Deep Attention Network Designed With Cluster For Intelligent Recommendation System

Dong Liang and Qiang Han*

Qiongtai Normal University, Haikou, Hainan 570100, China

** Corresponding author. E-mail: qiang_han89@outlook.com, hqhhh1616@163.com*

Received: March 09, 2026; Accepted: April 04, 2026

This paper proposes a deep attention network based on cluster design (ASDCLK), which effectively removes noise and dynamically adjusts the weights of knowledge triplets by introducing a degree sensitive graph structure denoising module and an attention-based knowledge aggregation mechanism, thereby improving recommendation accuracy. The experiment was conducted on public datasets for music recommendation and movie recommendation scenarios, and the results showed that the ASDCLK model outperformed current advanced recommendation methods in CTR prediction and top-K recommendation tasks. Especially on the Last.FM and MovieLens-1M datasets, ASDCLK performs well in AUC and MovieLens-1M, respectively Recall@20 There is a significant improvement in indicators compared to the baseline model. In addition, by adjusting the sampling probability of graph structure denoising, this study found that a moderate sampling probability can effectively remove noise while preserving key interactive information, thereby achieving optimal recommendation performance. The model is evaluated on the MovieLens-1M and Last.FM datasets, comprising thousands of users and interactions, using standard evaluation metrics such as AUC, F1-score, and Recall@K to comprehensively assess recommendation performance.

Keywords: Intelligent Recommendation System; Graph Neural Network (GNN); Attention Mechanism; Knowledge Graph; Top-K Recommendation

http://dx.doi.org/10.6180/jase.202609_32.072

1. Introduction

With the rapid development of information technology, the exponential growth of data has intensified the problem of information overload for users. In recent years, deep learning techniques, particularly graph neural networks (GNNs), have been widely adopted in recommendation systems due to their strong capability in modeling graph-structured data and improving representation learning [1].

GNN-based recommendation approaches have significantly enhanced performance; however, they suffer from inherent limitations. During message passing, high-degree nodes tend to lose their distinctive features due to excessive aggregation, leading to representation degradation [2, 3]. Additionally, user interaction data often contains noise, which negatively impacts recommendation accuracy. Although edge pruning techniques have been proposed to reduce noise, existing studies largely overlook the broader impact of graph structure noise on model effectiveness [4, 5]. Furthermore, current methods face challenges in effectively handling complex interaction graphs, often relying on fixed aggregation strategies that fail to capture varying importance among knowledge triplets [6, 7]. Existing contrastive learning approaches are also limited to single-view settings, restricting their ability to learn diverse relational features [8, 9]. Compared with recent knowledge-aware recommendation frameworks such as KGAT, KGIN, and other GNN-based approaches, which

primarily rely on uniform neighbor aggregation or static attention mechanisms, the proposed cluster-based attention mechanism introduces structured clustering prior to attention computation. Recent advancements in graph neural network-based recommendation systems have significantly improved representation learning by incorporating attention mechanisms, contrastive learning strategies, and knowledge graph integration. Despite these improvements, challenges such as noise in interaction graphs and ineffective knowledge aggregation still persist. Compared with existing GNN-based recommendation models, ASDCLK introduces a degree-sensitive graph denoising mechanism to reduce interaction noise. Additionally, it integrates clustering with attention-based aggregation to capture both structural and semantic relationships. This article proposes a Cluster designed Deep Attention Network (CDAN) based on cluster design, which effectively solves the above problems by introducing graph structure denoising and a knowledge aggregation mechanism based on learnable attention [10]. Clustering-based recommendation methods rely on static clusters, limiting adaptation to evolving user preferences and reducing personalization. Many separate clustering from representation learning and are sensitive to noise and sparsity, causing instability. To address this, an adaptive attention-integrated framework is proposed. With top-K set to 10, clusters are formed using embedding similarity and iteratively refined during training to improve representations. The cluster size is selected based on empirical evaluation using the silhouette score, where the optimal number of clusters is determined by maximizing intra-cluster similarity and minimizing inter-cluster distance. The convergence of the clustering process is achieved when the centroid updates fall below a predefined threshold or when the maximum number of iterations is reached. Additionally, computational overhead is analyzed in terms of training time complexity, which is primarily influenced by the number of samples, feature dimensionality, and iterations required for convergence. The clustering information is integrated into attention-based aggregation by grouping similar users and items. This guides the model to focus on relevant neighbors within clusters, assigning higher weights to similar interactions and improving embedding quality. A clustering strategy is integrated to capture structural similarities, where items are initially grouped based on cosine similarity of embeddings. During training, clusters are dynamically refined through iterative updates from GNN-based embedding propagation, adjusting similarity relationships and cluster boundaries. This article applies a degree-sensitive edge pruning method to denoise the user–item interaction graph, removing noisy edges while preserving important information by adapting pruning probabilities based on node degree, improving robustness [11]. It also uses a learnable attention scoring function to weight knowledge triplets dynamically, capturing user preferences more effectively than fixed aggregation methods [12, 13]. The existing Graph Neural Network-based recommender models such as KGAT and KGIN, which primarily rely on fixed message passing and static aggregation strategies, the proposed ASDCLK framework introduces a degree-sensitive graph denoising mechanism combined with cluster-aware attention learning. This allows the model to dynamically filter noisy interactions and adaptively reweight knowledge triplets during representation learning. The computational complexity of ASDCLK is estimated through graph neural propagation, scaling with the number of edges, and attention operations, scaling with sequence length and embedding dimension.

2. Methods and materials

2.1. User project interaction diagram for structural denoising

Graph neural networks enhance recommendation performance by improving user and item representation learning [14, 15]. However, high-degree nodes tend to lose their original features during message aggregation, and user interaction data often contains noise.

To address this, some studies propose edge pruning methods using multimodal features to sparsify the graph [16]. Several existing graph denoising methods in recommendation-oriented GNNs include edge dropout, uniform edge sampling, similarity-based pruning, and learnable edge weighting. However, these methods either randomly remove useful interactions or rely on static similarity thresholds, failing to consider node degree imbalance and noise accumulation in high-degree nodes.

Construct a symmetric adjacency matrix $V \in R^{|v|x|V|}, |V|$ based on the user project interaction matrix Y , which represents the number of nodes in the user project interaction graph G_U. The construction of adjacency matrix is shown in Eq. (1):

$$V = \begin{pmatrix} 0 & Y \\ Y^{\top} & 0 \end{pmatrix} \tag{1}$$

In the symmetric matrix V , if user u interacts with item i , then $V_{ui} = 1$; otherwise, $V_{ui} = 0$. When performing graph cropping, if the probability of cropping edges is set to ρ, the number of edges that should be cropped in the user project interaction graph is, $\lfloor \rho \mid \varepsilon \rfloor \rfloor$ where $[- \mid$ J is the base function, $|\varepsilon|$ is the number of edges in the graph, and the number of edges that should be retained is

$n = \lceil (1-\rho)|\varepsilon| \rceil$. For any edge $e_k (0 \leq k \nsubseteq \varepsilon\ |)$ in the user project interaction diagram that connects two nodes i and j, the probability of being sampled is shown in Eq. (2):

$$p_k = \frac{1}{\sqrt{\omega_i}\sqrt{\omega_j}} \tag{2}$$

Among them, ω_i, ω_j are the degrees of points i and j in the figure, respectively.

After obtaining the sampling probability of each edge, sample the edges from the polymorphic distribution with index n and parameter vector $p = \left\langle p_0, p_1, \cdots, p_{z|} \right\rangle$. During the sampling process, nodes with higher degrees have a lower probability of being sampled. Subsequently, based on Eq. (1) and the sampled edges, a denoised symmetric adjacency matrix V_ρ is obtained. Finally, the adjacency matrix is normalized according to Eq. (3) to obtain: V_p

$$V_\rho = D^{-1/2} V_\rho D^{-1/2} \tag{3}$$

Among them, $D \in R^{|v|x|v|}$ is the angle matrix. The proposed degree-sensitive graph structure denoising module employs neighborhood selection which is done according to the user-item interaction graph with only the directly connected nodes being viewed as neighbors. The sampling probability that directs the edge pruning process is based on the node degree, and high degree nodes have lower retention probabilities to minimize noise. The pruning level is set at a moderate level, which is picked on an empirical basis to prevent excessive removal of noise and loss of information.

The proposed degree-sensitive graph structure denoising module employs neighborhood selection which is done according to the user-item interaction graph with only the directly connected nodes being viewed as neighbors. The sampling probability that directs the edge pruning process is based on the node degree, and high degree nodes have lower retention probabilities to minimize noise. The pruning level is set at a moderate level, which is picked on an empirical basis to prevent excessive removal of noise and loss of information.

The graph attention network differentiates the importance of neighboring nodes using attention mechanisms and updates node features accordingly [17]. IoT-based optical fiber sensor network with HMI integration for real-time water level monitoring and feature extraction [18].

This article adopts this aggregation mechanism using GAT on the denoised user–item interaction graph to learn user and item representations, as shown in Eq. (4).

$$e_u^{(I)} = \sum_{i \in N} \alpha e_i^{(I-1)} \tag{4}$$

Among them, $e_u^{(l)}$ and $e_j^{(l-1)}$ are the embeddings of users and projects respectively, is the neighbor set of users, and the aggregation of projects is similar to Eq. (4). The calculation of neighbor weights is shown in Eq. (5):

$$\begin{aligned} \eta_{ij} &= a\left([V\chi_i]\, V_j \right) \\ \alpha &= \frac{\exp\left(\text{LeakyReLUn}_{ij} \right))}{\sum_{k=N} \exp\left(\text{LeakyReLUn}_{ij} \right))} \end{aligned} \tag{5}$$

Among them, e_i, e_j is the embedding of nodes i and j, W is a learnable weight matrix, vector a maps high-dimensional feature vectors to a real number, and softmax function is used to normalize the weight coefficients.

Finally, add up the representations of each layer to obtain the user and project representations of the user project interaction graph:

$$e_u^u = e_u^{(0)} + e_u^{(l)} + \cdots e_u^{\left(L_j\right)} \tag{6}$$

Among them, L_u represents the number of layers in the user project interaction graph aggregation layer. During graph convolution, node embeddings are progressively refined through neighborhood aggregation, capturing higher-order structural patterns and propagating enriched representations across layers. These representations are further enhanced in the attention stage by assigning adaptive weights to important interactions, ensuring effective modeling of both structural dependencies and user preferences. To provide a clearer computational workflow, the clustering component is tightly integrated with the GNN-based feature propagation process. Initially, user and item embeddings are grouped into clusters based on similarity measures. These cluster assignments are then incorporated into the message passing mechanism, where each node prioritizes aggregating information from neighbors within the same cluster.

2.2. Attention based knowledge aggregation

Most recommendation systems use relation-aware aggregation without considering neighbor importance, limiting preference[19, 20]. Existing knowledge graph recommendation models such as TransE, KGAT, and KGIN use triplet weighting and embedding mechanisms, but they show limitations in large-scale environments. Their static or locally normalized weighting strategies fail to capture global structural complexity, and as the graph size increases, noise and weak relations reduce the reliability of learned importance scores. First, low dimensional embeddings of entities and relationships in the knowledge network are learned using the translation based embedding method TransE [21]. The basic idea is to treat the relationship as a translation in the

embedding space. If triplet (h, r, t) holds, the embedding e_t of the tail entity should be close to the embedding of the head entity e_h plus the displacement e_r of the relationship. If $f_d(\cdot)$ represents the 1 L norm to measure the similarity function of embedding vectors, then for a given triplet, it can be expressed as shown in Eq. (7):

$$f_d = \| (e_h + e_r - e_t) \tag{7}$$

Finally, the optimization loss function for the initial embedding of the knowledge graph can be defined as shown in Eq. (8):

$$L_{kg} = \sum_{(h,r\neq t')\in G_k} -1n\sigma\left(f_d\left(e_h + e_r - e_{t'}\right)\right) - f_d\left(e_h + e_r - e_t\right) \tag{8}$$

Negative sample $t'\mathrm{G_k}$ was generated by randomly replacing the tail in triplet of knowledge graph $(h, r, t)\sigma(.).The functionisasigmoidfunction.$

Eq. (9) displays the score calculation for each knowledge triplet. Following the initialization embedding of the knowledge graph, a learnable attention mechanism is utilized to dynamically weight each knowledge triplet.

$$f(h, r, t) = \frac{\left(\mathbf{e}_h W^Q\right) \cdot \left(\left(\mathbf{e}_t W^K\right) \|\mathbf{e}_r\right)^\top}{\sqrt{d}} \tag{9}$$

Among them, h, r, and t represent the head entity, relationship, and tail entity of the knowledge triplet, respectively, and e_h, e_r, e_t is the embedding of the three; $W^Q, W^K, \in R^{d\times d}$ is a trainable attention weight matrix, d is the embedding dimension, representing element wise multiplication. The score of the triplet indicates its importance in helping user preferences. In order to ensure comparability of weighted scores between neighbors of the same entity, the softmax function is used to standardize the weighted scores of neighbors as shown in Eq. (10):

$$\pi(h, r, t) = \frac{\exp(f(h, r, t))}{\sum_{(h,r't')=N_n} \exp\left(h, r', t'\right)} \tag{10}$$

Among them, N_n represents the set of triplets with the same head entity. In the knowledge aggregation step, the model embeds relational context into neighboring entities and assigns dynamic importance weights to them using the weighting function defined in Eq. (10). This weighted neighbor information is then aggregated for the head entity as described in Eq. (11).

$$\boldsymbol{e}_h^{(I)} = \frac{1}{|\mathcal{N}_h|} \sum_{(h,r_i)\in\mathcal{N}_h} \pi(h, r, t)\boldsymbol{e}_r \|\boldsymbol{e}_r^{(I-1)} \tag{11}$$

Among them, N_n is the first-order neighbor set of h in the knowledge graph, and 1 represents the number of layers in the aggregator. The project diagram construction helps uncover semantic relationships between projects, while Eq. (11) learns item representations from the knowledge graph through iterative aggregation of related and neighboring entities. Additionally, cosine similarity is used to build a project graph based on semantic relevance, and Eq. (12) computes similarity between projects I and J.

$$S_{ij} = \frac{\mathbf{e}_i^\top \mathbf{e}_j}{\|\mathbf{e}_i\| \left\|\mathbf{e}_j\right\|} \tag{12}$$

Due to the high computational complexity of fully connected graphs and the potential introduction of noise and unimportant edges, N_k sparsity is applied to the resulting dense graph. For item i, only the edges with top-k confidence scores are retained. The process is shown in Eq. (13)):

$$\hat{S}_{ij} = \left\{ \begin{array}{ll} S_{ij}, S_{ij} \in \text{ top } -k\left(S_i\right) \\ 0, \quad \text{ot herwi se} \end{array} \right\} \tag{13}$$

After obtaining the sparse directed graph adjacency matrix, the normalization matrix $\hat{S}$ of the item graph is obtained according to Eq. (13), where D is the angle matrix and $D_{ij} = \sum_j \hat{S}_{ij}$ After constructing the project graph, Light GCN is used for aggregation operations to learn project representations, which removes feature transformations and nonlinear activations that contribute little to recommendation performance in ordinary graph convolutional networks. The message passing and aggregation operations are shown in Eq. (14):

$$e_i^{(I)} = \sum_{j\in N_l} \tilde{S}_j^{(l-1)} \tag{14}$$

Among them, N_i is the set of neighboring projects of project i, $\hat{\mathrm{S}}$ is the normalized adjacency matrix of the project graph, 1 represents the current aggregation level, and $e_j^{(0)}$ is the ID embedding of the project. Finally, add the project representations of each layer in the project graph aggregation L times to obtain the final project representation:

$$e_i^j = e_i^{(0)} + e_i^{(1)} + \cdots + e_i^{(L_i)} \tag{14}$$

Among them, L_r represents the number of layers in the graph aggregation layer of the project diagram. The computational complexity of the proposed model is mainly influenced by graph denoising, clustering, and attention-based aggregation, with operations scaling linearly with the number of edges, neighbors, and embedding dimensions. Overall, the model remains efficient and scalable, making it suitable for large-scale recommendation datasets with manageable computational cost.

3. Results and discussions

3.1. Experimental dataset

Experiments use Last.FM and MovieLens-1M datasets, applying implicit feedback conversion, negative sampling,

and MCCLK-based knowledge graph construction for evaluation. In the knowledge graph construction process, entities are extracted from the original datasets and external sources such as the Microsoft Satori knowledge base to enrich item information. Relations are defined based on semantic associations, including attributes like genre and category. During preprocessing, low-confidence triplets are removed and entity alignment is performed to ensure consistency, thereby improving the quality, transparency, and reproducibility of the framework. Using MovieLens-1M, movie-related triplets are extracted from Microsoft Satori with confidence >0.9, matched with dataset movie IDs, filtered for valid entities, and aligned with head triplets for final construction. Table 1 shows statistics.

Table 1. Dataset Introduction.

	MovieLens-1M			Last.FM	
Number of users		6038		1	870
Number of projects		2444		3	844
Number of interactions	755		770	43	344
Number of entities	180		11	9	366
Number of relationships		13		60	
Number of triplets	1	240	999	15	517

The statistics presented in Table 1 reflect the scale of the knowledge graph and user–item interaction data used in this study. These components are essential for constructing the knowledge graph representation, which is further utilized in the attention-based knowledge aggregation module to learn enriched item and user embeddings.

3.2. Comparison Model

The proposed model is compared with several baseline methods. CKE integrates structural, textual, and visual features using TransR embeddings. RippleNet models user preferences through knowledge graph propagation. KGCN applies graph convolution to capture high-order relationships. KGAT introduces attention mechanisms in collaborative knowledge graphs. KGIN models user intent using relational paths, while MCCLK enhances representations through multi-view contrastive learning across collaborative, semantic, and structural perspectives.

3.3. Experimental setup and evaluation criteria

The model in this article is built using the PyTorch framework, with experimental environments including Python 3.8, NumPy1.23, and SciKit Learn 1.3.2. All experiments were conducted on a system equipped with a standard processor configuration and sufficient memory capacity to support model training. The implementation was carried out using PyTorch along with supporting libraries such as NumPy and Scikit-learn to ensure efficient computation and reproducibility. The model parameters are initialized using the Xavier initialization method to ensure stable training. The training process is performed over multiple iterations until convergence, where model parameters are updated in each epoch based on the optimization objective. The batch size is set to 4096 by default, and the temperature hyperparameter is set to 0.2. During training, the Adam optimizer is employed for parameter optimization with a predefined learning rate. A fixed learning rate strategy is adopted for stable convergence, and the batch size is set to 4096. The model is trained iteratively until convergence, and early stopping is applied based on validation performance to prevent overfitting. The stable convergence during the dynamic adjustment of knowledge triplet weights, the proposed model employs a controlled optimization strategy. Specifically, the attention weights assigned to knowledge triplets are updated gradually through backpropagation using a bounded activation function and normalized via softmax, preventing abrupt weight fluctuations. The top-k is set to 10 when constructing the project graph. Table 2 presents hyperparameters for both datasets, where d denotes the embedding dimension, and 1 represents the learning rate, tuned in $\{1\times10^{-3}, 3\times10^{-3}, 1\times10^{-4}, 3\times10^{-4}\}$. The L_u regularization coefficient is tuned in $\{10^{-3}, 10^{-4}, 10^{-5}, 10^{-6}\}$ .L_c, L_u, and L_i denote aggregation levels of the collaborative knowledge map, user-project interaction map, and project interaction map, respectively, each adjusted in $\{1, 2, 3\}$. After training, evaluation is conducted on test data under CTR prediction and Top-K recommendation scenarios. CTR performance is measured using AUC (Area Under Curve) and F1 metrics. The Top-K recommendation task is optimized using traditional ranking objectives like pairwise and pointwise methods, aiming to maximize scores of observed interactions and minimize unobserved ones. Unlike conventional approaches treating all negatives equally, the proposed strategy emphasizes ranking consistency within top-K candidates, prioritizing highly relevant items and improving effectiveness in real-world scenarios where only top-ranked recommendations matter.

Recommend the top K items predicted by the model to users, and choose from the top-K recommendation options Recall@K evaluate the predictive performance using evaluation metrics, with K values set to 5, 10, 20, 50, and 100 respectively, to assess the performance of the model. Recall@K measures the proportion of relevant items successfully retrieved within the top-K recommended list. A higher Recall@K value indicates that the model is more

effective in ranking relevant items at top positions, thereby reflecting better recommendation quality and ranking performance. Each experiment was repeated 5 times with different random initializations, and the average performance was reported.

For AUC measures the probability that a randomly chosen positive sample is ranked higher than a negative one by comparing all positive–negative pairs based on predicted scores. Recall@K evaluates performance by selecting top-K ranked items for each user and computing the proportion of relevant items within this list, reflecting recommendation effectiveness.

Table 2. Model hyperparameter settings.

-	d	1	λ	L_c	L_u	L_1
MovieLens-1M	64	0.002	1×10^{-5}	2	1	1
Last.FM	64	0.0002	1×10^{-5}	2	1	2

The hyperparameters listed in Table 2, such as embedding dimension (d) and aggregation layers, directly influence the representation learning process in both the knowledge graph and the attention-based aggregation module.

3.4. Analysis and Outcomes of the Experiments

3.4.1. Model Comparison Experiment

Table 3. Model Comparison Experiment.

Model	MovieLens-1M		Last.FM	
	AUC	F1	AUC	F1
CKE	0.905	0.801	0.745	0.676
Ripple Net	0.921	0.844	0.777	0.705
KGCN	0.911	0.836	0.805	0.707
KGAT	0.916	0.844	0.927	0.744
KGIN	0.917	0.844	0.847	0.762
MCCLK	0.933	0.865	0.875	0.803
ASDCLK	0.936	0.866	0.885	0.805

Table 3 shows ASDCLK outperforms baselines on MovieLens-1M and Last.FM in AUC and F1, improving MCCLK via denoising, attention, and contrastive learning. Table 3 AUC and F1 results predicted by CTR. Top-K recommendation is evaluated using Recall@K, where ASDCLK outperforms advanced baselines on both MovieLens-1M and Last.FM datasets. It achieves significant improvements, including 13.01% on MovieLens-1M and 11.5% on Last.FM over MCCLK, showing better preference modeling and accuracy.

The performance improvements observed in Fig. 1 can be attributed to the integration of knowledge graph representation with the attention-based aggregation mechanism. The model effectively captures structural relationships and assigns adaptive importance to different knowledge triplets, resulting in enhanced recommendation accuracy.

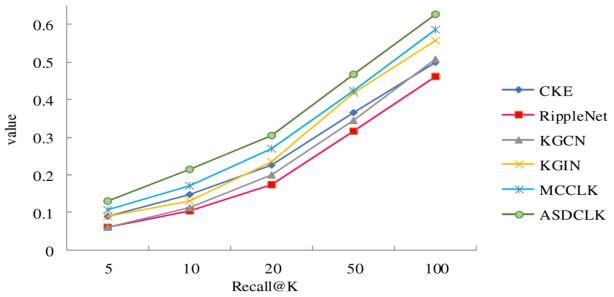

Fig. 1. Recall@K Comparison results of indicators.

3.4.2. Graph Structure Denoising Sampling Probability Super Parameter Experiment

Experiments tested pruning probabilities 0.1, 0.3, 0.5, 0.7 using AUC on two datasets, with best performance at 0.5. This setting balances noise removal and information retention, improving accuracy and reducing overfitting through effective graph denoising.

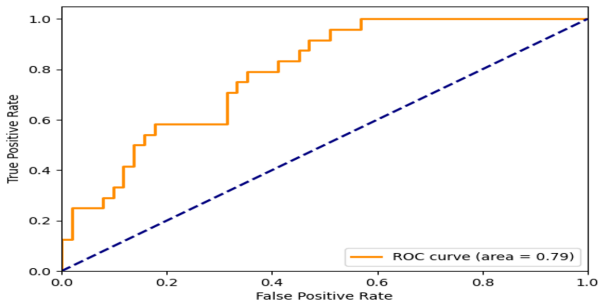

Fig. 2. AUC results of different pruning probabilities on Last.FM.

The variations in AUC values observed in Fig. 22 demonstrate the impact of graph structure denoising on the knowledge graph representation. The pruning probability directly affects the quality of connections between entities, which in turn influences the attention-based knowledge aggregation process and the overall recommendation performance.

As illustrated in Fig. 3, the pruning probability is crucial for balancing noise removal and information preservation in the knowledge graph. This balance improves the attention-based aggregation module by retaining meaningful structural relationships and enhancing overall model performance.

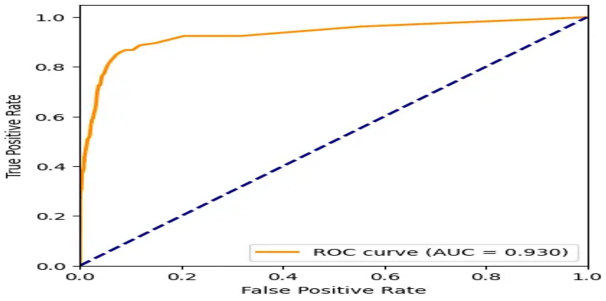

Fig. 3. AUC results of different pruning probabilities on MovieLens-1M.

4. Conclusion

This article proposes a deep attention network based on cluster design (ASDCLK) to solve the common problem of user item interaction graph noise in recommendation systems and the challenge of dynamically adjusting the importance of neighboring entities in the knowledge aggregation process. By introducing a degree sensitive graph structure denoising module, this paper effectively sparsifies the user item interaction graph, removes noise interference, and preserves key interaction information. Based on the Last.FM and MovieLens-1M datasets in particular, the experimental results demonstrate that ASDCLK performs much better than current state-of-the-art models in real-world scenarios involving music and movie selection. In future work will apply ASDCLK to large-scale real-world recommendation systems like e-commerce, streaming services, and content platforms to improve accuracy, personalization, and user experience in dynamic interaction environments.

References

[1] M. Sangeetha and M. D. Thiagarajan, (2023) *"Attribute Preserving Recommendation System Based on Graph Attention Mechanism"* **Journal of Intelligent & Fuzzy Systems** ***44***(6): 9419–9430. DOI: 10.3233/JIFS-223775.

[2] J. Liu, W. Wang, B. Yi, X. Shen, and H. Zhang, (2024) *"Contrastive Multi-Interest Graph Attention Network for Knowledge-Aware Recommendation"* **Expert Systems with Applications**: 124748. DOI: 10.1016/j.eswa.2024.124748.

[3] Q. Wang, H. Cui, J. Zhang, Y. Du, Y. Zhou, and X. Lu, (2023) *"Neighbor-Augmented Knowledge Graph Attention Network for Recommendation"* **Neural Processing Letters** ***55***(6): 8237–8253. DOI: 10.1007/s11063-023-11310-4.

[4] Y. Li, L. Hou, and J. Li, (2023) *"Preference-Aware Graph Attention Networks for Cross-Domain Recommendations with Collaborative Knowledge Graph"* **ACM Transactions on Information Systems** ***41***(3): 1–26. DOI: 10.1145/3576921.

[5] T. Ma, L. Huang, Q. Lu, and S. Hu, (2023) *"KR-GCN: Knowledge-Aware Reasoning with Graph Convolution Network for Explainable Recommendation"* **ACM Transactions on Information Systems** ***41***(1): 1–27. DOI: 10.1145/3511019.

[6] Q. Li, Z. Zhang, F. Zhuang, Y. Xu, and C. Li, (2023) *"Topic-Aware Intention Network for Explainable Recommendation with Knowledge Enhancement"* **ACM Transactions on Information Systems** ***41***(4): 1–23. DOI: 10.1145/3579993.

[7] J. Zhang, Y. Li, R. Zou, J. Zhang, R. Jiang, Z. Fan, and X. Song, (2024) *"Hyper-Relational Knowledge Graph Neural Network for Next POI Recommendation"* **World Wide Web** ***27***(4): 1–19. DOI: 10.1007/s11280-024-01279-y.

[8] S. Liang, J. Shao, J. Zhang, and B. Cui, (2023) *"Graph-Based Non-Sampling for Knowledge Graph Enhanced Recommendation"* **IEEE Transactions on Knowledge and Data Engineering** ***35***(9): 9462–9475. DOI: 10.1109/TKDE.2023.3240832.

[9] Y. Yang, C. Zhang, X. Song, Z. Dong, H. Zhu, and W. Li, (2023) *"Contextualized Knowledge Graph Embedding for Explainable Talent Training Course Recommendation"* **ACM Transactions on Information Systems** ***42***(2): 1–27. DOI: 10.1145/3597022.

[10] F. Akram, T. Ahmad, and M. Sadiq, (2024) *"Recommendation systems-based software requirements elicitation process—a systematic literature review"* **Journal of Engineering and Applied Science** ***71***(1): 29. DOI: 10.1186/s44147-024-00363-4.

[11] R. Zhang, H. Ma, Q. Li, Y. Wang, and Z. Li, (2023) *"FIRE: Knowledge-Enhanced Recommendation with Feature Interaction and Intent-Aware Attention Networks"* **Applied Intelligence** ***53***(13): 16424–16444. DOI: 10.1007/s10489-022-04300-x.

[12] Y. Zhang, X. Wu, Q. Fang, S. Qian, and C. Xu, (2023) *"Knowledge-Enhanced Attributed Multi-Task Learning for Medicine Recommendation"* **ACM Transactions on Information Systems** ***41***(1): 1–24. DOI: 10.1145/3527662.

[13] D. Wang, X. Zhang, Y. Yin, D. Yu, G. Xu, and S. Deng, (2023) *"Multi-View Enhanced Graph Attention Network for Session-Based Music Recommendation"* **ACM Transactions on Information Systems *42***(1): 1–30. DOI: 10.1145/3592853.

[14] H. Xia, K. Huang, and Y. Liu, (2023) *"Unexpected Interest Recommender System with Graph Neural Network"* **Complex & Intelligent Systems *9***(4): 3819–3833. DOI: 10.1007/s40747-022-00849-9.

[15] S. Li, B. Yang, and D. Li, (2023) *"Entity-Driven User Intent Inference for Knowledge Graph-Based Recommendation"* **Applied Intelligence *53***(9): 10734–10750. DOI: 10.1007/s10489-022-04048-4.

[16] D. Cai, S. Qian, Q. Fang, J. Hu, and C. Xu, (2023) *"User Cold-Start Recommendation via Inductive Heterogeneous Graph Neural Network"* **ACM Transactions on Information Systems *41***(3): 1–27. DOI: 10.1145/3560487.

[17] X. Zhang and M. Gan, (2024) *"Hi-GNN: Hierarchical Interactive Graph Neural Networks for Auxiliary Information-Enhanced Recommendation"* **Knowledge and Information Systems *66***(1): 115–145. DOI: 10.1007/s10115-023-01949-9.

[18] H. G. Sri, (2021) *"Integrating HMI Display Module into Passive IoT Optical Fiber Sensor Network for Water Level Monitoring and Feature Extraction"* **World Journal of Advanced Engineering Technology and Sciences *2***(1): 132–139. DOI: 10.30574/wjaets.2021.2.1.0087.

[19] Y. Wu, (2024) *"Exploration of the Integration and Application of the Modern New Chinese Style Interior Design"* **International Journal for Housing Science and Its Applications *45***(2): 28–36.

[20] P. Chen, (2024) *"Research on Business English Approaches from the Perspective of Cross-Cultural Communication Competence"* **International Journal for Housing Science and Its Applications *45***(2): 13–22.

[21] W. Wang, (2024) *"ESG Performance on the Financing Cost of A-Share Listed Companies and an Empirical Study"* **International Journal for Housing Science and Its Applications *45***(2): 1–7.