scientific reports

OPEN

Fusing multiplex heterogeneous networks using graph attention-aware fusion networks

Ziynet Nesibe Kesimoglu[1,3] & Serdar Bozdag[1,2,3]✉

Graph Neural Networks (GNN) emerged as a deep learning framework to generate node and graph embeddings for downstream machine learning tasks. Popular GNN-based architectures operate on networks of single node and edge type. However, a large number of real-world networks include multiple types of nodes and edges. Enabling these architectures to work on networks with multiple node and edge types brings additional challenges due to the heterogeneity of the networks and the multiplicity of the existing associations. In this study, we present a framework, named GRAF (Graph Attention-aware Fusion Networks), to convert multiplex heterogeneous networks to homogeneous networks to make them more suitable for graph representation learning. Using attention-based neighborhood aggregation, GRAF learns the importance of each neighbor per node (called *node-level attention*) followed by the importance of each network layer (called *network layer-level attention*). Then, GRAF processes a network fusion step weighing each edge according to the learned attentions. After an edge elimination step based on edge weights, GRAF utilizes Graph Convolutional Networks (GCN) on the fused network and incorporates node features on graph-structured data for a node classification or a similar downstream task. To demonstrate GRAF's generalizability, we applied it to four datasets from different domains and observed that GRAF outperformed or was on par with the baselines and state-of-the-art (SOTA) methods. We were able to interpret GRAF's findings utilizing the attention weights. Source code for GRAF is publicly available at https://github.com/bozdaglab/GRAF.

Keywords Attention aware network fusion, Graph neural networks, Drug ADR prediction

Graphs naturally represent complex relationships in multimodal datasets including biological and biomedical datasets. For instance, multi-omics datasets can be represented as gene-gene similarity networks and drug- and protein-based datasets can be represented as drug-target networks.

To train machine learning (ML) models on graph-structured data, several shallow (e.g., DeepWalk[1], node2vec[2], NECo[3]) and deep learning methods such as Graph Neural Networks (GNN)[4,5] have emerged. GNN utilizes deep neural networks on graph-structured data to learn node embeddings that capture both the graph topology and the features of node, edge, and/or graph[4–7]. Every node iteratively updates its current embedding by aggregating information from its local neighborhood. Graph Convolutional Networks (GCN) is one of the most popular GNN methods[6], which treat all neighboring nodes with equal importance during information aggregation. Inspired from[8], attention mechanisms are applied to graph-structured data[7], where information aggregation from neighborhood is based on the importance of neighboring nodes in a given network.

Most GNN-based architectures are primarily designed for homogeneous networks-those composed of a single type of node and edge. However, real-world networks often exhibit multiplex (i.e., having multiple types of edges) and heterogeneous (i.e., having multiple types of nodes) characteristics. For example, nodes in a network could represent papers, authors, and venues, with edges denoting relationships such as authorship and publication. We refer to each layer of the multiplex network as a *network layer*, which corresponds to each subnetwork within the multiplex network that contains edges of a distinct type. A heterogeneous network can be converted into a multiplex homogeneous network (i.e., multiple edge types and single node type) using meta-paths. In general, a meta-path is a path in a graph that visits different types of nodes via different types of edges. To build multiplex homogeneous networks, a meta-path starts and ends at the same node type and visits specific edge types in a given order to measure the similarity between the start and end nodes. Two meta-paths of equal length that follow the same node and edge types belong to the same meta-path type. For instance, in a heterogeneous network with node types author, paper, and venue, a meta-path author-paper-author defines the

[1]Department of Computer Science and Engineering, University of North Texas, Denton, TX, USA. [2]Department of Mathematics, University of North Texas, Denton, TX, USA. [3]BioDiscovery Institute, University of North Texas, Denton, TX, USA. ✉email: Serdar.Bozdag@unt.edu

similarity between two authors based on co-authorship, whereas a meta-path author-paper-venue-paper-author defines the similarity between two authors who publish at the same venue.

To perform graph representation learning on a multiplex network, GNN could be applied separately to each network layer. For instance, MOGONET[9] constructs a multiplex patient similarity network where each network layer was based on a distinct omics type. MOGONET applies a separate GCN on each network layer and integrates label distributions from each to determine final node labels. Similarly, SUPREME[10] learns node embeddings from each omic-based network layer in a multiplex patient similarity network using GCN. Then, it trains an ML model for each embedding combination to predict patient diagnosis. However, this operation could be computationally expensive when there are many omics types. In addition, these models typically overlook attention of nodes and edges, highlighting the need for more efficient and advanced methodologies in multiplex network analysis.

To address these limitations, in this study, we introduce GRAF (Graph Attention-aware Fusion Networks), a computational framework designed to transform multiplex heterogeneous networks to homogeneous networks for effective graph representation learning. GRAF utilizes node- and network layer-level attention as in[11] during the fusion process of these networks. Once fused, GRAF employs GCN to perform a node classification or a similar downstream task, incorporating node features.

We applied GRAF to four networks (including three heterogeneous networks and one multiplex network) spanning various domains to perform node classification. Our results show that GRAF outperformed most state-of-the-art (SOTA) and baseline methods across all datasets. Utilizing attention weights, GRAF provides interpretable results, highlighting the significance of nodes and network layers crucial for the prediction task.

The contributions of our work are summarized as follows:

- We developed GRAF, a framework to convert multiplex heterogeneous networks to homogeneous networks with an attention-aware network fusion strategy. GRAF runs GCN on the fused network for the desired node classification or a similar downstream task.
- GRAF provides attention values for each node and network layer, enabling the identification of critical network components for downstream tasks.
- We applied GRAF to four different networks-three heterogeneous and one multiplex-across four node classification problems from various domains, showing its robustness and generalizability.
- We conducted extensive evaluations to measure the performance of GRAF including an ablation study to assess the effectiveness of GRAF's components and their contributions to overall performance.

Related work

GNN-based methods

GNN attracted high interest as a deep learning framework to learn node, subgraph, and graph embeddings. Several GNN-based architectures have been developed with different approaches in message aggregation[6,7,12,13]. GCN uses self edges in the neighborhood aggregation and normalizes across neighbors with equal importance[6]. On the cancer type prediction problem, in[14], the authors leveraged GCN on a single biological network with one data modality, thus limiting the utilization of multiple data and networks. In[15], the authors proposed a hybrid model leveraging graph convolution and relation network on the breast cancer classification task, while in[16], the authors used a GCN-based model on drug and protein interaction network for multirelational link prediction. While most GNN-based models ignore edge directionality, Dir-GNN[17] extends GNN to preserve edge directionality, showing improved performance over conventional GNN-based models.

Generalizing the self-attention mechanisms of transformers[8], Graph Attention Networks (GAT) has been developed using attention-based neighborhood aggregation learning the importance of each neighbor[7]. A follow-up study has shown that GAT computes static attention, maintaining consistent rankings for attention coefficients within the same graph. They proposed GATv2[18] by changing the order of operations, and improved the expressiveness of GAT. SuperGAT[19] improves upon standard GAT by introducing a self-supervised approach that enhances attention robustness in noisy graphs by encoding edge presence and absence.

GNN-based methods on multiplex and heterogeneous networks

To utilize more knowledge, studies utilized GNN-based architectures to operate on multiplex network[9,10]. MOGONET[9] runs three different GCN models, each operating on a distinct patient similarity network constructed using a distinct data modality. Then, it uses the label distribution from three different models and utilizes them to predict the final label of each node. SUPREME[10] is a GCN-based node classification framework that operates on each layer of a multiplex network individually, encoding features from all data modalities within each network. In contrast to MOGONET, SUPREME utilizes intermediate embeddings and integrates them with node features, resulting in a consistent and improved performance. Also, SUPREME integrates embeddings by evaluating all network combinations to identify the best model.

In the realm of heterogeneous networks, Heterogeneous Graph Attention Network (HAN)[11] introduces a GNN-based architecture on a heterogeneous network, incorporating attention mechanisms. HAN first generates meta-path-based networks from a heterogeneous network and applies individual transformation matrices (i.e., matrices used to linearly transform node features) to nodes of different types. It then learns node-level attention within each node's meta-path-based neighborhood and network layer-level attention across meta-paths to improve model expressiveness . Similarly, Heterogeneous Graph Transformer (HGT)[20] handles graph heterogeneity by characterizing the heterogeneous attention over each edge. In addition, PreGAT[21] introduces predicate-aware graph attention networks to integrate relational information and enhance node differentiation, resulting in enriched embeddings that improve downstream node importance models.

Network fusion methods

Since multiplex networks may contain complementary information, some studies integrated these networks into a single network[22,23]. For instance, Similarity Network Fusion (SNF)[22] builds a patient similarity network based on each data modality, fuses all networks into one consensus network by applying a nonlinear step, and performs clustering on this consensus network. Affinity Network Fusion (ANF)[23] builds on SNF by simplifying the required computational operations. Network fusion methods show good performance without using probabilistic modeling, however, they heavily rely on constructing a similarity network to integrate information from multiple data modalities. In addition, these tools cannot utilize node features within the network, which could be potentially informative.

Materials and methods

GRAF

GRAF is a computational framework that transforms heterogeneous and/or multiplex networks into a homogeneous network using attention mechanisms and network fusion simultaneously (Fig. 1). Briefly, the first step of GRAF is to generate a meta-path-based multiplex network if the initial network is a heterogeneous network. In the second step, GRAF computes node- and network layer-level attention. In the third step, GRAF fuses multiple networks into a single weighted network using node- and network layer-level attention weights. Following this, GRAF removes edges from the fused network based on their strength. Finally, GRAF learns node embeddings using GCN and performs downstream ML tasks. The detailed explanation of each step in GRAF is as follows.

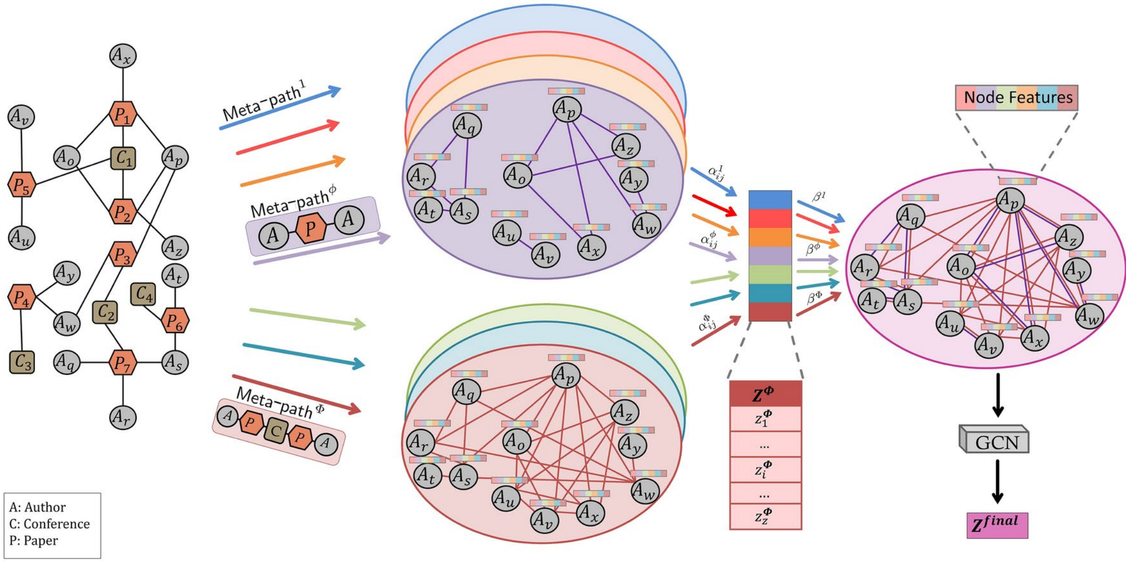

Fig. 1. The GRAF pipeline on a heterogeneous network. Initially, GRAF generates meta-path-based neighborhood. Then, it obtains node- and network layer-level attention. Using these attentions, GRAF fuses multiple network layers into a single weighted network. GRAF subsequently removes low-weighted edges and learns node embeddings through graph convolutions applied to the fused network.

Multiplex network generation

Networks generated based on meta-paths are referred to as meta-path-based networks. If the input network is a heterogeneous network (IMDB, ACM, and DBLP data for our case), GRAF converts this network into a multiplex network using meta-paths that start and end with the node types relevant to the downstream task. If the input network is already a multiplex network (DrugADR data for our case), GRAF skips this transformation. Below, we provide a detailed explanation of the conversion from heterogeneous to multiplex networks.

Let's assume we have a heterogeneous network G_H. We denote the nodes in G_H as $\mathrm{V} = \{v_1, v_2, \ldots, v_n\}$, where *n* is the total number of nodes. Each meta-path-based network is represented by a set of edges, including self-edges, denoted as E^ϕ. For every node pair $(v_i, v_j) \in G_H$, if there is a path between them based on the meta-path ϕ, then we add an edge to the edge set E^ϕ, that is $(v_i, v_j) \in \mathrm{E}^\phi$, where $\phi \in \{1, 2 \ldots \Phi\}$ and Φ is the total number of meta-path types. This edge can be formalized using an indicator function *I*:

$$I_{\mathrm{E}^\phi}(v_i, v_j) = \begin{cases} 1 & \text{if } (v_i, v_j) \in \mathrm{E}^\phi \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

After constructing all E^ϕ in G_H, we obtain a graph $\mathrm{G}^\phi = (\mathrm{V}, \mathrm{E}^\phi)$. All datasets have undirected graphs, $(v_i, v_j) \in \mathrm{E}^\phi \iff (v_j, v_i) \in \mathrm{E}^\phi$. In that way, we obtained a multiplex network from a heterogeneous network with a separate network layer ϕ for each meta-path type.

The neighborhood N_i^ϕ of node v_i is defined as $\mathrm{N}_i^\phi = \left\{v_j : (v_i, v_j) \in \mathrm{E}^\phi\right\}$, representing nodes associated with v_i according to meta-path ϕ. Additionally, a feature matrix $\mathrm{X} \in \mathrm{R}^{nxf}$ is generated, where $x_i \in \mathrm{R}^f$ represents the original node features of v_i, and *f* is the input feature size. X serves as input for the attention model and the final GCN model.

Computing node- and network layer-level attention

GRAF computes node-level attention α_{ij}^ϕ to learn the importance of each neighbor v_j relative to node v_i based on network layer ϕ. In addition, GRAF learns the network layer-level attention β^ϕ, which indicates the importance of the network layer ϕ to the prediction task. GRAF extracts node- and network layer-level attention values using the end-to-end HAN architecture[11] (see Supplementary Methods 1.1 for details). Alternatively, these attention values could be obtained through different approaches.

Attention-aware network fusion

Node pairs may have edges in multiple network layers. For each node pair, their attention (i.e., influence) to each other can vary from network layer to network layer. Furthermore, some network layers could be more influential than others. Therefore, when fusing multiple network layers, we ought to consider both node- and network layer-level attention weights.

Incorporating attention weights at both levels, we computed the edge weight from v_i to v_j (denoted as $score_{(v_i,v_j)}$) using a weighted sum of existing edges, defined as follows:

$$score_{(v_i,v_j)} = \sum_{\phi \in \{1,2\ldots\Phi\}} \left(\beta^\phi \alpha_{ij}^\phi I_{\mathrm{E}^\phi}(v_i, v_j)\right) \tag{2}$$

Intuitively, edges with higher node- or network layer-level attention receive greater weight. Thus, we considered the importance of node neighbors and their respective network layers. This edge scoring approach ensures a proper prioritization of all edges. These scores were utilized to construct a weighted network for the prediction task.

The overall attention-aware network fusion strategy is shown in Algorithm 1. Bias vectors prior to non-linearity are omitted for simplicity.

```
Input: Graph $\mathsf{G} = (\mathsf{V}, \mathsf{E}^{\phi})$ where $\mathsf{V}$ is a set of $n$ nodes, i.e., $\mathsf{V} = \{v_1, v_2, ..., v_n\}$, and $\mathsf{E}^{\phi}$ is a set of edges between nodes
       based on network layer $\phi$ where $\phi \in \{1, 2...\Phi\}$ and $\Phi$ is the total number of network layer types.
       $(v_i, v_j) \in \mathsf{E}^{\phi} \iff (v_j, v_i) \in \mathsf{E}^{\phi}$ (undirected graph)
       feature matrix $\mathsf{X} \in \mathsf{R}^{nxf}$
       C: number of repeats
Output: Adjacency matrix $\mathsf{A}$
for $c \in \{1, 2...C\}$ do
    for $\phi \in \{1, 2...\Phi\}$ do
        Node type-specific transformation: $h_i = M_{\Theta}.x_i$
        for $v_i \in \mathsf{V}$ do
            for $v_j \in \mathsf{N}_i^{\phi} = \{v_j : (v_i, v_j) \in \mathsf{E}^{\phi}\}$ do
                $e_{ij}^{\phi} = \mathrm{LeakyReLU}\left((a^{\phi})^{\top}.[h_i||h_j]\right)$
                Weight coefficient $\alpha_{ij}^{\phi}$ =softmax$_j(e_{ij}^{\phi}) = \frac{\exp(e_{ij}^{\phi})}{\sum_{k \in \mathsf{N}_i^{\phi}} \exp\left(e_{ik}^{\phi}\right)}$
        Network layer-specific embedding $z_i^{\phi} = \sigma\left(\sum_{v_j \in \mathsf{N}_i^{\phi}} \alpha_{ij}^{\phi}.h_j\right)$
        Network layer-specific combined embedding $\mathsf{Z}^{\phi}$
        $f^{\phi} = \frac{1}{|\mathsf{V}|} \cdot \sum_{v_i \in \mathsf{V}} q^{\top}.\tanh(M_0.z_i^{\phi})$
    Network layer weight coefficient $\beta^{\phi}$ =softmax$_{\phi}(f^{\phi}) = \frac{\exp(f^{\phi})}{\exp\left(\sum_{i \in \Phi} f^i\right)}$
    Final embedding $\mathsf{Z} = \sigma\left(\sum_{i \in \Phi} \beta^i.\mathsf{Z}^i\right)$
    Cross-entropy loss
    Backpropagation and final parameter obtention
    $\beta_c^{\phi} = \beta^{\phi}$
    $\alpha_{ijc}^{\phi} = \alpha_{ij}^{\phi}$
$\beta^{\phi} = \frac{1}{C} \sum_{c \in \{1,2...C\}} \beta_c^{\phi}$
$\alpha_{ij}^{\phi} = \frac{1}{C} \sum_{c \in \{1,2...C\}} \alpha_{ijc}^{\phi}$
$\mathsf{A}[i, j] = \sum_{\phi \in \{1,2...\Phi\}} \left(\beta^{\phi} \alpha_{ij}^{\phi} I_{\mathsf{E}^{\phi}}(v_i, v_j)\right)$
```

Algorithm 1. Attention-aware network fusion.

Edge elimination

The fused network keeps all the edges from multiple network layers regardless of their weight. Depending on the input network layer quality, this may result in a densely connected network with many weak edges. To address this, we included an edge elimination step, where we eliminated some portion of the edges.

We used edge weights as probabilities to keep each edge in the network. We preserved a specified percentage, x%, of edges by randomly eliminating them based on a probability distribution that is proportional to their weights. Here, x is a hyperparameter. This approach intuitively removes edges with low attention or those from less important network layers from the fused network . Now, the fused network is ready to be utilized in GCN model for downstream tasks.

Node classification task

To train the fused network for downstream tasks utilizing node features and network topology, GRAF generates node embeddings using a 2-layer GCN[6]. This step can be optimized for various downstream tasks such as subgraph classification or link prediction.

For a GCN model operating on a single network with edge set E, the adjacency matrix $\mathrm{A} \in \mathrm{R}^{nxn}$ is defined as:

$$\mathrm{A}[i,j] = \begin{cases} score_{(v_i,v_j)} & \text{if } (v_i,v_j) \in \mathrm{E} \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

The iteration process of the model is: $\mathrm{H}^{(l+1)} = \sigma\left(\mathrm{D}^{-\frac{1}{2}}\mathrm{A}\mathrm{D}^{-\frac{1}{2}}\mathrm{H}^{(l)}\mathrm{W}^{(l)}\right)$ with $\mathrm{H}^{(0)} = \mathrm{X}$ where

$$\mathrm{D}[i,i] = \sum_{j=1}^{n} \mathrm{A}[i,j], \tag{4}$$

$\mathrm{X} \in \mathrm{R}^{nxf}$ is the feature matrix, and $\mathrm{H}^{(l)}$ and $\mathrm{W}^{(l)}$ are activation matrix and trainable weight matrix of l^{th} layer, respectively. Feature aggregation on the local neighborhood of each node is done by multiplying X by *nxn*-sized scaled adjacency matrix $\mathrm{A}^{'}$ where $\mathrm{A}' = \mathrm{D}^{-\frac{1}{2}}\mathrm{A}\mathrm{D}^{-\frac{1}{2}}$.

Using a 2-layer GCN model, we had the forward model giving the output $\mathrm{Z}_{\mathrm{final}} \in \mathrm{R}^{nxc}$ where

$$\mathrm{Z}_{\mathrm{final}} = \mathrm{softmax}\left(\mathrm{A}'\mathrm{ReLU}\left(\mathrm{A}'\mathrm{X}\mathrm{W}^{(1)}\right)\mathrm{W}^{(2)}\right) \tag{5}$$

with $\mathrm{W}^{(1)} \in \mathrm{R}^{fxf'}$, $\mathrm{W}^{(2)} \in \mathrm{R}^{f'xc}$ and c is the number of class labels. Cross-entropy was used as the loss function.

See Supplementary Methods 1.1, 1.2, and 1.3 for methodology details.

Experiments

We applied our tool to four prediction tasks: movie genre prediction using IMDB data (https://www.imdb.com), paper type prediction using ACM data (http://dl.acm.org), author research area prediction using DBLP data (https://dblp.uni-trier.de/), and adverse drug reaction (ADR) prediction using ADReCS[24].

IMDB: For movie genre prediction task, we collected and processed IMDB data using PyTorch Geometric library[25]. The dataset is represented as a heterogeneous network with three node types: movie (M), actor (R), and director (D); and two edge types: movie-actor and movie-director. We converted the heterogeneous network into a multiplex network for the movie node type using two meta-paths: MRM and MDM. Movie node features, extracted as elements of a bag-of-words, are obtained from the library's data processing. We predicted the genre of the movies in this multiplex network. Movie nodes had three class labels: action, comedy, and drama.

ACM: For paper type prediction task, we collected ACM data using Deep Graph Library[26]. The dataset is represented as a heterogeneous network with three node types: paper (P), author (A), and subject (S); and two edge types: paper-author and paper-subject. We converted the heterogeneous network into a multiplex network for the paper node type using two meta-paths: PAP and PSP. Paper node features are the elements of a bag-of-words representation, obtained from the library. We predicted the area of the papers in this multiplex network. Paper nodes had three class labels: database, wireless communication, and data mining.

DBLP: For author research area prediction task, we collected DBLP data from[27] and preprocessed data using[28]. The dataset is represented as a heterogeneous network with four node types: paper (P), author (A), conference (C), and term (T); and three edge types: paper-author, paper-conference, and paper-term. We converted the heterogeneous network into a multiplex network for the author node type using four meta-paths: APA, APAPA, APCPA, and APTPA. Author features are from the preprocessed data in[11]. We predicted the research area of the authors in this multiplex network. Author nodes had four class labels: database, data mining, artificial intelligence, and information retrieval.

DrugADR: For ADR prediction task, we collected drug-ADR pairs from ADReCS[24]. We obtained multiplex drug similarity network with four network layers from[29]. We generated SMILES fingerprints as drug node features (see Supplementary Methods 1.4 for details). We predicted the ADR of the drugs in this multiplex network. Drug nodes had five most frequent ADRs as class labels: dizziness, hypersensitivity, pyrexia, rash, and vomiting.

A detailed description of each dataset is shown in Table 1.

SOTA and baseline methods

Here, we list SOTA and baseline methods compared with GRAF. Here, all networks are converted to multiplex network using the same procedure (see "Multiplex network generation" section in "Materials and methods"):

GCN[6]: Since GCN cannot operate on multiplex networks, we ran GCN on each network layer and reported the best performance.

GAT[7] **and GATv2**[18]: GAT and GATv2 involve attention mechanism designed for homogeneous networks, precluding their direct application to multiplex networks. Therefore, we ran them individually on each network layer and reported the best performance.

Baseline methods: We evaluated Multi-layer Perceptron (MLP), Random Forest (RF), and Support Vector Machine (SVM), which use only node features, without utilizing graph-structured data.

Dir-GNN[17]: Dir-GNN extends GNN to preserve edge directionality. We ran it on each network layer and reported the best performance.

SuperGAT[19]: SuperGAT improves upon graph attention models to enhance attention robustness in noisy graphs by encoding edge presence and absence. We ran this method on each network layer and reported the best performance.

HGT[20]: HGT works on heterogeneous graphs using heterogeneous attention mechanisms.

Dataset	# Nodes	# Features	# Classes	Network layer type*	# Edges
IMDB	4278	3066	3	MRM	85,358
				MDM	17,446
ACM	3025	1870	3	PAP	29,281
				PSP	2,210,761
DBLP	4057	334	4	APA	11,113
				APAPA	40,703
				APCPA	5,000,495
				APTPA	7,043,627
DrugADR	664	1024	5	G-G_1	8158
				G-G_2	10,518
				G-G_3	7328
				G-G_4	3512

Table 1. Datasets used in the study. [*A: Author, C: Conference, D: Director, M: Movie, P: Paper, R: Actor, S: Subject, T: Term. G-G_x denotes drug-drug similarity networks based on four similarities: drug ATC (Anatomical Therapeutic Chemical) code-based similarity, drug interactions-based similarity, chemical structures-based molecular fingerprints similarity, and drug side effects-based similarity. IMDB, ACM, and DBLP networks were converted from heterogeneous network to multiplex network using meta-paths. See text for details.].

Method	IMDB	ACM	DBLP	DrugADR
GCN	58.7 ± 0	91.5 ± 0	90.5 ± 0	*32.9 ± 0*
GAT	56.8 ± 0	91.0 ± 0	91.4 ± 1	31.6 ± 2
GATv2	56.8 ± 1	90.9 ± 1	90.0 ± 1	31.2 ± 2
MLP	55.0 ± 1	89.0 ± 1	78.4 ± 1	22.0 ± 4
RF	53.4 ± 0	88.9 ± 0	69.3 ± 0	28.8 ± 1
SVM	55.1 ± 0	88.5 ± 0	76.5 ± 0	24.8 ± 0
Dir-GNN	53.7 ± 1	84.1 ± 1	90.5 ± 1	25.8 ± 2
SuperGAT	55.8 ± 1	84.5 ± 0	90.2 ± 1	28.9 ± 4
HGT	56.5 ± 1	84.0 ± 2	86.4 ± 2	29.2 ± 2
HAN	*60.9 ± 0*	92.0 ± 1	91.5 ± 1	30.2 ± 0
$\mathrm{SUPREME}_{min}$	53.7 ± 2	90.7 ± 0	77.9 ± 2	31.3 ± 5
$\mathrm{SUPREME}_{med}$	57.0 ± 2	92.4 ± 1	90.8 ± 1	31.4 ± 4
$\mathrm{SUPREME}_{max}$	60.8 ± 3	**93.4 ± 1**	**92.3 ± 2**	32.1 ± 3
GRAF	**62.1 ± 0**	*92.6 ± 0*	*91.7 ± 1*	**34.7 ± 2**

Table 2. Node classification performance evaluated through macro F1 scores (%) across four distinct tasks: movie genre prediction from IMDB data, paper type prediction task from ACM data, author research area prediction task from DBLP data, and ADR (adverse drug reaction) prediction task. Results highlight the best score in bold and the second-best in italic. $\mathrm{SUPREME}_{min}$, $\mathrm{SUPREME}_{med}$, and $\mathrm{SUPREME}_{max}$ represents the models achieving the minimum, median, and best model based on validation macro F1 scores among all network combinations, respectively. GCN, GAT, GATv2, Dir-GNN, and SuperGAT were evaluated for every single network, and the best performance was reported. [GAT: Graph Attention Network, GCN: Graph Convolutional Network, MLP: Multi-layer Perceptron, RF: Random Forest, SVM: Support Vector Machine].

HAN[11]: HAN integrates multiplex networks utilizing attention mechanisms.

SUPREME[10]: SUPREME learns node embeddings from multiple networks using GCN and trains separate models for each network layer combination to find the best performance. To ensure a fair comparison, we reported the minimum ($\mathrm{SUPREME}_{min}$), median ($\mathrm{SUPREME}_{med}$), and maximum ($\mathrm{SUPREME}_{max}$) scores based on validation macro F1 across all combinations.

Results

We evaluated GRAF and the other tools, reporting their performance based on three metrics: macro F1 score, weighted F1 score, and accuracy (with median scores across 10 repeats).

Comparison with SOTA/baseline: According to our results, GRAF achieved the best performance or was on par with the other tools across all metrics and datasets (Tables 2 and S1). GRAF consistently outperformed GCN, GAT, GATv2, Dir-GNN, and SuperGAT in macro F1 score across all datasets, highlighting the efficacy of utilizing multiple networks. While GRAF generally performed better than the median SUPREME results, $\mathrm{SUPREME}_{max}$ (i.e., SUPREME model with the best performing network layer combination) showed slightly better performance than GRAF on ACM and DBLP data. However, as the number of network layers increases, SUPREME's computational cost rises notably, making it impractical to evaluate all possible combinations. Consequently, selecting the optimal SUPREME model becomes challenging, and subsetting the network layer combinations may be necessary. Conversely, GRAF demonstrated substantial superiority over all SUPREME models in IMDB and DrugADR datasets. GRAF also outperformed both HGT and HAN in all prediction tasks related to handling graph heterogeneity. This improved performance over HAN indicates that our attention-aware network fusion strategy enhances the utilization of multiple graph-structured data further.

We also observed that GRAF, HAN, HGT, GCN, GAT, GATv2, Dir-GNN, RF, and SVM exhibited more consistent performance with small standard deviations, while other tools had higher standard deviations, which was particularly notable in DrugADR dataset. MLP, RF, and SVM exhibited the lowest performance, showing the importance of graph-structured data utilization. Overall, integrative approaches (i.e., SUPREME, GRAF, and HAN) had better performance.

Ablation studies: To assess the importance of various components within the GRAF architecture, we generated three variants: GRAF_{net_lay} considers only network layer-level attention in edge scoring (thus excluding node-level attention). Therefore the score function is replaced with:

$$score_{(v_i,v_j)} = \sum_{\phi \in \{1,2\ldots\Phi\}} \left(\beta^{\phi} I_{\mathrm{E}^{\phi}}(v_i, v_j)\right) \tag{6}$$

Thus, the same importance is assigned to every edge within the same network layer. GRAF_{node} considers only node-level attention in edge scoring (excluding network layer-level attention). That is, it assigns equal importance to each network layer type by replacing the score function with:

$$score_{(v_i,v_j)} = \sum_{\phi \in \{1,2\ldots\Phi\}} \left(\alpha_{ij}^{\phi} I_{\mathrm{E}^{\phi}}(v_i, v_j)\right) \tag{7}$$

GRAF_{edge} includes both attentions without eliminating edges (i.e., keeps all fused edges).

We observed that both node- and network layer-level attentions are crucial for GRAF's performance (see Table 3). Using only network layer-level attention, GRAF_{net_lay} exhibited lower performance across all datasets, which is not surprising as all edges within the same network layer were assigned equal importance. On the other hand, using only node-level attention, GRAF_{node} had lower performance than GRAF overall, yet outperformed GRAF_{net_lay}. GRAF_{node} assigned equal importance to each network layer, but the inclusion of node-level attention preserved substantial amount of knowledge. GRAF_{edge} demonstrated comparable performance to GRAF.

To check GRAF's performance across various data splits, we generated four additional split sets using IMDB data (Supplementary Methods 1.5). In all split sets, GRAF consistently achieved superior performance compared to other methods (Figs. 2, S1, and S2). We also observed that most methods showed a tendency to increase their performance with higher training sample size, which aligns with expectations.

To assess the impact of percentage of eliminated edges on the fused network, we compared performance across all datasets (Figs. S3, S4, and S5). In all cases, including relatively easier tasks such as those on ACM and DBLP data, as well as more complex tasks on IMDB and DrugADR data, we found no notable differences, even when comparing scenarios of keeping only 10% of edges versus no elimination. Specifically, in the IMDB dataset, hyperparameter tuning led to no edge elimination, yielding identical results for GRAF and GRAF_{edge}. GRAF models trained on other datasets utilized edge elimination (specifically keeping 70%, 70%, and %30 of the edges for ACM, DBLP, and DrugADR data, respectively).

Interpretation of results: GRAF enables interpretation of prediction results using node-level attention, network layer-level attention, and also fused edges combining both attentions. We reported network layer-level attention to determine the general usefulness of each network layer (see Supplementary Table S2). Our integrative analysis enhances understanding of drug characteristics across different similarity network layers. It emphasizes the drug side effects-based similarity network as particularly crucial, followed by the chemical

Method	IMDB	ACM	DBLP	DrugADR
GRAF_{net_lay}	56.3 ± 0	84.6 ± 2	89.7 ± 0	28.8 ± 2
GRAF_{node}	61.3 ± 0	90.9 ± 2	90.3 ± 1	31.8 ± 2
GRAF_{edge}	**62.1 ± 0**	92.3 ± 0	91.3 ± 1	33.9 ± 2
GRAF	**62.1 ± 0**	**92.6 ± 0**	**91.7 ± 1**	**34.7 ± 2**

Table 3. Ablation studies evaluated through macro F1 scores (%) across four distinct tasks: movie genre prediction from IMDB data, paper type prediction task from ACM data, author research area prediction task from DBLP data, and ADR (adverse drug reaction) prediction task. Results highlight the best score in bold. Models include GRAF_{net_lay} (with only network layer-level attention), GRAF_{node} (with only node-level attention), and GRAF_{edge} (without edge elimination). .

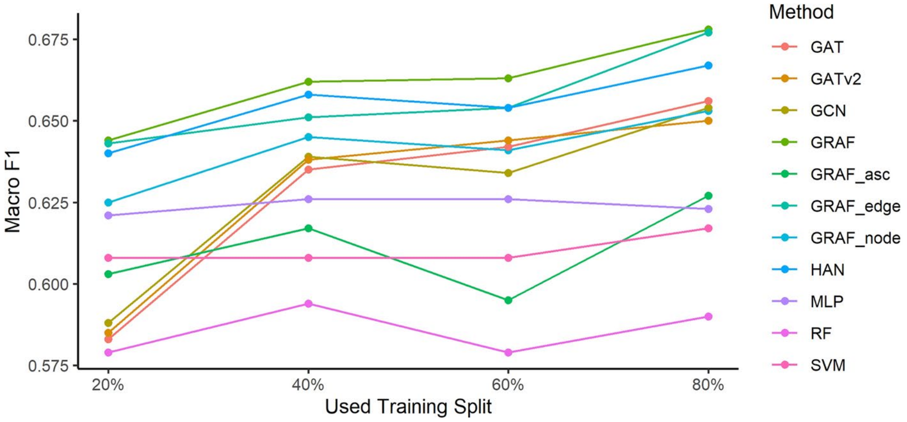

Fig. 2. Performance with different training splits on IMDB data (macro F1).

structures-based molecular fingerprints similarity network. For IMDB data, each network layer had similar attention, while ACM and DBLP data had one network layer with strong attention (> 0.6). Specifically, the network layer constructed using paper-author-paper meta-path had a higher attention value than the network layer constructed using paper-subject-paper meta-path in ACM dataset, while in DBLP dataset, the network layer constructed using author-paper-conference-paper-author meta-path was the best network layer. Across these datasets, GCN, GAT, and GATv2 achieved the highest performance using network layers with the highest attention values. This result was also consistent with HAN's findings[11].

We leveraged four distinct drug similarity network layers based on different criteria: ATC codes, drug interactions, chemical structures, and drug side effects. Our findings uncover notable patterns among highly active nodes within each network. Specifically, in the ATC code-based similarity network layer, the top five drugs, having the highest number of connections, predominantly belong to the vomiting class, with Cisplatin emerging as the most active drug. Cisplatin, a platinum-based chemotherapy agent, is widely used in the treatment of various cancers, including sarcomas, carcinomas, lymphomas, and germ cell tumors[30–32], albeit with associated risks such as ototoxicity in individuals with specific genotypes[33]. In drug interactions-based similarity network layer, Bupivacaine stands out as the most active drug, utilized extensively as a local anesthetic across diverse medical procedures[34]. Furthermore, Clomipramine and Pantoprazole emerge as pivotal drugs in chemical structures-based molecular fingerprints and drug side effects-based similarity network layers, respectively. Clomipramine, a tricyclic antidepressant, is indicated for treating conditions like obsessive-compulsive disorder, while Pantoprazole, a proton pump inhibitor, is prescribed for managing gastric acid-related disorders[35–37]. Both drugs show extensive reported drug interactions and ADRs, highlighting their clinical significance and challenges in therapeutic management.

Prior to fusing multiple networks, GRAF requires attention values, which we obtained using HAN[11]. HAN supports parallelization by computing attention across all nodes and meta-paths separately. The time complexity for node-level attention is $O(V_\phi F_1 F_2 K + E_\phi F_1 K)$ for a given meta-path ϕ, where K is the number of attention heads, V_ϕ is the number of nodes, E_ϕ is the number of meta-path-based edges, and F_1, F_2 are the dimensions (row and column) of the transformation matrix. HAN's overall complexity is linear to the number of nodes and edges. However, without parallelization, HAN may become computationally expensive, particularly with large networks or numerous networks to integrate. To address this limitation, node-level attention could be computed more efficiently using approaches like GAT. Furthermore, for network layer-level attention, graph sampling can be utilized to reduce computing cost.

Conclusion

In this study, we developed a computational framework to convert multiplex heterogeneous networks to homogeneous networks based on node- and network layer-level attention. Our extensive experiments on four different datasets showed that GRAF outperformed most methods in all tasks and it is a generalizable tool. Attention values computed by GRAF also makes it an interpretable tool. The proposed GRAF showed improved performance or was on par with SOTA and baseline methods, as well as its variants.

Data availability

The details of dataset used are explained in the Experiments section and in the supplemental file.

Code availability

Source code is publicly available at https://github.com/bozdaglab/GRAF.

Received: 1 April 2024; Accepted: 31 October 2024
Published online: 24 November 2024

References

1. Perozzi, B., Al-Rfou, R. & Skiena, S. Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 701–710 (2014).
2. Grover, A. & Leskovec, J. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 855–864 (2016).
3. Dursun, C., Smith, J. R., Hayman, G. T., Kwitek, A. E. & Bozdag, S. Neco: A node embedding algorithm for multiplex heterogeneous networks. In *2020 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, 146–149 (IEEE, 2020).
4. Gori, M., Monfardini, G. & Scarselli, F. A new model for learning in graph domains. In *Proceedings. 2005 IEEE International Joint Conference on Neural Networks*, vol. 2, 729–734 (2005).
5. Scarselli, F., Gori, M., Tsoi, A. C., Hagenbuchner, M. & Monfardini, G. The graph neural network model. *IEEE Trans. Neural Netw.* **20**, 61–80 (2008).
6. Kipf, T. N. & Welling, M. Semi-supervised classification with graph convolutional networks. arXiv preprint arXiv:1609.02907 (2016).
7. Veličković, P. et al. Graph attention networks. arXiv preprint arXiv:1710.10903 (2017).
8. Vaswani, A. et al. Attention is all you need. *Adv. Neural Inf. Process. Syst.* **30** (2017).
9. Wang, T. et al. Mogonet integrates multi-omics data using graph convolutional networks allowing patient classification and biomarker identification. *Nat. Commun.* **12**, 1–13 (2021).
10. Kesimoglu, Z. N. & Bozdag, S. Supreme: multiomics data integration using graph convolutional networks. *NAR Genom. Bioinform.* **5**, lqad063 (2023).
11. Wang, X. et al. Heterogeneous graph attention network. In *The World Wide Web Conference*, 2022–2032 (2019).
12. Xu, K., Hu, W., Leskovec, J. & Jegelka, S. How powerful are graph neural networks? arXiv preprint arXiv:1810.00826 (2018).
13. Wu, F. et al. Simplifying graph convolutional networks. In *International Conference on Machine Learning*, 6861–6871 (PMLR, 2019).
14. Ramirez, R. et al. Classification of cancer types using graph convolutional neural networks. *Front. Phys.* **8**, 203 (2020).
15. Rhee, S., Seo, S. & Kim, S. Hybrid approach of relation network and localized graph convolutional filtering for breast cancer subtype classification. arXiv preprint arXiv:1711.05859 (2017).
16. Zitnik, M., Agrawal, M. & Leskovec, J. Modeling polypharmacy side effects with graph convolutional networks. *Bioinformatics* **34**, i457–i466 (2018).
17. Rossi, E. et al. Edge directionality improves learning on heterophilic graphs. In *Learning on Graphs Conference*, 25–1 (PMLR, 2024).
18. Brody, S., Alon, U. & Yahav, E. How attentive are graph attention networks? arXiv preprint arXiv:2105.14491 (2021).
19. Kim, D. & Oh, A. How to find your friendly neighborhood: Graph attention design with self-supervision. arXiv preprint arXiv:2204.04879 (2022).
20. Hu, Z., Dong, Y., Wang, K. & Sun, Y. Heterogeneous graph transformer. *In Proceedings of the Web Conference*, vol. 2020, 2704–2710 (2020).
21. Zhang, T. et al. Label informed contrastive pretraining for node importance estimation on knowledge graphs. In *IEEE Transactions on Neural Networks and Learning Systems* (2024).
22. Wang, B. et al. Similarity network fusion for aggregating data types on a genomic scale. *Nat. Methods* **11**, 333–337 (2014).
23. Ma, T. & Zhang, A. Integrate multi-omic data using affinity network fusion (anf) for cancer patient clustering. In *2017 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, 398–403 (IEEE, 2017).
24. Cai, M.-C. et al. Adrecs: an ontology database for aiding standardization and hierarchical classification of adverse drug reaction terms. *Nucleic Acids Res.* **43**, D907–D913 (2015).
25. Fey, M. & Lenssen, J. E. Fast graph representation learning with pytorch geometric. arXiv preprint arXiv:1903.02428 (2019).
26. Wang, M. et al. Deep graph library: A graph-centric, highly-performant package for graph neural networks. arXiv preprint arXiv:1909.01315 (2019).
27. Ji, M., Sun, Y., Danilevsky, M., Han, J. & Gao, J. Graph regularized transductive classification on heterogeneous information networks. In *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2010, Barcelona, Spain, September 20–24, 2010, Proceedings, Part I 21*, 570–586 (Springer, 2010).
28. Fu, X., Zhang, J., Meng, Z. & King, I. Magnn: Metapath aggregated graph neural network for heterogeneous graph embedding. In *Proceedings of the Web Conference*, vol. 2020, 2331–2341 (2020).
29. Olayan, R. S., Ashoor, H. & Bajic, V. B. Ddr: efficient computational method to predict drug-target interactions using graph mining and machine learning approaches. *Bioinformatics* **34**, 1164–1173 (2018).
30. Zamble, D. B. & Lippard, S. J. Cisplatin and DNA repair in cancer chemotherapy. *Trends Biochem. Sci.* **20**, 435–439 (1995).
31. Makovec, T. Cisplatin and beyond: molecular mechanisms of action and drug resistance development in cancer chemotherapy. *Radiol. Oncol.* **53**, 148–158 (2019).
32. Tang, C., Livingston, M. J., Safirstein, R. & Dong, Z. Cisplatin nephrotoxicity: new insights and therapeutic implications. *Nat. Rev. Nephrol.* **19**, 53–72 (2023).
33. Sakano, S. et al. Nucleotide excision repair gene polymorphisms may predict acute toxicity in patients treated with chemoradiotherapy for bladder cancer. *Pharmacogenomics* **11**, 1377–1387 (2010).
34. Paganelli, M. A. & Popescu, G. K. Actions of bupivacaine, a widely used local anesthetic, on nmda receptor responses. *J. Neurosci.* **35**, 831–842 (2015).
35. Thorén, P., Åsberg, M., Cronholm, B., Jörnestedt, L. & Träskman, L. Clomipramine treatment of obsessive-compulsive disorder: I. A controlled clinical trial. *Arch. Gen. Psychiatry* **37**, 1281–1285 (1980).
36. McTavish, D. & Benfield, P. Clomipramine: an overview of its pharmacological properties and a review of its therapeutic use in obsessive compulsive disorder and panic disorder. *Drugs* **39**, 136–153 (1990).
37. Poole, P. Pantoprazole. *Am. J. Health-Syst. Pharm.* **58**, 999–1008 (2001).

Acknowledgements

This work was supported by the National Institute of General Medical Sciences of the National Institutes of Health under Award Number R35GM133657 to SB.

Author contributions

All authors contributed to the study conceptualization, analysis, interpretation of results, and manuscript writing/revision. Material preparation and data collection were performed by ZNK. Supervision was performed by SB. All authors read and approved the final manuscript.

Declarations

Competing interests
The authors declare no competing interests.

Additional information
Supplementary Information The online version contains supplementary material available at https://doi.org/10.1038/s41598-024-78555-4.

Correspondence and requests for materials should be addressed to S.B.

Reprints and permissions information is available at www.nature.com/reprints.