2025 International Conference on Intelligent Computing
July 26-29, Ningbo, China
https://www.ic-icc.cn/2025/index.php

ME-GCN: Motif-Enhanced Graph Convolutional Network for Recommendation Systems

Jianmin Xu[1] and Ping Lu[1]

School of Cyber Security and Computer, Hebei University, Baoding 071002, Hebei, China.
Xujianmin2025@gmail.com

Abstract. Recommendation systems play a key role in helping users cope with the vast amount of online information, and graph convolutional network (GCN)-based methods have attracted much attention due to their ability to model complex relationships in user-item interaction graphs. However, existing GCN models mainly focus on direct connections between nodes, ignoring the potential value of higher-order structural patterns such as triangles. In this paper, we propose a motif-enhanced graph convolutional network (ME-GCN) to improve recommendation performance by explicitly leveraging the triangle patterns in the user-item interaction graph. Specifically, we design an efficient sparse matrix algorithm to compute the triangle participation of nodes and integrate it into the node embedding via a learnable projection mechanism, which enhances the motif capability of higher-order structural patterns while retaining the simple architecture of GCN. Experiments on three public datasets (MovieLens-1M, Amazon-Books, and Yelp2018) show that ME-GCN significantly outperforms existing benchmark models, especially in sparse data scenarios (up to 7.47%). Ablation experiments further verify the importance of the triangular model, whose contribution far exceeds simple structural features such as the first-order node degree.

Keywords: Graph Convolutional networks; Recommendation systems; Motif-enhanced; Triangular structures; Sparse matrix computation.

1 Introduction

Recommendation systems have become an integral part of our digital experience, helping users navigate the vast amount of online information [14,30]. These systems analyze user preferences and behaviors to suggest items that users might find interesting or relevant. With the exponential growth of online platforms and digital content, the need for accurate and efficient recommendation systems has never been more critical. Graph-based recommendation approaches have gained significant attention in recent years due to their ability to model complex relationships between users and items [13,26]. Among these approaches, Graph Convolutional Networks (GCNs) have emerged as a powerful paradigm by leveraging the structural information inherent in user-item interaction graphs [2,33].

GCNs operate by propagating information through the graph structure, allowing nodes (users and items) to aggregate information from their neighbors, thereby capturing collaborative signals effectively.

Despite their success, traditional GCN-based recommendation models primarily focus on direct connections between nodes, often overlooking higher-order structural patterns that exist in the interaction graphs [28,9]. These higher-order structures, known as network motifs, represent recurring patterns of interconnections that appear significantly more frequently than in random networks [16]. In particular, triangular structures in graphs have been shown to carry important semantic information about the underlying relationships, potentially offering valuable insights for recommendation tasks.

In this paper, we propose a novel approach that enhances GCN-based recommendation models with triangle motif information. Our method efficiently computes the triangle participation of each node in the user-item interaction graph and incorporates this information into the initial node embeddings through a learnable projection mechanism. This approach preserves the elegant simplicity of existing GCN architectures while enriching the node representations with valuable structural information.

Our contributions can be summarized as follows:

1. We propose a motif-enhanced GCN framework for recommendation systems that effectively captures higher-order structural patterns in user-item interaction graphs.

2. We develop an efficient sparse matrix-based algorithm for triangle counting that scales well to large graphs, making our approach practical for real-world recommendation scenarios.

3. We design a learnable projection mechanism that seamlessly integrates triangle motif information into the node embedding process, enhancing the model's ability to capture complex user-item relationships.

2 Related Work

2.1 Graph Convolutional Networks for Recommendation

Graph Convolutional Networks (GCNs) have emerged as powerful tools for recommendation systems due to their ability to effectively model user-item interactions as a bipartite graph. The application of GCNs to recommendation tasks was pioneered by Berg et al. [4], who proposed a graph auto-encoder framework for matrix completion. Building upon this foundation, Wang et al. [24] introduced Neural Graph Collaborative Filtering (NGCF), which explicitly encodes the collaborative signal in the form of high-order connectivity in user-item bipartite graphs. This approach significantly improved recommendation performance compared to traditional matrix factorization methods.

He et al. [11] further refined GCN-based recommendation with LightGCN, which simplifies the design of GCNs by removing feature transformation and nonlinear activation components. LightGCN focuses solely on the essential neighborhood aggregation operation, resulting in a more efficient and effective model. Recent advancements in GCN-based recommendation include disentangled graph networks [25], which separate different types of user preferences, and interest-aware message-passing GCNs [6], which adaptively aggregate information based on user interests. Chen et al. [6] also investigated the impact of sampling strategies on GCN performance and proposed improved neighbor sampling methods.

Despite these advancements, most existing GCN-based recommendation models primarily focus on direct connections and message passing between nodes, without explicitly considering higher-order structural patterns in the graph [15]. Our work addresses this limitation by incorporating triangle motif information into GCN-based recommendation models.

2.2 Network Motifs in Graph Analysis

Network motifs, first introduced by Milo et al. [19], are defined as recurring, statistically significant subgraphs or patterns within a larger network. These motifs are considered the basic building blocks of complex networks and have been shown to carry important functional information in various domains, including biological networks, social networks, and technological networks.

Among various network motifs, triangular structures (three-node complete subgraphs) are particularly important in social and information networks. Triangles reflect the "friend of a friend is a friend" phenomenon in social networks and can indicate strong community structures. Eckmann and Moses [7] proposed using triangles as a measure of local network curvature, demonstrating their importance in understanding network topology.

The integration of motif information with graph neural networks has gained attention in recent years. Ahmed et al. [1] studied the role of different motifs in graph representation learning and proposed a motif-based attention mechanism. Sankar et al. [21] applied motifs to dynamic graph representation learning, capturing structural evolution patterns in temporal graphs. However, these works primarily focus on homogeneous graphs like social networks and citation networks, with limited application to bipartite user-item graphs in recommendation systems.

2.3 Structural Information in Recommendation Systems

Structural information in recommendation systems refers to the patterns and relationships that exist in the user-item interaction graph beyond simple direct connections. Early work in this direction includes ItemRank [8], which applies PageRank-like algorithms to the item-item correlation graph to generate recommendations. Similarly, Zhou et al. [32] proposed a graph-based recommendation algorithm that exploits the bipartite graph structure to address the cold-start problem.

More recent approaches have explored various ways to incorporate structural information into recommendation models. Wu et al. [27] proposed a session-based recommendation method that captures complex transitions between items using graph neural networks. Yu et al. [29] introduced a self-supervised learning framework that leverages the graph structure to generate auxiliary tasks for recommendation. Several studies have specifically investigated the importance of higher-order connectivity patterns in recommendation. Zhao et al. [31] demonstrated that incorporating higher-order connectivity information can significantly improve recommendation accuracy, especially in sparse data scenarios. Wang et al. [22] proposed a global-local neighborhood method that captures both local structures and global connectivity patterns.

Despite these advances, the explicit modeling of network motifs, particularly triangular structures, in recommendation systems remains relatively unexplored. Most existing methods either focus on direct connections or use general higher-order connectivity without specifically targeting meaningful structural patterns like triangles [5,23]. Our work bridges this gap by explicitly incorporating triangle motif information into GCN-based recommendation models, providing a new perspective on leveraging structural information for improved recommendation performance.

3 Preliminaries

3.1 Problem Formulation

In this section, we formally define the recommendation problem addressed in this paper. Let $\mathcal{U} = \{u_1, u_2, ..., u_M\}$ denote the set of M users and $\mathcal{I} = \{i_1, i_2, ..., i_N\}$ denote the set of N items. The user-item interactions are represented as a matrix $\mathbf{Y} \in \mathbb{R}^{M \times N}$, where $y_{ui} = 1$ if user u has interacted with item i (e.g., purchased, clicked, or rated), and $y_{ui} = 0$ otherwise.

The user-item interactions can be naturally modeled as a bipartite graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V} = \mathcal{U} \cup \mathcal{I}$ is the set of nodes (users and items), and $\mathcal{E} \subseteq \mathcal{U} \times \mathcal{I}$ is the set of edges representing observed interactions. The adjacency matrix of this graph, denoted as $\mathbf{A} \in \mathbb{R}^{(M+N) \times (M+N)}$, can be represented as:

$$\mathbf{A} = \begin{pmatrix} \mathbf{0}_{M \times M} & \mathbf{Y} \\ \mathbf{Y}^T & \mathbf{0}_{N \times N} \end{pmatrix} \tag{1}$$

where $\mathbf{0}$ represents a matrix of zeros, indicating that there are no direct connections between users or between items in the bipartite graph.

Given the interaction graph $\mathcal{G}$, the goal of the recommendation system is to predict the likelihood of unobserved interactions between users and items. Specifically, for each user $u \in \mathcal{U}$, we aim to generate a ranked list of items that the user has not interacted with, ordered by the predicted preference scores.

3.2 Basic Concepts of GCN

Graph Convolutional Networks (GCNs) extend the concept of convolution from regular grid data (like images) to irregular graph-structured data. In the context of recommendation systems, GCNs operate on the user-item interaction graph to learn node embeddings that capture collaborative signals.

The core operation in GCNs is neighborhood aggregation, where each node aggregates information from its neighbors to update its representation. A general form of this operation can be expressed as:

$$\mathbf{h}_v^{(l+1)} = \sigma \left(\sum_{u \in \mathcal{N}(v)} \frac{1}{\sqrt{|\mathcal{N}(v)|} \cdot \sqrt{|\mathcal{N}(u)|}} \mathbf{W}^{(l)} \mathbf{h}_u^{(l)} \right) \tag{2}$$

where $\mathbf{h}_v^{(l)}$ is the representation of node v at layer l, $\mathcal{N}(v)$ is the set of neighbors of node v, $\mathbf{W}^{(l)}$ is a learnable weight matrix, and σ is a non-linear activation function.

In LightGCN [11], a simplified version of GCN tailored for recommendation, the feature transformation and non-linear activation are removed, resulting in the following propagation rule:

$$\mathbf{e}_v^{(l+1)} = \sum_{u \in \mathcal{N}(v)} \frac{1}{\sqrt{|\mathcal{N}(v)|} \cdot \sqrt{|\mathcal{N}(u)|}} \mathbf{e}_u^{(l)} \tag{3}$$

where $\mathbf{e}_v^{(l)}$ is the embedding of node v at layer l. The final embedding of a node is obtained by combining embeddings from all layers:

$$\mathbf{e}_v = \sum_{l=0}^{L} \alpha_l \mathbf{e}_v^{(l)} \tag{4}$$

where α_l is the weight for layer l (often set to $\frac{1}{L+1}$ for simplicity), and L is the total number of layers.

The prediction score for a user-item pair (u, i) is then computed as the inner product of their final embeddings:

$$\hat{y}_{ui} = \mathbf{e}_u^T \mathbf{e}_i \tag{5}$$

3.3 Triangle Motifs in Graphs

Network motifs are recurring, statistically significant subgraphs or patterns within a larger network. Among various motifs, triangles (three-node complete subgraphs) are particularly important in social and information networks. A triangle in a graph consists of three nodes that are all connected to each other, forming a cycle of length three.

In the context of a user-item bipartite graph, triangles cannot exist directly since there are no connections between nodes of the same type. However, we can consider triangles in the projected graphs or in the graph after adding virtual connections. For instance, if we add virtual connections between users who have interacted with the same item, triangles in this augmented graph can indicate users with similar preferences.

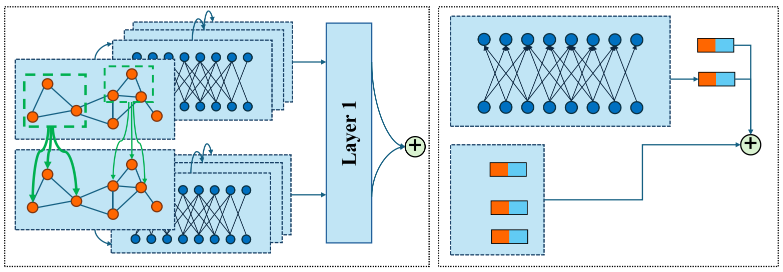

Fig. 1. Overview of the proposed Motif-Enhanced Graph Convolutional Network (ME-GCN) framework. The framework first computes triangle participation for each node, projects these features into the embedding space, and integrates them with the initial node embeddings before performing GCN-based message passing.

Formally, for a graph with adjacency matrix $\mathbf{A}$, the number of triangles that a node v participates in can be computed using the diagonal elements of the cube of the adjacency matrix:

$$\text{triangles}(v) = \frac{(\mathbf{A}^3)_{vv}}{2} \tag{6}$$

The division by 2 is necessary because each triangle is counted twice in the matrix multiplication (once in each direction). This formula provides an efficient way to compute the triangle participation of each node in the graph, which serves as a measure of the node's involvement in tightly-knit communities. In our approach, we compute the triangle participation for each node in the user-item interaction graph and use this information to enhance the node's initial embedding. This allows the model to capture higher-order structural patterns that are not explicitly modeled by traditional GCN-based recommendation methods.

4 Methodology

4.1 Overview of the Proposed Framework

In this section, we present our Motif-Enhanced Graph Convolutional Network (ME-GCN) framework for recommendation systems. Figure 1 illustrates the overall architecture of our proposed framework. We first compute the triangle participation count for each node in the user-item interaction graph using an efficient sparse matrix-based algorithm. This step quantifies the involvement of each node in triangular structures, providing valuable information about local community patterns. The triangle counts are then projected into the embedding space through a learnable projection layer. This transforms the scalar triangle counts into high-dimensional feature vectors that can be effectively combined with node embeddings. The projected motif features are integrated with the initial node embeddings, enriching them with structural information before the message passing process begins. The enhanced node embeddings are then processed through multiple GCN layers, where nodes aggregate information from their neighbors to capture collaborative signals. Finally, the learned user and item embeddings are used to compute prediction scores, and the model is optimized using a ranking-based loss function.

Our framework is designed to be flexible and can be integrated with various GCN-based recommendation models. In this paper, we demonstrate its effectiveness by enhancing two representative models: LightGCN [11] and IMP-GCN, resulting in ME-LightGCN and ME-IMP-GCN, respectively.

The key advantage of our approach is that it captures valuable structural information that is not explicitly modeled by traditional GCN-based methods, while maintaining computational efficiency. The triangle counting is performed only once as a preprocessing step, and the motif feature projection adds minimal computational overhead to the model.

4.2 Triangle Counting Algorithm

Efficient triangle counting is crucial for the practical application of our framework, especially when dealing with large-scale recommendation datasets. While various triangle counting algorithms exist in the literature, many of them are designed for dense graphs and do not scale well to the sparse, large-scale graphs commonly encountered in recommendation systems.

We propose an efficient sparse matrix-based algorithm for triangle counting that leverages the power of modern matrix libraries and hardware acceleration. The algorithm is based on the observation that for a graph with adjacency matrix $\mathbf{A}$, the number of triangles that a node v participates in can be computed using the diagonal elements of $\mathbf{A}^3$:

$$\text{triangles}(v) = \frac{(\mathbf{A}^3)_{vv}}{2} \tag{7}$$

A naive implementation of this formula would involve computing the full cube of the adjacency matrix, which is computationally expensive for large graphs. Instead, we leverage sparse matrix operations to efficiently compute only the diagonal elements of $\mathbf{A}^3$.

Algorithm 1 outlines our approach for efficient triangle counting in sparse graphs.

The key to the efficiency of our algorithm lies in the use of sparse matrix operations. Modern deep learning frameworks like PyTorch and TensorFlow provide highly optimized implementations of sparse matrix multiplication that can leverage GPU acceleration. By using these operations, we can compute triangle counts for large graphs with millions of nodes and edges in a matter of seconds.

For the user-item bipartite graph, we need to make a slight modification to the standard triangle counting approach. Since triangles cannot exist in a

Algorithm 1 Efficient Triangle Counting in Sparse Graphs

```
Require: Sparse adjacency matrix $\mathbf{A} \in \mathbb{R}^{n \times n}$
Ensure: Triangle count vector $\mathbf{t} \in \mathbb{R}^n$
1: Convert $\mathbf{A}$ to COO format and normalize
2: Compute $\mathbf{A}^2 = \mathbf{A} \times \mathbf{A}$ using sparse matrix multiplication
3: Compute $\mathbf{A}^3 = \mathbf{A}^2 \times \mathbf{A}$ using sparse matrix multiplication
4: Extract diagonal elements of $\mathbf{A}^3$ to get vector $\mathbf{d}$
5: Compute triangle counts $\mathbf{t} = \mathbf{d}/2$
6: return $\mathbf{t}$
```

bipartite graph (as there are no connections between nodes of the same type), we first transform the bipartite graph into a unified graph by adding virtual connections.

Specifically, we construct a unified adjacency matrix $\mathbf{A}$ as follows:

$$\mathbf{A} = \begin{pmatrix} \mathbf{0}_{M \times M} & \mathbf{Y} \\ \mathbf{Y}^T & \mathbf{0}_{N \times N} \end{pmatrix} \tag{8}$$

where $\mathbf{Y} \in \mathbb{R}^{M \times N}$ is the user-item interaction matrix, M is the number of users, and N is the number of items.

When we compute $\mathbf{A}^3$, the diagonal elements correspond to the number of paths of length 3 that start and end at the same node. In a bipartite graph, these paths must traverse nodes of different types, effectively capturing higher-order connectivity patterns between users and items.

For users, the triangle count represents the number of paths $u \to i \to u' \to i' \to u$, where u and u' are users, and i and i' are items. These paths indicate situations where users have interacted with common items, suggesting potential similarity in preferences. Similarly, for items, the triangle count captures paths that indicate potential relationships between items based on user interaction patterns.

The time complexity of our triangle counting algorithm is dominated by the sparse matrix multiplication operations. For a graph with n nodes and m edges, the complexity is approximately $O(m^{1.5})$, which is significantly better than the $O(n^3)$ complexity of dense matrix multiplication. This makes our approach scalable to large recommendation datasets with millions of users and items.

4.3 Motif Feature Projection

After computing the triangle participation counts for each node, we need to transform this scalar information into a form that can be effectively integrated with the node embeddings used in GCN models. Simply using the raw triangle counts as features may not be optimal due to several reasons: (1) the scale of triangle counts can vary significantly across different nodes and datasets, (2) the relationship between triangle counts and node preferences may be complex and non-linear, and (3) we need to map the one-dimensional triangle counts to the high-dimensional embedding space.

To address these challenges, we propose a learnable projection mechanism that transforms the triangle counts into the embedding space. Specifically, for each node v with triangle count t_v, we compute a motif feature vector $\mathbf{m}_v \in \mathbb{R}^d$ as follows:

$$\mathbf{m}_v = f_\theta(t_v) \tag{9}$$

where f_θ is a projection function parameterized by θ, and d is the embedding dimension.

We implement f_θ as a simple yet effective linear transformation followed by a non-linear activation:

$$f_\theta(t_v) = \sigma(\mathbf{W}_m \cdot t_v + \mathbf{b}_m) \tag{10}$$

where $\mathbf{W}_m \in \mathbb{R}^{d\times 1}$ and $\mathbf{b}_m \in \mathbb{R}^d$ are learnable parameters, and σ is a non-linear activation function such as ReLU or Leaky ReLU.

To handle potential numerical issues with extremely large triangle counts, we apply a logarithmic transformation to the raw counts before projection:

$$\tilde{t}_v = \log(1 + t_v) \tag{11}$$

This transformation helps compress the range of triangle counts and makes the learning process more stable, especially for graphs with highly skewed triangle count distributions.

4.4 Integration with GCN Models

With the projected motif features in hand, we now describe how to integrate them with GCN-based recommendation models. Our approach is designed to be model-agnostic and can be applied to various GCN architectures. In this paper, we demonstrate its effectiveness with two representative models: LightGCN and IMP-GCN.

Integration with LightGCN LightGCN [11] is a state-of-the-art GCN-based recommendation model that simplifies the design of GCNs by removing feature transformation and non-linear activation components. The core of LightGCN is the light graph convolution operation:

$$\mathbf{e}^{(k+1)} = \tilde{\mathbf{A}}\mathbf{e}^{(k)} \tag{12}$$

where $\mathbf{e}^{(k)}$ is the embedding matrix at layer k, and $\tilde{\mathbf{A}}$ is the normalized adjacency matrix with self-loops.

To enhance LightGCN with motif information, we integrate the projected motif features with the initial embeddings before the message passing process begins:

$$\mathbf{e}_v^{(0)} = \mathbf{e}_v^{(0)} + \alpha \cdot \mathbf{m}_v \tag{13}$$

where $\mathbf{e}_v^{(0)}$ is the initial embedding of node v, $\mathbf{m}_v$ is the projected motif feature, and α is a hyperparameter that controls the influence of motif information.

The enhanced initial embeddings are then processed through the standard LightGCN layers:

$$\mathbf{e}^{(k+1)} = \tilde{\mathbf{A}}\mathbf{e}^{(k)}, \quad k = 0, 1, ..., K-1 \tag{14}$$

The final embeddings are obtained by combining embeddings from all layers:

$$\mathbf{e}_v = \sum_{k=0}^{K} \alpha_k \mathbf{e}_v^{(k)} \tag{15}$$

where α_k is the weight for layer k (often set to $\frac{1}{K+1}$ for simplicity).

Integration with IMP-GCN IMP-GCN (Interest-aware Message Passing GCN) extends LightGCN by introducing a group-specific message passing mechanism. It first assigns users to different interest groups and then performs message passing within each group separately. To enhance IMP-GCN with motif information, we integrate the projected motif features with the initial embeddings before the group assignment and message passing:

$$\mathbf{e}_v^{(0)} = \mathbf{e}_v^{(0)} + \alpha \cdot \mathbf{m}_v \tag{16}$$

The enhanced initial embeddings are then used for group assignment and subsequent message-passing operations, following the standard IMP-GCN procedure.

This integration approach is simple yet effective, as it enhances the initial node representations with structural information while preserving the original model architecture. The motif information influences both the initial node representations and the subsequent message-passing process, allowing the model to better capture complex user-item relationships.

4.5 Model Training

We train our motif-enhanced GCN models using the Bayesian Personalized Ranking (BPR) loss [20], which is widely used for implicit feedback recommendation. The BPR loss aims to maximize the margin between the predicted scores of observed interactions and unobserved ones:

$$\mathcal{L}_{BPR} = \sum_{(u,i,j)\in\mathcal{D}} -\ln \sigma(\hat{y}_{ui} - \hat{y}_{uj}) + \lambda \|\Theta\|_2^2 \tag{17}$$

where (u, i, j) denotes that user u has interacted with item i but not with item j, $\hat{y}_{ui}$ and $\hat{y}_{uj}$ are the predicted scores, σ is the sigmoid function, Θ represents the model parameters, and λ is the regularization coefficient.

The predicted score for a user-item pair is computed as the inner product of their final embeddings:

Table 1. Statistics of the datasets used in our experiments.

Dataset	Users	Items	Interactions	Density	Avg. Degree
MovieLens-1M	6,040	3,706	1,000,209	4.47%	165.6
Amazon-Books	52,643	91,599	2,984,108	0.06%	56.7
Yelp2018	31,668	38,048	1,561,406	0.13%	49.3

$$\hat{y}_{ui} = \mathbf{e}_u^T \mathbf{e}_i \tag{18}$$

where $\mathbf{e}_u$ and $\mathbf{e}_i$ are the final embeddings of user u and item i, respectively.

5 Experiments

5.1 Experimental Setup

Datasets We conduct experiments on three widely-used public recommendation datasets with varying scales and domains: **MovieLens-1M** [10]: A movie rating dataset containing 1 million ratings from 6,040 users on 3,706 movies. Following common practice, we convert the explicit ratings to implicit feedback by treating ratings of 4 or higher as positive interactions. **Amazon-Books** [18]: A subset of the Amazon product review dataset focusing on books. After filtering out users and items with fewer than 5 interactions, the dataset contains 52,643 users, 91,599 books, and 2,984,108 interactions. **Yelp2018** [3]: A business review dataset from the Yelp Challenge 2018. After preprocessing to remove users and businesses with fewer than 10 interactions, it contains 31,668 users, 38,048 businesses, and 1,561,406 reviews.

Table 1 summarizes the statistics of the three datasets after preprocessing.

For each dataset, we randomly split the user-item interactions into training, validation, and test sets with a ratio of 8:1:1. The validation set is used for hyperparameter tuning and early stopping, while the test set is used for the final performance evaluation.

Baselines We compare our proposed methods with **BPR** [20], **NeuMF** [12], **NGCF** [24], **LightGCN** [11], **DGCF** [25], **UltraGCN** [17] and **IMP-GCN**. In addition to these baselines, we implement two variants of our approach, **ME-LightGCN**, our motif-enhanced version of LightGCN and **ME-IMP-GCN**, Our motif-enhanced version of IMP-GCN.

Evaluation Metrics We adopt two widely used metrics for evaluating top-N recommendation performance:

Recall@K is the proportion of relevant items that are successfully retrieved in the top-K recommendations:

$$\text{Recall@K} = \frac{|\text{relevant items} \cap \text{recommended items at K}|}{|\text{relevant items}|} \tag{19}$$

NDCG@K is normalized Discounted Cumulative Gain at K, which takes into account the position of relevant items in the recommendation list:

$$\text{NDCG@K} = \frac{\text{DCG@K}}{\text{IDCG@K}} \tag{20}$$

where DCG@K is the discounted cumulative gain at K, and IDCG@K is the ideal DCG@K. For both metrics, higher values indicate better performance. We report results for K = 10 and K = 20.

Following common practice in recommendation research, we adopt the all-ranking protocol for evaluation. For each user in the test set, we rank all items that the user has not interacted with in the training set and compute the metrics based on the ground truth interactions in the test set.

Implementation Details All experiments are implemented in PyTorch on NVIDIA V100 GPUs. Hyperparameters are validation-tuned with unified cfg: 64-dim embeddings, Adam (lr=0.001), batch size=2048, L2=1e-4, and 10-epoch early stopping. GCN-based models use 3 layers (LightGCN/ME-LightGCN) or 6 layers (IMP-GCN/ME-IMP-GCN), with 4 interest groups for IMP variants. Motif-enhanced models employ $\alpha \in 0.1, 0.3, 0.5, 0.7, 1.0$ and LeakyReLU (slope=0.2) for feature projection. Training continues until convergence (200 epochs max), with the best validation performers selected for testing.

5.2 Performance Comparison

Table 2 presents the overall performance comparison of our proposed methods and baseline methods on the three datasets. The best results are highlighted in bold, and the second-best results are underlined.

From the results, we can make observations that GCN-based methods (NGCF, LightGCN, DGCF, UltraGCN, IMP-GCN) generally outperform traditional methods (BPR, NeuMF), confirming the effectiveness of graph-based approaches for recommendation. Our motif-enhanced models (ME-LightGCN and ME-IMP-GCN) consistently outperform their base models and all baseline methods across all datasets and metrics. This confirms the effectiveness of incorporating triangle motif information into GCN-based recommendation models. ME-IMP-GCN achieves the best performance among all methods, with significant improvements over the best baseline (IMP-GCN): 3.64% and 4.43% on MovieLens-1M, 6.78% and 7.47% on Amazon-Books, and 5.89% and 5.85% on Yelp2018 for Recall@20 and NDCG@20, respectively.

The figure 2 shows the changes in the Recall@20 and NDCG@20 metrics of ME-IMP-GCN, ME-LightGCN, UltraGCN and LightGCN at different layers. The results show that the performance of ME-IMP-GCN and ME-LightGCN steadily improves with the increase of the number of layers. UltraGCN shows

Table 2. Overall performance comparison. The best results are in bold, and the second-best results are underlined. Improvements over the best baseline are shown in the last row.

Method	MovieLens-1M		Amazon-Books		Yelp2018	
	Recall@20	NDCG@20	Recall@20	NDCG@20	Recall@20	NDCG@20
BPR	0.1982	0.1213	0.0389	0.0298	0.0543	0.0442
NeuMF	0.2104	0.1345	0.0412	0.0324	0.0561	0.0459
NGCF	0.2170	0.1404	0.0478	0.0368	0.0579	0.0477
LightGCN	0.2341	0.1561	0.0517	0.0402	0.0649	0.0530
DGCF	0.2356	0.1573	0.0525	0.0410	0.0657	0.0538
UltraGCN	0.2372	0.1589	0.0529	0.0413	0.0660	0.0542
IMP-GCN	0.2389	0.1602	0.0531	0.0415	0.0662	0.0547
ME-LightGCN	0.2425	0.1634	0.0548	0.0429	0.0683	0.0561
ME-IMP-GCN	**0.2476**	**0.1673**	**0.0567**	**0.0446**	**0.0701**	**0.0579**
Improvement	3.64%	4.43%	6.78%	7.47%	5.89%	5.85%

Table 3. Ablation study results on the MovieLens-1M dataset.

Method	Recall@10	NDCG@10	Recall@20	NDCG@20
LightGCN	0.1723	0.1382	0.2341	0.1561
LightGCN + Degree	0.1742	0.1401	0.2365	0.1579
LightGCN + Triangle	0.1798	0.1452	0.2425	0.1634
IMP-GCN	0.1782	0.1423	0.2389	0.1602
IMP-GCN + Degree	0.1805	0.1446	0.2412	0.1625
IMP-GCN + Triangle	0.1863	0.1498	0.2476	0.1673

slight overfitting at deeper layers, while LightGCN performs the weakest and has a limited improvement. This shows that motifs can improve the results of the model at deeper levels.

In addition, the improvements are more substantial on the Amazon-Books and Yelp2018 datasets compared to MovieLens-1M. This may be because these datasets are sparser (as shown in Table 1), and the structural information provided by triangle motifs is particularly valuable in sparse scenarios where direct user-item interactions are limited.

5.3 Ablation Studies

To understand the contribution of triangle motif information to the overall performance , we conduct ablation studies by comparing different variants of our approach. Table 3 presents the results on the MovieLens-1M dataset.

In this ablation study, we compare the following variants:

- **Base models**: LightGCN and IMP-GCN without any additional structural information.

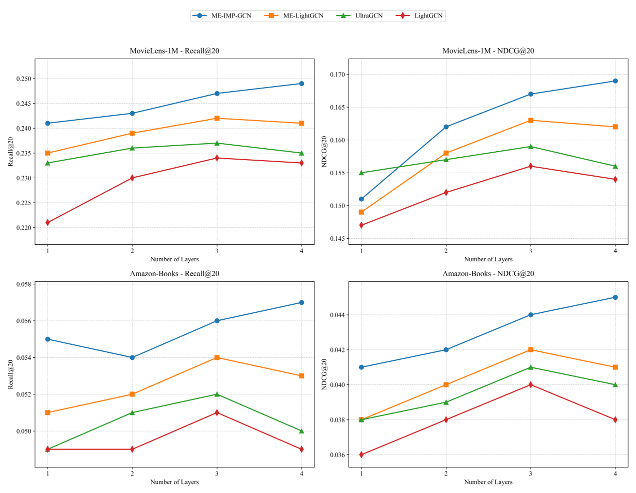

Fig. 2. Comparison of the ME-GCN variant with the baseline model on the MovieLens-1M and Amazon-Books datasets for different numbers of layers

- + **Degree**: Enhancing the base models with node degree information instead of triangle motifs. Node degree is a simple first-order structural feature that indicates the number of direct connections a node has.
- + **Triangle**: Our full models that enhance the base models with triangle motif information.

From the results, we can observe that both degree and triangle information bring improvements over the base models, confirming the value of incorporating structural information into GCN-based recommendation models. Triangle motif information leads to more significant improvements compared to degree information. For LightGCN, triangle motifs bring 3.59% and 4.68% improvements in Recall@20 and NDCG@20, while degree information only brings 1.03% and 1.15% improvements. Similar patterns are observed for IMP-GCN. Furthermore, the superior performance of triangle motifs over node degree suggests that higher-order structural patterns capture more valuable information about user preferences and item relationships compared to simple first-order connectivity.

These findings demonstrate that triangle motif information makes a significant contribution to the overall performance, and this contribution is more substantial than that of simpler structural features like node degree.

5.4 Efficiency Analysis

Toevaluate the computational efficiency of our approach, we compare the training time and inference time of different methods on the MovieLens-1M dataset. Table 4 presents the results.

Table 4. Efficiency comparison on the MovieLens-1M dataset.

Method	Training Time (s/epoch)	Inference Time (ms/user)	#Parameters (M)
BPR	0.42	1.23	0.62
NeuMF	0.78	1.87	1.05
NGCF	1.35	1.56	1.27
LightGCN	0.86	1.42	0.62
DGCF	2.43	1.89	1.35
UltraGCN	0.92	1.45	0.63
IMP-GCN	1.24	1.53	0.65
ME-LightGCN	0.89	1.44	0.63
ME-IMP-GCN	1.28	1.55	0.66

From the results, we can make observations that our motif-enhanced models (ME-LightGCN and ME-IMP-GCN) have only slightly higher training and inference times compared to their base models (LightGCN and IMP-GCN), with increases of about 3-4%. This confirms that incorporating triangle motif information adds minimal computational overhead.

In addition, the number of parameters in our motif-enhanced models is also only slightly higher than the base models, with increases of about 1-2%. This is because the motif projection layer adds only a small number of parameters (embedding dimension $\times$ 1 for the weight matrix and embedding dimension for the bias vector). The triangle counting step, which is performed only once as a preprocessing step, takes about 0.5 seconds for the MovieLens-1M dataset. This is negligible compared to the total training time, which is typically several hours.

Therefore, compared to more complex models like DGCF, our motif-enhanced models achieve better performance with significantly lower computational costs. These findings answer demonstrate that our approach is computationally efficient, with minimal overhead compared to the base models. This makes it practical for real-world recommendation scenarios where computational resources may be limited.

6 Conclusion

In this paper, we proposed a novel Motif-Enhanced Graph Convolutional Network (ME-GCN) framework for recommendation systems. Our approach leverages triangle motif information to capture higher-order structural patterns in user-item interaction graphs, enhancing the node representations used in GCN-based recommendation models. Extensive experiments on three public datasets demonstrated that our approach consistently outperforms state-of-the-art recommendation methods, with significant improvements in both accuracy and ranking metrics.

References

1. Ahmed, N.K., Rossi, R.A., Lee, J.B., Willke, T.L., Zhou, R., Kong, X., Eldardiry, H.: Role-based graph embeddings. IEEE Transactions on Knowledge and Data Engineering **34**(5), 2401–2415 (2020)
2. Anand, V., Maurya, A.K.: A survey on recommender systems using graph neural network. ACM Transactions on Information Systems **43**(1), 1–49 (2025)
3. Asghar, N.: Yelp dataset challenge. https://www.yelp.com/dataset/challenge (2016)
4. Berg, R.v.d., Kipf, T.N., Welling, M.: Graph convolutional matrix completion. arXiv preprint arXiv:1706.02263 (2017)
5. Boccaletti, S., De Lellis, P., Del Genio, C., Alfaro-Bittner, K., Criado, R., Jalan, S., Romance, M.: The structure and dynamics of networks with higher order interactions. Physics Reports **1018**, 1–64 (2023)
6. Chen, Z., Zhang, S., Hu, Z., Luo, Z., Xu, C.Z.: Revisiting graph based collaborative filtering: A linear residual graph convolutional network approach. arXiv preprint arXiv:2011.01280 (2020)
7. Eckmann, J.P., Moses, E.: Curvature of co-links uncovers hidden thematic layers in the world wide web. Proceedings of the National Academy of Sciences **99**(9), 5825–5829 (2002)
8. Gori, M., Pucci, A.: Itemrank: A random-walk based scoring algorithm for recommender engines. In: IJCAI. vol. 7, pp. 2760–2765 (2007)
9. Han, J., Tang, Y., Tao, Q., Xia, Y., Zhang, L.: Dual homogeneity hypergraph motifs with cross-view contrastive learning for multiple social recommendations. ACM Transactions on Knowledge Discovery from Data **18**(6), 1–24 (2024)
10. Harper, F.M., Konstan, J.A.: The movielens datasets: History and context. ACM Transactions on Interactive Intelligent Systems (TiiS) **5**(4), 1–19 (2015)
11. He, X., Deng, K., Wang, X., Li, Y., Zhang, Y., Wang, M.: Lightgcn: Simplifying and powering graph convolution network for recommendation. arXiv preprint arXiv:2002.02126 (2020)
12. He, X., Liao, L., Zhang, H., Nie, L., Hu, X., Chua, T.S.: Neural collaborative filtering. In: Proceedings of the 26th international conference on world wide web. pp. 173–182 (2017)
13. Huang, Y., Zhu, A., Zeng, C., Hu, C., Lai, X., Feng, W., Chen, F.: Rfis-fpi: Reversible face image steganography neural network for face privacy interactions. In: 2024 IEEE 18th International Conference on Automatic Face and Gesture Recognition (FG). pp. 1–10. IEEE (2024)
14. Lu, J., Wu, D., Mao, M., Wang, W., Zhang, G.: Recommender system application developments: a survey. Decision support systems **74**, 12–32 (2015)
15. Luo, T.: Improving representation learning on graph-structural data for classification, generation, and recommendation (2024)
16. Majhi, S., Perc, M., Ghosh, D.: Dynamics on higher-order networks: A review. Journal of the Royal Society Interface **19**(188), 20220043 (2022)
17. Mao, K., Zhu, J., Xiao, X., Lu, B., Wang, Z., He, X.: Ultragcn: ultra simplification of graph convolutional networks for recommendation. In: Proceedings of the 30th ACM international conference on information & knowledge management. pp. 1253–1262 (2021)
18. McAuley, J., Targett, C., Shi, Q., Van Den Hengel, A.: Image-based recommendations on styles and substitutes. In: Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval. pp. 43–52 (2015)

19. Milo, R., Shen-Orr, S., Itzkovitz, S., Kashtan, N., Chklovskii, D., Alon, U.: Network motifs: simple building blocks of complex networks. Science **298**(5594), 824–827 (2002)
20. Rendle, S., Freudenthaler, C., Gantner, Z., Schmidt-Thieme, L.: Bpr: Bayesian personalized ranking from implicit feedback. Proceedings of the twenty-fifth conference on uncertainty in artificial intelligence pp. 452–461 (2009)
21. Sankar, A., Wu, Y., Gou, L., Zhang, W., Yang, H.: Dysat: Deep neural representation learning on dynamic graphs via self-attention networks. In: Proceedings of the 13th international conference on web search and data mining. pp. 519–527 (2020)
22. Wang, D., Hu, W., Cao, E., Sun, W.: Global-to-local neural networks for document-level relation extraction. arXiv preprint arXiv:2009.10359 (2020)
23. Wang, Q., Cao, H., Li, X., Chang, K.C.C., Cheng, R.: From motif to path: Connectivity and homophily. In: 2024 IEEE 40th International Conference on Data Engineering (ICDE). pp. 2751–2764. IEEE (2024)
24. Wang, X., He, X., Wang, M., Feng, F., Chua, T.S.: Neural graph collaborative filtering. In: Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval. pp. 165–174 (2019)
25. Wang, X., Jin, H., Zhang, A., He, X., Xu, T., Chua, T.S.: Disentangled graph collaborative filtering. In: Proceedings of the 43rd international ACM SIGIR conference on research and development in information retrieval. pp. 1001–1010 (2020)
26. Wang, Z., Huang, Y.: Gdo: Gradual domain osmosis. arXiv preprint arXiv:2501.19159 (2025)
27. Wu, S., Tang, Y., Zhu, Y., Wang, L., Xie, X., Tan, T.: Session-based recommendation with graph neural networks. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 33, pp. 346–353 (2019)
28. Yin, Y., Zhu, X., Wang, W., Zhang, Y., Wang, P., Fan, Y., Guo, J.: Hec-gcn: Hypergraph enhanced cascading graph convolution network for multi-behavior recommendation. arXiv preprint arXiv:2412.14476 (2024)
29. Yu, F., Liu, C., Yuan, K., Liu, Y., Wu, Y.: Self-supervised learning for large-scale item recommendations. arXiv preprint arXiv:2105.07874 (2021)
30. Zhang, S., Yao, L., Sun, A., Tay, Y.: Deep learning based recommender system: A survey and new perspectives. ACM computing surveys (CSUR) **52**(1), 1–38 (2019)
31. Zhao, M., Zhang, J., Zhang, C., Zhang, W.: Leveraging heterogeneous auxiliary tasks to assist crowd counting. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 12736–12745 (2019)
32. Zhou, T., Jiang, L., Su, R., Zhang, Y.: Solving the apparent diversity-accuracy dilemma of recommender systems. Proceedings of the National Academy of Sciences **107**(10), 4511–4515 (2010)
33. Zhu, H., Kapoor, V., Sharma, P.: Reviewing developments of graph convolutional network techniques for recommendation systems. arXiv preprint arXiv:2311.06323 (2023)