Dwivedi, C. K. Joshi, A. T. Luu, T. Laurent, Y. Bengio, and X. Bresson, "Benchmarking graph neural networks," *Journal of Machine Learning Research*, vol. 24, no. 43, pp. 1–48, 2023.

[4] P. Veličković and C. Blundell, "Neural algorithmic reasoning," *Patterns*, vol. 2, no. 7, 2021.

[5] X. Li, L. Sun, M. Ling, and Y. Peng, "A survey of graph neural network based recommendation in social networks," *Neurocomputing*, vol. 549, p. 126441, 2023.

[6] L. Najman, J. Cousty, and B. Perret, "Playing with kruskal: algorithms for morphological trees in edge-weighted graphs," in *International Symposium on Mathematical Morphology and Its Applications to Signal and Image Processing*. Springer, 2013, pp. 135–146.

[7] S. Guimarães, Y. Kenmochi, J. Cousty, Z. Patrocinio, and L. Najman, "Hierarchizing graph-based image segmentation algorithms relying on region dissimilarity," *Mathematical Morphology-Theory and Applications*, vol. 2, no. 1, pp. 55–75, 2017.

[8] J. Cousty and L. Najman, "Incremental algorithm for hierarchical minimum spanning forests and saliency of watershed cuts," in *International Symposium on Mathematical Morphology and Its Applications to Signal and Image Processing*. Springer, 2011, pp. 272–283.

[9] S. Wang, L. Hu, Y. Wang, X. He, Q. Z. Sheng, M. A. Orgun, L. Cao, F. Ricci, and S. Y. Philip, "Graph learning based recommender systems: A review," in *IJCAI International Joint Conference on Artificial Intelligence*. International Joint Conferences on Artificial Intelligence, 2021, pp. 4644–4652.

[10] X. Wang, X. He, M. Wang, F. Feng, and T.-S. Chua, "Neural graph collaborative filtering," in *Proceedings of the 42nd international ACM SIGIR conference on Research and development in Information Retrieval*, 2019, pp. 165–174.

[11] X. He, K. Deng, X. Wang, Y. Li, Y. Zhang, and M. Wang, "Lightgcn: Simplifying and powering graph convolution network for recommendation," in *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval*, 2020, pp. 639–648.

[12] Q. Wang, Y. Wei, J. Yin, J. Wu, X. Song, and L. Nie, "Dualgnn: Dual graph neural network for multimedia recommendation," *IEEE Transactions on Multimedia*, vol. 25, pp. 1074–1084, 2021.

[13] Y. Hao, J. Ma, P. Zhao, G. Liu, X. Xian, L. Zhao, and V. S. Sheng, "Multi-dimensional graph neural network for sequential recommendation," *Pattern Recognition*, vol. 139, p. 109504, 2023.

[14] 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*, 2009, pp. 452–461.

[15] G. Cornuejols, M. Fisher, and G. L. Nemhauser, "On the uncapacitated location problem," in *Annals of Discrete Mathematics*. Elsevier, 1977, vol. 1, pp. 163–177.

[16] K. Han, Y. Wang, J. Guo, Y. Tang, and E. Wu, "Vision gnn: An image is worth graph of nodes," in *NeurIPS*, 2022.

[17] J. P. O. Batisteli, S. J. F. Guimarães, and Z. K. G. Patrocínio, "Multiscale image graph representation: A novel gnn approach for image classification through scale importance estimation," in *2023 IEEE International Symposium on Multimedia (ISM)*, 2023, pp. 62–68.

[18] P. Anderson, X. He, C. Buehler, D. Teney, M. Johnson, S. Gould, and L. Zhang, " Bottom-Up and Top-Down Attention for Image Captioning and Visual Question Answering ," in *2018 IEEE/CVF Conference on*

Computer Vision and Pattern Recognition (CVPR). Los Alamitos, CA, USA: IEEE Computer Society, Jun. 2018, pp. 6077–6086.

[19] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, “Attention is all you need,” *Advances in neural information processing systems*, vol. 30, 2017.