Enhanced Graph Similarity Learning via Adaptive Multi-scale Feature Fusion

Cuifang Zou[1], Guangquan Lu[1,2,*], Wenzhen Zhang[1], Xuxia Zeng[2], Shilong Lin[2], Longqing Du[2], Shichao Zhang[1,2]

[1]Key Lab of Education Blockchain and Intelligent Technology, Ministry of Education, Guangxi Normal University, Guilin, China

[2]Guangxi Key Lab of Multi-source Information Mining & Security, Guangxi Normal University, Guilin, China

zcf2023@stu.gxnu.edu.cn, lugq@mailbox.gxnu.edu.cn, jianjiu17@outlook.com, {zxx09051230,shilonglin,dulongqing}@stu.gxnu.edu.cn, zhangsc@mailbox.gxnu.edu.cn

Abstract

Graph similarity computation plays a crucial role in a variety of fields such as chemical molecular structure comparison, social network analysis and code clone detection. However, due to inadequate feature representation, existing methods often struggle to cope with complex graph structures, which in turn limits the feature fusion capability and leads to low accuracy of similarity computation. To address these issues, this paper introduces an Adaptive Multi-scale Feature Fusion(AMFF) framework. AMFF firstly enhances feature extraction through a residual graph neural network, which robustly captures key information in complex graph structures. Based on this, a multi-pooled attention network is used to aggregate multi-scale features and accurately extract key node features while minimizing information loss. Finally, the adaptive multi-scale feature fusion mechanism dynamically adjusts the feature fusion weights according to the interactions between nodes and graph embeddings, thus improving the accuracy and sensitivity of similarity computation. Extensive experiments on benchmark datasets including AIDS700nef, LINUX, IMDBMulti, and PTC show that AMFF significantly outperforms existing methods on several metrics. These results confirm the efficiency and robustness of AMFF in graph similarity computation, providing a promising solution for assessing the similarity of complex graph data.

1 Introduction

Graph similarity computation is a fundamental task with diverse applications, including chemical molecular structure comparison, social network analysis, and code clone detection. In the chemical field, it aids in tasks like molecular activity prediction, drug discovery, and compound screening [Reiser *et al.*, 2022; Herrera *et al.*, 2024]. Social network analysis understands social dynamics and improves recommendation systems by comparing network structures [Wu *et al.*, 2022; Du *et al.*, 2020]. In the field of software engineering, it facilitates code optimization and maintenance by efficiently detecting code clones [Liu *et al.*, 2023b; Zhang and Saber, 2024]. Its performance directly impacts downstream tasks such as molecular activity prediction, community dynamics analysis, and software optimization. However, the irregularity and complexity of graph data—characterized by diverse node and edge features as well as multi-scale structural dependencies—pose significant challenges for accurate and efficient similarity computation. Traditional methods and deep learning approaches each face unique limitations in this domain, motivating the need for novel solutions.

Traditional methods, such as graph isomorphism algorithms (e.g., VF2)[Wu *et al.*, 2023] and approximate heuristic approaches (e.g., Graph Edit Distance, GED)[Blumenthal and Gamper, 2020], compute similarity based on graph structure. While graph isomorphism algorithms excel at exact matching, they suffer from high computational complexity and poor scalability to large-scale or attribute-rich graphs. Approximate methods like GED improve efficiency by estimating similarity via edit costs but struggle with noisy or heterogeneous graph data. Furthermore, these methods are often computationally prohibitive for large datasets and fail to capture subtle differences in complex graph structures due to their reliance on rigid matching criteria.

Deep learning-based methods, such as GCN[Bhatti *et al.*, 2023], GAT[Dong *et al.*, 2022], and GraphSAGE[Ding *et al.*, 2021], have introduced a paradigm shift in graph similarity computation. These methods leverage the power of automatic feature learning to extract graph embeddings and capture complex associations between graph structures and node features. GCNs aggregate local information through convolutional operations, while GATs introduce attention mechanisms to adaptively assign weights to neighboring nodes. Despite these advances, existing deep learning methods still face three core challenges: (1)Inadequate multi-level feature fusion: Current methods often fail to integrate local and global structural features, leading to incomplete graph representations. (2)Limited robustness: Many models exhibit performance degradation when handling noisy, heterogeneous, or large-scale graphs[Liu *et al.*, 2023c]. (3)Static fusion strate-

* Corresponding author

gies: Methods relying on fixed feature fusion techniques (e.g., simple concatenation or weighted averaging) lack the adaptability needed to address the diverse characteristics of graph data[Ju *et al.*, 2024].

To address these challenges, this paper proposes the Adaptive Multi-scale Feature Fusion (AMFF), a novel solution designed to enhance feature extraction, fusion, and similarity computation for complex graph data. The framework introduces three key components, each tailored to tackle a specific limitation:

(1)Residual Graph Neural Networks (R-GIN): By incorporating a residual mechanism, this module enhances multi-layer feature representation, enabling the model to robustly capture both local and global structural dependencies while addressing gradient vanishing and over-smoothing issues.

(2)Multi-Pooling Attention Network: This module combines multi-scale feature aggregation with an attention mechanism to retain critical node information while minimizing information loss, ensuring a comprehensive representation of graph features.

(3)Adaptive Multi-scale Feature Fusion: A novel mechanism that dynamically adjusts attention weights based on interactions between node embeddings and graph embeddings, allowing for flexible and precise feature fusion. This approach improves the model's sensitivity and accuracy in similarity computation.

The proposed framework effectively bridges the gap between local and global information while ensuring robustness to complex graph data. Extensive experiments on benchmark datasets, including AIDS700nef, LINUX, IMDBMulti, and PTC, demonstrate that AMFF significantly outperforms existing methods across multiple metrics. These results highlight the efficiency and scalability of AMFF, establishing it as a promising solution for graph similarity computation tasks in diverse domains.

2 Related Work

2.1 Graph Representation Learning

Graph representation learning transforms the nodes, edges, and structures of a graph into low-dimensional embeddings that capture both structural and node-level features, supporting tasks such as node classification, graph classification, and graph similarity computation[Dong *et al.*, 2020]. To handle large-scale graph data, graph embedding methods reduce computational complexity and enhance efficiency by automatically learning effective low-dimensional representations. These techniques can be broadly categorized into methods based on matrix decomposition, such as DeepWalk[Perozzi *et al.*, 2014] and Metapath2Vec[Chen *et al.*, 2023], and those based on deep neural networks, including GCN and GAT. In recent years, graph neural networks (GNNs), such as GCN, GAT, and GraphSAGE, have leveraged the message-passing mechanism to effectively capture local structural features of graphs, driving advancements in graph similarity computation. GNN-based methods evaluate inter-graph similarity by employing metrics like Euclidean distance or cosine similarity through graph embeddings that map node features into low-dimensional spaces. Additionally, models such as graph autoencoders (GAE)[Li *et al.*, 2022a] and variational graph autoencoders (VGAE)[Kipf and Welling, 2016] further improve the efficiency of complex graph similarity evaluation through unsupervised learning.

2.2 Feature Fusion

Graph data contains a variety of complex features, and a single feature cannot fully describe the graph similarity[Wu *et al.*, 2020]. Therefore, feature fusion is crucial in improving the accuracy and robustness of graph similarity computation. Graph data usually has node-level and graph-level features, and multi-scale feature fusion can generate a more comprehensive graph representation. Traditional GCNs mainly aggregate local information and tend to ignore the global structure. To enhance global perception, commonly employed feature fusion methods include weighted summation, concatenation and attention mechanism. Notably, models leveraging the attention mechanism can dynamically adjust feature weights, thereby optimizing the fusion effect. In recent years, methods such as multimodal graph embedding [Islam *et al.*, 2023] and graph self-attention mechanism [Wu and Zhou, 2025] have further enhanced the feature fusion capability. However, feature fusion still faces challenges, such as information distortion due to differences in feature size and distribution, as well as computational efficiency and memory issues in large-scale graph data processing.

3 Problem Formulation

Graph similarity computation aims to quantify the similarity between two graphs based on their structural and feature-level characteristics. A graph is defined as $G = (V, E)$, where V is the set of nodes, E is the set of edges, and $n = |V|$ is the number of nodes. We focus on undirected, unweighted graphs. The structural information is represented by the adjacency matrix $A \in \mathbb{R}^{n\times n}$, and the node features by the feature matrix $X \in \mathbb{R}^{n\times d}$, where d is the feature dimensionality.

Given a reference dataset $\mathcal{D} = \{G_1, G_2, \ldots, G_m\}$ and a query set $\mathcal{Q} = \{Q_1, Q_2, \ldots, Q_k\}$, the goal is to define a similarity function s that assigns a normalized score $S_{ij} \in [0, 1]$ to each graph pair (G_i, Q_j):

$$S_{ij} = s(G_i, Q_j), \quad s : \mathcal{D} \times \mathcal{Q} \to [0, 1] \tag{1}$$

Here, the function s takes as input the Cartesian product $D\times Q$ of sets D and Q, which represents pairs of graphs from the datasets D and Q. It outputs a score within the range [0,1], where a score closer to 1 indicates high similarity between the graphs, and a score closer to 0 indicates dissimilarity.

4 Adaptive Multi-scale Feature Fusion for Graph Similarity Learning

To overcome the limitations existing in the current graph similarity computation, this paper proposes the AMFF framework shown in Fig. 1. The framework aims to achieve more efficient and accurate graph similarity assessment through deep feature extraction and adaptive multi-scale feature fusion techniques. In the following, the implementation details of the AMFF model are described in detail, including three parts: node feature extraction, multi-pooling attention network, and adaptive multi-scale feature fusion module.

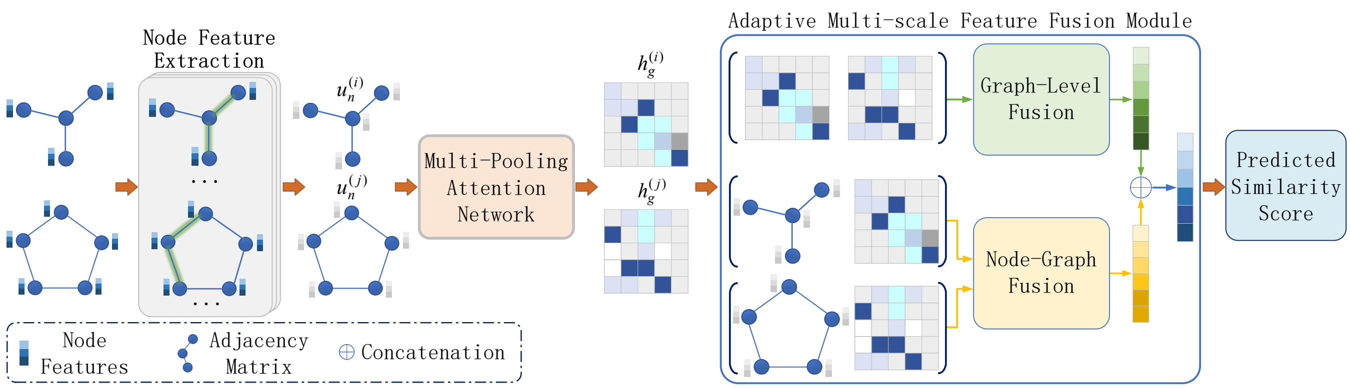

Figure 1: The framework of the AMFF model. The model takes adjacency matrices and one-hot encoded feature matrices of graph pairs as input. The Node Feature Extraction module generates node-level feature matrices for each graph. These are then processed by the Multi-Pooling Attention Network to obtain graph-level features. Both node-level and graph-level features are fused through the Node-Graph Fusion module and the Graph-Level Fusion module, respectively. The resulting similarity scores are concatenated and passed through a fully connected layer to compute the predicted similarity score.

4.1 Node Feature Extraction

In the AMFF framework, node feature extraction is a fundamental step in computing graph similarity. Inspired by the residual connectivity mechanism in Residual Neural Networks[Li *et al.*, 2022b; Amelio *et al.*, 2023], we design a novel Residual Graph Neural Network (R-GIN) for efficiently extracting node features and enhancing the stability of the model in complex graph structures. The input of the module consists of an adjacency matrix A and a node feature matrix X, where X uses a unique thermally encoded vector to represent the features of each node. The R-GIN enhances the model's learning capability by adapting the residual connectivity mechanism to graph convolutional layers. The output dimensions of the three R-GIN layers are 64, 32 and 16, respectively.

The first two layers aggregate the features of neighbouring nodes through graph convolution operations:

$$\begin{aligned} H^{(1)} &= \mathrm{relu}(\mathrm{GINConv}_1(X, A)) \\ H^{(2)} &= \mathrm{relu}(\mathrm{GINConv}_2(H^{(1)}, A)) \end{aligned} \tag{2}$$

where X is the input node feature matrix, and A represents the adjacency matrix (or edge index). To effectively combine information from multiple layers, the output of the first layer ($H^{(1)}$) is transformed using a linear operation and fused with the output of the second layer ($H^{(2)}$) via residual joins:

$$H^{(\text{dense})} = \mathrm{Linear}(H^{(1)}) + H^{(2)} \tag{3}$$

where $\mathrm{Linear}(\cdot)$ is a linear transformation. This feature fusion strategy preserves the information from earlier layers and seamlessly integrates it into deeper features.

Finally, the fused features are passed through the third layer of the graph isomorphism network to generate the final node embeddings:

$$u_n = H^{(3)} = \mathrm{GINConv}_3(H^{(\text{dense})}, A) \tag{4}$$

where u_n denotes the embedding vector of the n-th node. With this multi-layer design, the model is able to effectively capture shallow and deep node representations, while residual joins alleviate problems such as over-smoothing or gradient vanishing, and improve feature quality. These high-quality node features are subsequently used in downstream similarity assessment tasks.

4.2 Multi-Pooling Attention Network

Traditional pooling methods, such as average pooling (P_{avg}) and maximum pooling (P_{max}), generate graph-level representations by aggregating node features, but may lose important information or pay excessive attention to extreme values[Woo *et al.*, 2018; Liu *et al.*, 2023a]. To address these issues, we design the Multi-Pooling Attention Network (MPA), which combines multi-pooling and attention mechanisms to dynamically adjust node contributions to the graph-level representation.

First, the module extracts global features and locally salient features of the graph through average pooling and maximum pooling:

$$P_{\text{multi}} = P_{\text{avg}} \oplus P_{\text{max}} \tag{5}$$

where P_{avg} and P_{max} are defined as:

$$P_{\text{avg}} = \frac{1}{N}\sum_{i=1}^{N} u_n^{(i)} \tag{6}$$

$$P_{\text{max}} = \max_{i=1}^{N} u_n^{(i)} \tag{7}$$

Here, P_{avg} captures the overall distribution of features, and P_{max} identifies the most salient features. $u_n^{(i)}$ denotes the feature vector of the i-th node of graph g, N is the total number of nodes, and $\oplus$ represents the concatenation operation, which combines the results of the two pooling strategies.

The pooling results are then nonlinearly transformed by a multilayer perceptron (MLP)[Gardner and Dorling, 1998] to generate a more expressive global feature representation:

$$P_{\text{multi}} = \sigma(\mathrm{MLP}(P_{\text{multi}})) \tag{8}$$

Next, the attention weights are computed, the global features are interacted with each node feature, and the node features are adjusted by weighting to finally generate a graph-level representation, see Eq.9:

$$\begin{aligned} h_g &= \sum_{n=1}^{N} \sigma\left(u_n^T c\right) u_n, \\ h_g &= \sum_{n=1}^{N} \sigma\left(u_n^T \tanh\left(P_{multi} \times W\right)\right) u_n \end{aligned} \tag{9}$$

where σ is an activation function, W denotes a learnable weight matrix.

MPA combines global pooling with a local attention mechanism to enhance the robustness of graph-level representations, which plays an important role in graph similarity computation in the AMFF framework.

4.3 Adaptive Multi-scale Feature Fusion Module

In order to further enhance the model's ability to assess graph similarity, the AMFF framework introduces an adaptive multi-scale feature fusion module. This module is divided into two parts: node-graph feature interaction and graph-level feature interaction.

Node-Graph Adaptive Feature Fusion(NGFusion)

The node-graph adaptive feature fusion module aims at the weighted fusion of node embedding and graph embedding information by means of an adaptive attention mechanism and generates a comprehensive representation for calculating similarity. This module mainly consists of the following steps:

For the input node embedding u_n and batch index, each batch of nodes is embedded into a graph-level node representation using a weighted aggregation operation.

$$u_m^{(i)} = f(u_n^{(i)}, batch) \tag{10}$$

where f is an aggregation function used to summarise dispersed node information into a graph-level representation by batch index. After splicing the node embeddings with the graph embeddings, the attention weights are generated by a linear transformation.

$$\alpha^{(i)} = \sigma(W(u_m^{(i)} \oplus h_g^{(i)})) \tag{11}$$

The node features are weighted using the attention weights $\alpha^{(i)}$ to obtain an updated node representation. After splicing the weighted node embeddings with the graph embeddings again, the integrated graph-level features are generated by MLP:

$$h_{fusion}^{(i)} = \text{MLP}((\alpha^{(i)} \odot u_m^{(i)}) \oplus h_g^{(i)}) \tag{12}$$

where $\odot$ is element-wise Multiplication.

Ultimately, the similarity of the composite features $h_{fusion}^{(i)}$ and $h_{fusion}^{(j)}$ of a graph pair is quantified by the Euclidean distance metric in the pairwise distance function, see Eq.13:

$$\begin{aligned} s_{\text{NG}} &= \exp\left(\|h_{fusion}^{(i)} - h_{fusion}^{(j)}\|_2\right) \\ \|h_{fusion}^{(i)} - h_{fusion}^{(j)}\|_2 &= \sqrt{\sum_{k=1}^{d}\left(h_{fusion,k}^{(i)} - h_{fusion,k}^{(j)}\right)^2} \end{aligned} \tag{13}$$

where d denotes the dimension of the embedding vector, and $h_{fusion,k}^{(i)}$ and $h_{fusion,k}^{(j)}$ are the components of the fusion features of the graphs g_i and g_j in the k-th dimension, respectively.

NGFusion enables the capture of global interaction features of nodes with the graph and weights and integrates them to make the generated graph-level representation more robust and informative, thus improving the accuracy of similarity computation.

Graph-Level Adaptive Feature Fusion (GLFusion)

The graph-level adaptive feature fusion module combines weighted differences and cosine similarities of graph embeddings to capture key relationships between graph pairs. This design, based on GSLSim [Zou *et al.*, 2025], enhances the feature representation of graph embeddings for a more thorough understanding of graph relationships.

GLFusion first computes the weighted differences between the graph pair embeddings:

$$\Delta h = (h_g^{(j)} - h_g^{(i)})^T W (h_g^{(j)} - h_g^{(i)}) \tag{14}$$

where Δh represents the weighted feature differences that highlight dissimilarities between graph embeddings, $h_g^{(i)}$ and $h_g^{(j)}$ are the graph-level embeddings of the input graphs, and W is a learnable tensor weight matrix used to enhance feature interactions.

The cosine similarity is then computed to evaluate the directional alignment between the two graph embeddings in a normalized vector space:

$$\text{Distance} = \frac{h_g^{(i)} \bullet (h_g^{(j)})^T}{\| h_g^{(i)} \| \| h_g^{(j)} \|} \tag{15}$$

Finally, the graph similarity score is obtained by integrating the weighted differences and cosine similarity:

$$s_{\text{GL}} = \sigma\left(\Delta h \odot \text{Distance} + b\right) \tag{16}$$

where σ is the relu activation function, $\odot$ denotes element-wise multiplication, and b is a bias vector.

By combining graph embedding disparities and alignments, GLFusion effectively captures both differences and similarities between graph pairs, thereby improving the accuracy and robustness of graph similarity computation.

4.4 Similarity Score Calculation

The model combines node-graph and graph-level adaptive information to derive representative similarity features. First, the node-graph adaptive fusion score s_{NG} and the graph-level adaptive fusion score s_{GL} are concatenated to form a unified feature representation. These features are then passed through a fully connected layer to compute the final similarity score:

$$p(g_i, g_j) = FC(s_{\text{NG}}(g_i, g_j) \oplus s_{\text{GL}}(g_i, g_j)) \tag{17}$$

where $p(g_i, g_j)$ is the predicted similarity score, $\oplus$ denotes the concatenation operation, and FC represents the fully connected layer.

Algorithm 1 The Algorithm of AMFF.

Input: Node features X, adjacency matrix A
Output: Graph similarity score $p(g_i, g_j)$

```
Step 1: Node embedding generation
for l = 1, 2 do
    Update node embeddings using Eq.(2).
end for
Perform dense feature fusion H^(dense) using Eq.(3).
Generate final node embeddings u_n using Eq. (4).
Step 2: Graph-level embedding generation
Perform multi-pooling using Eq.(5-8).
Generate graph embeddings h_g using Eq.(9).
Step 3: Adaptive multi-scale feature fusion
for Each graph pair (g_i, g_j) do
    Enter u_n^(i), u_n^(j), h_g^(i) and h_g^(j) into NGFusion and cal-
    culate score s_NG using Eq. (13).
    Input h_g^(i), h_g^(j) into GLFusion and compute score s_GL
    by Eq.(16).
end for
Step 4: Similarity score calculation
Final prediction score p(g_i, g_j) using Eq.(17).
Step 5: Loss calculation
Compute loss L using Eq.(18).
Ground truth similarity t(g_i, g_j) using Eq.(19).
```

To optimize the model, the error between the predicted similarity score and the true similarity score is minimized using the Mean Squared Error (MSE) loss function:

$$\mathcal{L} = \frac{1}{N} \sum_{(i,j) \in N} (p(g_i, g_j) - t(g_i, g_j))^2 \tag{18}$$

where N is the total number of graph pairs in the training set.

The true similarity score $t(g_i, g_j)$ is computed based on the normalized Graph Edit Distance (GED) as follows:

$$t(g_i, g_j) = \lambda \left(\frac{d_{\text{GED}}(g_i, g_j)}{(|g_i| + |g_j|)/2} \right) \tag{19}$$

Here, λ represents an exponential function, $d_{\text{GED}}(\cdot)$ is the Graph Edit Distance function [Riesen *et al.*, 2013], and $|g_i|, |g_j|$ are the number of nodes in graph g_i and g_j, respectively. The comprehensive training methodology for the introduced approach is outlined in Algorithm 1.

5 Experiment

5.1 Datasets

AIDS700nef, 700 chemical graphs with 2-10 atoms (C, O, H) and bonds, suitable for small-scale molecular similarity tasks. **LINUX**, 1000 hierarchical file system graphs with 4-10 nodes (files/folders) and parent-child edges, ideal for testing tree-structured GNNs. **IMDBMulti**, 1500 movie social networks with 4-89 actor nodes and co-occurrence edges, labeled with 29 genres (e.g., action, comedy) for graph similarity and classification. **PTC**, 344 chemical graphs with 2-109 atoms and bonds, labeled with 19 toxicity or chemical properties, benchmark for complex molecular graph processing.

5.2 Baselines

We compare our model with traditional GED methods (A*-beam search[Neuhaus *et al.*, 2006], Hungarian algorithm[Kuhn, 1955], VJ algorithm[Fankhauser *et al.*, 2011]) and neural network baseline methods: SimGNN[Bai *et al.*, 2019], combines graph-level embeddings and node-level attention for similarity computation. GraphSim[Bai *et al.*, 2020], directly matches two sets of node embeddings without relying on fixed graph-level representations. GENN[Wang *et al.*, 2021], combines traditional search with learned embeddings to efficiently solve GED. MGMN[Ling *et al.*, 2021], integrates node-graph and global graph interactions to enhance performance and robustness. H2MN[Zhang *et al.*, 2021], transforms graphs into hypergraphs, enabling high-order representations and multi-perspective subgraph matching. NA-GSL[Tan *et al.*, 2023], uses multiple attention mechanisms for node embeddings, interaction modeling, and similarity prediction. CLSim[Zou *et al.*, 2025], aligns graph pair features via attention, aggregates node features, and integrates node-level and graph-level embeddings for detailed interaction modeling.

5.3 Evaluation Metrics

To comprehensively measure the model's performance, we employ the following evaluation metrics: (1) Mean Squared Error (MSE) quantifies the deviation between predicted and ground truth values, providing an intuitive measure of prediction accuracy. (2) Spearman's Rank Correlation Coefficient (ρ)[Spearman, 1961] and Kendall's Rank Correlation Coefficient (τ)[Kendall, 1938] assess the consistency between predicted and true rankings. These metrics are crucial for evaluating the model's performance on ranking tasks. (3) k-Accuracy (p@k) evaluates the overlap between the top-k predicted results and the top-k ground truth results, reflecting the model's ability to identify relevant items in recommendation or retrieval tasks. These formulas are calculated as follows:

- MSE $= \frac{1}{n} \sum_{i=1}^{n} (\hat{y}_i - y_i)^2$
- $\rho = 1 - \frac{6 \sum_{i=1}^{n} d_i^2}{n(n^2-1)}$
- $\tau = (C - D) / \frac{n(n-1)}{2}$
- p@k $= \frac{|\text{Top-}k(\hat{y}) \cap \text{Top-}k(y)|}{k}$

where n is the total number of samples, $\hat{y}_i$ and y_i are the predicted and true values for the i-th sample, and d_i is the difference between their ranks. C and D denote concordant and discordant pairs, and $\binom{n}{2}$ is the total number of pairwise comparisons. Top-$k(\hat{y})$ and Top-$k(y)$ are the top-k predicted and ground truth results, respectively, with k set to 10 and 20 in this study.

5.4 Effectiveness

We conducted experiments on simple (AIDS700nef, LINUX) and complex (IMDBMulti, PTC) graph-structured datasets to validate the model. The experimental results are clearly presented in Table 1, showing that AMFF performs well on all datasets. Specifically, it outperforms traditional GED methods and deep learning methods in terms of accuracy and

Method	AIDS700nef mse(10^{-3})	AIDS700nef ρ	AIDS700nef τ	AIDS700nef p@10	AIDS700nef p@20	LINUX mse(10^{-3})	LINUX ρ	LINUX τ	LINUX p@10	LINUX p@20
Beam[Neuhaus *et al.*, 2006]	12.090	0.609	0.463	0.481	0.493	9.268	0.827	0.714	0.973	0.924
Hungarian[Kuhn, 1955]	25.296	0.510	0.378	0.360	0.392	29.810	0.638	0.517	0.913	0.836
VJ[Fankhauser *et al.*, 2011]	29.157	0.517	0.383	0.310	0.345	63.860	0.581	0.450	0.287	0.251
SimGNN[Bai *et al.*, 2019]	2.158	0.861	0.689	0.464	0.538	0.465	0.979	0.881	0.954	0.948
GraphSim[Bai *et al.*, 2020]	2.417	0.512	0.672	0.255	0.329	3.173	0.878	0.739	0.200	0.320
GENN[Wang *et al.*, 2021]	2.071	0.877	0.711	0.379	0.481	1.357	0.964	0.842	0.344	0.583
MGMN[Ling *et al.*, 2021]	2.368	**0.902**	**0.746**	0.461	0.535	2.020	0.964	0.852	0.915	0.892
H2MN[Zhang *et al.*, 2021]	**1.017**	0.869	0.717	0.469	0.553	0.334	0.980	0.906	0.940	0.938
NA-GSL[Tan *et al.*, 2023]	2.261	0.877	0.729	0.487	0.569	0.258	**0.991**	**0.957**	0.983	0.974
CLSim[Zou *et al.*, 2025]	1.950	0.875	0.705	0.527	0.593	0.290	0.983	0.892	0.981	0.966
AMFF(ours)	1.538	0.898	0.735	**0.590**	**0.655**	**0.130**	0.988	0.908	**0.988**	**0.981**

Method	IMDBMulti mse(10^{-3})	IMDBMulti ρ	IMDBMulti τ	IMDBMulti p@10	IMDBMulti p@20	PTC mse(10^{-3})	PTC ρ	PTC τ	PTC p@10	PTC p@20
SimGNN[Bai *et al.*, 2019]	0.949	0.846	0.734	0.810	0.809	2.281	0.924	0.781	0.498	0.587
GraphSim[Bai *et al.*, 2020]	15.980	0.515	0.644	0.149	0.562	2.816	0.921	0.776	0.402	0.510
GENN[Wang *et al.*, 2021]	2.565	0.810	0.690	0.407	0.494	-	-	-	-	-
MGMN[Ling *et al.*, 2021]	26.064	0.732	0.542	0.295	0.503	1.763	0.950	0.810	0.478	0.591
H2MN[Zhang *et al.*, 2021]	0.556	0.832	0.729	0.842	0.854	**1.287**	0.912	0.762	0.468	0.587
NA-GSL[Tan *et al.*, 2023]	0.988	0.852	0.769	0.812	0.828	-	-	-	-	-
CLSim[Zou *et al.*, 2025]	0.574	0.929	0.811	0.852	0.848	1.608	0.939	0.802	0.528	0.612
AMFF(ours)	**0.515**	**0.941**	**0.834**	**0.863**	**0.875**	1.334	**0.953**	**0.828**	**0.542**	**0.627**

Table 1: Performance comparison. The best is indicated by bolding, and the second best is underlined. The lower the mse, the better, while the higher the ρ, τ, p@10, p@20, the better. '-' indicates that the method was unable to produce results on the corresponding dataset.

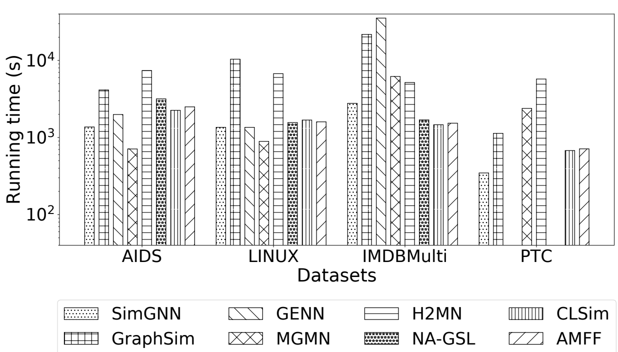

Figure 2: Model runtime comparison.

error metrics. On simple graph-structured datasets, AMFF shows excellent adaptability and efficiency. For complex graph-structured datasets, the model maintains leading performance, fully demonstrating its ability to handle complex graph structures. It is worth noting that some baseline methods (e.g., GENN and NA-GSL) fail to run due to excessive memory when processing complex datasets, while AMFF runs smoothly, which further highlights its storage efficiency and computational stability. In conclusion, the robust performance of AMFF on both simple and complex graph-structured datasets confirms its wide applicability. The model has made great strides in balancing complexity and computational efficiency, highlighting its potential and advantages in graph similarity computation tasks.

5.5 Efficiency

As shown in Fig. 2, AMFF shows strong performance on various datasets in the efficiency analysis. On simple datasets (Linux, AIDS700nef), its running time is comparable to SimGNN and GraphSim, and better than GENN and MGMN. on complex datasets (IMDBMulti, PTC), AMFF outperforms GENN and NA-GSL, and is more efficient than H2MN on PTC, and comparable to CLSim. It is worth noting that AMFF successfully handles all datasets, while some do not, highlighting its stability and reliability in handling graphs of varying complexity.

5.6 Ablation Study

GNN selection in node feature extraction module

Node feature extraction is crucial for improving model performance. To this end, we compare the performance of several mainstream GNN models (GCN [Bhatti *et al.*, 2023], GAT [Vrahatis *et al.*, 2024], GraphSAGE [Bhatkar *et al.*, 2023] and GIN [Liao *et al.*, 2024]) on the same benchmark dataset (see Table 2). The experimental results show that R-GIN performs best on the AIDS700nef and LINUX datasets, especially on the MSE and ρ metrics, demonstrating strong feature extraction capabilities. While on the IMDBMulti and PTC datasets, GCN and two-layer GIN slightly dominate in some metrics (e.g., MSE and p@10). Overall, R-GIN and GIN family perform particularly well in graph structure learning and ranking accuracy.

Impact of individual modules on overall performance

This experiment evaluates the impact of each component in the AMFF framework on model performance, and observes

	AIDS700nef			LINUX			IMDBMulti			PTC		
GNN	mse(10^{-3})	ρ	p@10	mse(10^{-3})	ρ	p@10	mse(10^{-3})	ρ	p@10	mse(10^{-3})	ρ	p@10
gcn	1.836	0.882	0.530	0.229	0.986	0.987	**0.462**	0.943	**0.869**	1.342	0.949	0.536
gat	2.449	0.854	0.444	0.341	0.982	0.976	0.512	0.940	0.864	**1.320**	0.951	0.516
graphsage	1.973	0.878	0.518	0.150	0.987	0.981	0.528	0.938	0.849	1.603	0.950	0.528
gin(1 layer)	1.647	0.891	0.558	0.219	0.987	0.985	0.468	0.940	**0.869**	1.365	0.950	0.542
gin(2 layers)	1.584	0.897	0.579	**0.130**	**0.988**	0.983	0.500	**0.945**	0.863	1.332	0.951	0.520
gin(3 layers)	1.593	0.895	0.571	0.191	0.987	0.985	0.545	0.939	0.855	1.424	0.950	0.528
r-gin	**1.538**	**0.898**	**0.590**	**0.130**	**0.988**	**0.988**	0.515	0.941	0.863	1.334	**0.953**	**0.627**

Table 2: Performance comparison of different graph neural networks in node feature extraction module.

Components					Metrics				
R	MP	GL	NG	Dis	mse	ρ	τ	p@10	p@20
×	√	√	√	√	0.191	0.987	0.904	0.985	0.972
√	×	√	√	√	0.137	0.987	0.907	0.986	0.976
√	√	×	√	√	0.555	0.976	0.874	0.967	0.957
√	√	√	×	√	0.183	0.986	0.904	0.983	0.973
√	√	√	√	×	0.134	0.987	0.906	0.987	0.980
√	√	√	√	√	**0.130**	**0.988**	**0.908**	**0.988**	**0.981**

Table 3: Module ablation experiments. 'R' is the Residual R-GIN Module (Section 4.1). 'MP' is Multi-Pooling Attention (Section 4.2). 'GL' is Graph-Level Fusion (NLFusion in section 4.3). 'NG' is Node-Graph Fusion (NGFusion in section 4.3). 'Dis' is cosine similarity in the GLFusion.

its performance changes on the PTC dataset by gradually disabling the module, see Table 3. The conclusions are summarised as follows: (1) R: after disabling, the feature learning ability decreases and the model accuracy and ranking relevance are slightly affected, which verifies the importance of the residual mechanism on the model performance. (2) MP: the significantly increased error and performance degradation indicate that this module is crucial in capturing multi-scale features. (3) GL: its removal has the largest negative impact on performance, indicating the critical role of graph-level feature fusion in integrating global features. (4) NG: the model accuracy decreases after its disabling, demonstrating the importance of node and graph feature fusion in improving accuracy and robustness. (5) Dis : a slight decrease in performance after removal, showing its positive role in feature fusion. In summary, the best model performance is achieved when all modules are enabled, indicating that these components complement each other in graph similarity computation, and together improve the accuracy and robustness of the model.

5.7 Case Study

Randomly select query graphs in the test set, pair them with all graphs in the training set to generate graph pairs and calculate similarity scores. By sorting, ideally, the true matching graphs of the Query Graph should be ranked top. In order to analyse the model performance intuitively, we designed a visual display (see Fig. 3). For clarity, only the top two query graphs are displayed for each dataset. Each case includes: (1) Real matching results (top): the query graph and its true matching graph set, annotated with real similarity scores ranging from 0 to 1 (where values closer to 1 indicate higher similarity); and (2) Predicted matching results (bottom): the query graph and the most relevant graphs predicted by the model, ordered by ranking and annotated with the predicted rankings. This case study demonstrates the model's applicability across various graph similarity tasks and its ability to handle different data scenarios effectively. The alignment between the predicted and real rankings further underscores the model's robustness and accuracy in graph similarity computation.

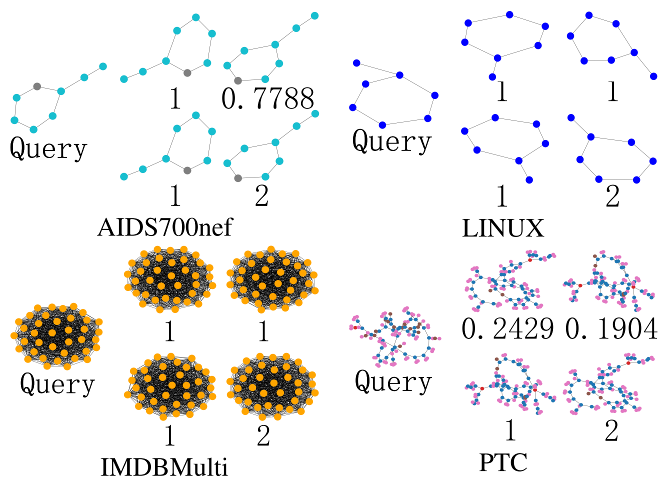

Figure 3: Case study.

6 Conclusion

In this paper, a graph similarity computation framework based on adaptive multi-scale feature fusion (AMFF) is proposed to solve the similarity assessment and feature extraction problems of existing models in complex graph structures. By introducing the residual mechanism and multi-pooling attention mechanism, the feature expression ability and feature aggregation effect are enhanced. Meanwhile, the proposed adaptive feature enhancement fusion strategy dynamically adjusts the attention weights of node embeddings and graph embeddings, which significantly improves the model's sensitivity to graph similarity and computational accuracy. Experimental results show that the framework exhibits significant performance enhancement on multiple datasets, verifying its efficiency and wide applicability.

Acknowledgments

The work is supported partly by the Project of Guangxi Science and Technology (2025GXNSFFA069015), National Natural Science Foundation of China (No. 62372119, 62166003), the Key Lab of Education Blockchain and Intelligent Technology, Ministry of Education(No. EBME24-03), Open Research Fund of Guangxi Key Lab of Human-machine Interaction and Intelligent Decision (No.GXHIID2206), Innovation Project of Guangxi Graduate Education (JGY2024080, JGY2022046), the Guangxi Collaborative Innovation Center of Multi-Source Information Integration, Innovation Project of Guangxi Graduate Education (YCSW2025220).

References

[Amelio *et al.*, 2023] Alessia Amelio, Gianluca Bonifazi, Francesco Cauteruccio, Enrico Corradini, Michele Marchetti, Domenico Ursino, and Luca Virgili. Representation and compression of residual neural networks through a multilayer network based approach. *Expert Systems with Applications*, 215:119391, 2023.

[Bai *et al.*, 2019] Yunsheng Bai, Hao Ding, Song Bian, Ting Chen, Yizhou Sun, and Wei Wang. Simgnn: A neural network approach to fast graph similarity computation. In *Proceedings of the twelfth ACM international conference on web search and data mining*, pages 384–392, 2019.

[Bai *et al.*, 2020] Yunsheng Bai, Hao Ding, Ken Gu, Yizhou Sun, and Wei Wang. Learning-based efficient graph similarity computation via multi-scale convolutional set matching. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 3219–3226, 2020.

[Bhatkar *et al.*, 2023] Sarthak Bhatkar, Purva Gosavi, Vishakha Shelke, and John Kenny. Link prediction using graphsage. In *2023 International Conference on Advanced Computing Technologies and Applications (ICACTA)*, pages 1–5. IEEE, 2023.

[Bhatti *et al.*, 2023] Uzair Aslam Bhatti, Hao Tang, Guilu Wu, Shah Marjan, and Aamir Hussain. Deep learning with graph convolutional networks: An overview and latest applications in computational intelligence. *International Journal of Intelligent Systems*, 2023(1):8342104, 2023.

[Blumenthal and Gamper, 2020] David B Blumenthal and Johann Gamper. On the exact computation of the graph edit distance. *Pattern Recognition Letters*, 134:46–57, 2020.

[Chen *et al.*, 2023] Lei Chen, Yuan Li, Yong Lei, and Xingye Deng. Metarelation2vec: A metapath-free scalable representation learning model for heterogeneous networks. *Tsinghua Science and Technology*, 29(2):553–575, 2023.

[Ding *et al.*, 2021] Yao Ding, Xiaofeng Zhao, Zhili Zhang, Wei Cai, and Nengjun Yang. Multiscale graph sample and aggregate network with context-aware learning for hyperspectral image classification. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 14:4561–4572, 2021.

[Dong *et al.*, 2020] Yuxiao Dong, Ziniu Hu, Kuansan Wang, Yizhou Sun, and Jie Tang. Heterogeneous network representation learning. In *IJCAI*, volume 20, pages 4861–4867, 2020.

[Dong *et al.*, 2022] Yanni Dong, Quanwei Liu, Bo Du, and Liangpei Zhang. Weighted feature fusion of convolutional neural network and graph attention network for hyperspectral image classification. *IEEE Transactions on Image Processing*, 31:1559–1572, 2022.

[Du *et al.*, 2020] Zhi-jiao Du, Han-yang Luo, Xu-dong Lin, and Su-min Yu. A trust-similarity analysis-based clustering method for large-scale group decision-making under a social network. *Information Fusion*, 63:13–29, 2020.

[Fankhauser *et al.*, 2011] Stefan Fankhauser, Kaspar Riesen, and Horst Bunke. Speeding up graph edit distance computation through fast bipartite matching. In *Graph-Based Representations in Pattern Recognition: 8th IAPR-TC-15 International Workshop, GbRPR 2011, Münster, Germany, May 18-20, 2011. Proceedings 8*, pages 102–111. Springer, 2011.

[Gardner and Dorling, 1998] Matt W Gardner and SR Dorling. Artificial neural networks (the multilayer perceptron)—a review of applications in the atmospheric sciences. *Atmospheric environment*, 32(14-15):2627–2636, 1998.

[Herrera *et al.*, 2024] Luis P Taracena Herrera, Søren N Andreassen, Jimmy Caroli, Ismael Rodríguez-Espigares, Ali A Kermani, György M Keserű, Albert J Kooistra, Gáspár Pándy-Szekeres, and David E Gloriam. Gpcrdb in 2025: adding odorant receptors, data mapper, structure similarity search and models of physiological ligand complexes. *Nucleic Acids Research*, page gkae1065, 2024.

[Islam *et al.*, 2023] Md Milon Islam, Sheikh Nooruddin, Fakhri Karray, and Ghulam Muhammad. Multi-level feature fusion for multimodal human activity recognition in internet of healthcare things. *Information Fusion*, 94:17–31, 2023.

[Ju *et al.*, 2024] Wei Ju, Zheng Fang, Yiyang Gu, Zequn Liu, Qingqing Long, Ziyue Qiao, Yifang Qin, Jianhao Shen, Fang Sun, Zhiping Xiao, et al. A comprehensive survey on deep graph representation learning. *Neural Networks*, page 106207, 2024.

[Kendall, 1938] Maurice G Kendall. A new measure of rank correlation. *Biometrika*, 30(1-2):81–93, 1938.

[Kipf and Welling, 2016] Thomas N Kipf and Max Welling. Variational graph auto-encoders. *arXiv preprint arXiv:1611.07308*, 2016.

[Kuhn, 1955] Harold W Kuhn. The hungarian method for the assignment problem. *Naval research logistics quarterly*, 2(1-2):83–97, 1955.

[Li *et al.*, 2022a] Jingci Li, Guangquan Lu, and Zhengtian Wu. Multi-view graph autoencoder for unsupervised graph representation learning. In *2022 26th International Conference on Pattern Recognition (ICPR)*, pages 2213–2218. IEEE, 2022.

[Li *et al.*, 2022b] Wei Li, Tianzhao Yang, Xiao Wu, and Zhaoquan Yuan. Learning graph-based residual aggregation network for group activity recognition. In *IJCAI*, pages 1102–1108, 2022.

[Liao *et al.*, 2024] Bin Liao, Hangxu Zuo, Yang Yu, and Yong Li. Graphmrinet: a few-shot brain tumor mri image classification model based on prewitt operator and graph isomorphic network. *Complex & Intelligent Systems*, pages 1–14, 2024.

[Ling *et al.*, 2021] Xiang Ling, Lingfei Wu, Saizhuo Wang, Tengfei Ma, Fangli Xu, Alex X Liu, Chunming Wu, and Shouling Ji. Multilevel graph matching networks for deep graph similarity learning. *IEEE Transactions on Neural Networks and Learning Systems*, 34(2):799–813, 2021.

[Liu *et al.*, 2023a] Chuang Liu, Yibing Zhan, Xueqi Ma, Liang Ding, Dapeng Tao, Jia Wu, and Wenbin Hu. Gapformer: Graph transformer with graph pooling for node classification. In *IJCAI*, pages 2196–2205, 2023.

[Liu *et al.*, 2023b] Shangqing Liu, Xiaofei Xie, Jingkai Siow, Lei Ma, Guozhu Meng, and Yang Liu. Graphsearchnet: Enhancing gnns via capturing global dependencies for semantic code search. *IEEE Transactions on Software Engineering*, 49(4):2839–2855, 2023.

[Liu *et al.*, 2023c] Zhaowei Liu, Dong Yang, Yingjie Wang, Mingjie Lu, and Ranran Li. Egnn: Graph structure learning based on evolutionary computation helps more in graph neural networks. *Applied Soft Computing*, 135:110040, 2023.

[Neuhaus *et al.*, 2006] Michel Neuhaus, Kaspar Riesen, and Horst Bunke. Fast suboptimal algorithms for the computation of graph edit distance. In *Structural, Syntactic, and Statistical Pattern Recognition: Joint IAPR International Workshops, SSPR 2006 and SPR 2006, Hong Kong, China, August 17-19, 2006. Proceedings*, pages 163–172. Springer, 2006.

[Perozzi *et al.*, 2014] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 701–710, 2014.

[Reiser *et al.*, 2022] Patrick Reiser, Marlen Neubert, André Eberhard, Luca Torresi, Chen Zhou, Chen Shao, Houssam Metni, Clint van Hoesel, Henrik Schopmans, Timo Sommer, et al. Graph neural networks for materials science and chemistry. *Communications Materials*, 3(1):93, 2022.

[Riesen *et al.*, 2013] Kaspar Riesen, Sandro Emmenegger, and Horst Bunke. A novel software toolkit for graph edit distance computation. In *Graph-Based Representations in Pattern Recognition: 9th IAPR-TC-15 International Workshop, GbRPR 2013, Vienna, Austria, May 15-17, 2013. Proceedings 9*, pages 142–151. Springer, 2013.

[Spearman, 1961] Charles Spearman. The proof and measurement of association between two things. 1961.

[Tan *et al.*, 2023] Wenhui Tan, Xin Gao, Yiyang Li, Guangqi Wen, Peng Cao, Jinzhu Yang, Weiping Li, and Osmar R Zaiane. Exploring attention mechanism for graph similarity learning. *Knowledge-Based Systems*, 276:110739, 2023.

[Vrahatis *et al.*, 2024] Aristidis G Vrahatis, Konstantinos Lazaros, and Sotiris Kotsiantis. Graph attention networks: A comprehensive review of methods and applications. *Future Internet*, 16(9):318, 2024.

[Wang *et al.*, 2021] Runzhong Wang, Tianqi Zhang, Tianshu Yu, Junchi Yan, and Xiaokang Yang. Combinatorial learning of graph edit distance via dynamic embedding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5241–5250, 2021.

[Woo *et al.*, 2018] Sanghyun Woo, Jongchan Park, Joon-Young Lee, and In So Kweon. Cbam: Convolutional block attention module. In *Proceedings of the European conference on computer vision (ECCV)*, pages 3–19, 2018.

[Wu and Zhou, 2025] Yuejia Wu and Jian-tao Zhou. A hierarchical and interlamination graph self-attention mechanism-based knowledge graph reasoning architecture. *Information Sciences*, 686:121345, 2025.

[Wu *et al.*, 2020] Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and S Yu Philip. A comprehensive survey on graph neural networks. *IEEE transactions on neural networks and learning systems*, 32(1):4–24, 2020.

[Wu *et al.*, 2022] Shiwen Wu, Fei Sun, Wentao Zhang, Xu Xie, and Bin Cui. Graph neural networks in recommender systems: a survey. *ACM Computing Surveys*, 55(5):1–37, 2022.

[Wu *et al.*, 2023] Zhengfeng Wu, Isabel Song, and Ioannis Savidis. Hybrid utilization of subgraph isomorphism and relational graph convolutional networks for analog functional grouping annotation. In *2023 ACM/IEEE 5th Workshop on Machine Learning for CAD (MLCAD)*, pages 1–6. IEEE, 2023.

[Zhang and Saber, 2024] Zixian Zhang and Takfarinas Saber. Assessing the code clone detection capability of large language models. In *2024 4th International Conference on Code Quality (ICCQ)*, pages 75–83. IEEE, 2024.

[Zhang *et al.*, 2021] Zhen Zhang, Jiajun Bu, Martin Ester, Zhao Li, Chengwei Yao, Zhi Yu, and Can Wang. H2mn: Graph similarity learning with hierarchical hypergraph matching networks. In *Proceedings of the 27th ACM SIGKDD conference on knowledge discovery & data mining*, pages 2274–2284, 2021.

[Zou *et al.*, 2025] Cuifang Zou, Guangquan Lu, Longqing Du, Xuxia Zeng, and Shilong Lin. Graph similarity learning for cross-level interactions. *Information Processing & Management*, 62(1):103932, 2025.