# muxGNN: Multiplex Graph Neural Network for Heterogeneous Graphs

Joshua Melton and Siddharth Krishnan

***Abstract*—Graph neural networks (GNNs) have become effective learning techniques for many downstream network mining tasks including node and graph classification, link prediction, and network reconstruction. However, most GNN methods have been developed for homogeneous networks with only a single type of node and edge. In this work we present muxGNN, a multiplex graph neural network for heterogeneous graphs. To model heterogeneity, we represent graphs as multiplex networks consisting of a set of relation layer graphs and a coupling graph that links node instantiations across multiple relations. We parameterize relation-specific representations of nodes and design a novel coupling attention mechanism that models the importance of multi-relational contexts for different types of nodes and edges in heterogeneous graphs. We further develop two complementary coupling structures: node invariant coupling suitable for node- and graph-level tasks, and node equivariant coupling suitable for link-level tasks. Extensive experiments conducted on six real-world datasets for link prediction in both transductive and inductive contexts and graph classification demonstrate the superior performance of muxGNN over state-of-the-art heterogeneous GNNs. In addition, we show that muxGNN's coupling attention discovers interpretable connections between different relations in heterogeneous networks.**

***Index Terms*—Graph neural networks, graph representation learning, heterogeneous graphs, multiplex networks.**

## I. Introduction

HETEROGENEOUS graphs are commonly used abstractions for modeling complex systems, in which different types of entities interact with one another in a variety of ways. Over the past several years, researchers have increasingly explored techniques for mining heterogeneous data. Such heterogeneous graphs arise naturally in many domains, such as a protein-protein interaction network wherein a set of proteins interact within a number of different biological tissues, or in data provenance, where system operations are captured via a tool such as CamFlow [1], that uses relational paradigm to define different types of system entities and a set of different interaction between such entities. Classical paradigms such as metapath2vec employ random walks over defined meta-paths to model the structure of heterogeneous networks. Recently, graph neural networks (GNNs) have become increasingly successful in graph-based learning tasks including node classification, link prediction, graph classification, and network reconstruction [2], [3], [4], [5], [6]. As a result of their successes, there have been a number of attempts to adopt GNNs to learn over heterogeneous networks [7], [8], [9], [10], [11], [12]. However, these approaches face several issues in capturing the multi-faceted nature of nodes in heterogeneous networks. First, many such approaches require the design of specific meta-paths for a heterogeneous graph, which necessitates specific domain knowledge and engineering. Secondly, many approaches keep distinct non-sharing weights for either node type or edge type alone, which limits them from capturing the full heterogeneous properties of the network structure.

Another approach to the study of heterogeneous networks has been through the lens of multiplex systems. A multiplex is a multi-layer representation of a network that has been commonly used to model heterogeneous and temporal networks. In particular, multiplex analysis has often focused on the spectral properties of such complexes as a means to define centrality metrics, conduct temporal analyses, characterize intra- and inter-layer relationships, and perform spectral clustering [13], [14], [15], [16]. Despite fruitful developments in multiplex analysis, such approaches have not fully adopted the deep graph neural network approaches that now dominate the study of homogeneous networks and are increasing important for heterogeneous graph analysis. Conversely, most heterogeneous GNN approaches have tended toward the spatial interpretation of graph convolutional networks and avoided reference to the spectral properties of heterogeneous graphs, despite the origins of GNNs as approximating spectral convolutions over the graph Laplacian [3], [17], [18], [19].

In light of these limitation and challenges, we propose to study heterogeneous graph neural networks from the perspective of spectral convolutions on a multiplex system. In this work, we present the multiplex GNN (muxGNN) architecture. We introduce a multiplex construction of a heterogeneous network composed of a set of layer graphs and coupling graph. Such a construction has the goal of maintaining composite node- and edge-type dependent representations, avoiding customized meta paths, and maintaining efficiency by decomposing convolutions over a multiplex into component convolutions over the layer graphs and coupling graph. We show that such a decomposition of multiplex convolutions can be reached by exploiting the shared eigenbasis of the set of layer graphs, coupling graph, and the parent multiplex. We utilize the relationship between the

Manuscript received 3 November 2022; revised 18 March 2023; accepted 21 March 2023. Date of publication 29 March 2023; date of current version 4 August 2023. This work was sponsored by the Army Research Office and was accomplished under Grant W911NF-22-1-0035. Recommended for acceptance by J.M. Solomon. *(Corresponding author: Joshua Melton.)*

The authors are with the Department of Computer Science, University of North Carolina at Charlotte, Charlotte, NC 28223 USA (e-mail: jmelto30@uncc.edu; skrishnan@uncc.edu).

Digital Object Identifier 10.1109/TPAMI.2023.3263079

eigenvalues of the multiplex components to leverage existing graph neural network architectures for the layer graphs, and we introduce a novel coupling graph attention mechanism to capture graph heterogeneity. Instead of parameterizing each type of node or edge separately, we construct a multi-layer representation of a heterogeneous network by breaking down each edge $e = (u, v)$ on the basis of its relation triple, i.e. $\langle$node type of $u$, edge type of $e$, node type of $v\rangle$. We use this relational structure to define the set of relation-specific layer graphs and to parameterize weight matrices for calculating a node's local representation with respect to each relation in which it participates. We then define a novel coupling structure and attention mechanism over the separate node-relation instantiations of a node. Due to the nature of its architecture, muxGNN can incorporate information from higher order neighbors and across relations through message passing over multi $k$-hop neighborhoods of nodes, implicitly extracting "soft" meta-paths over the network. In addition, the proposed coupling attention can automatically and implicitly learn "meta-paths" that are relevant for individual downstream tasks.

We further develop two coupling graph structures that allow our attention mechanism to be applied in either a *node invariant* manner or in a *node equivariant* manner, providing additional flexibility to learn a composite node representation suitable for node-level or graph-level tasks or to learn a set of multi-embeddings encoding relational views of a node suitable for link-level tasks.

We demonstrate the effectiveness of the proposed muxGNN on link prediction tasks in both transductive and inductive setting on four datasets drawn from a variety of domains and on a graph classification task on two data provenance graph datasets. Experimental results show that muxGNN can significantly improve performance on downstream tasks over state-of-the-art heterogeneous GNNs on both tasks and in both transductive and inductive settings. We further conduct an ablation analysis and case study to show that the proposed coupling attention can indeed automatically capture important inter-layer relationships in a protein-protein interaction network that recover intuitive biological connections between different tissues.

## II. Related Work

Research over the last several years has seen the increasing success of graph neural networks in modeling relational data for many downstream graph learning tasks [2], [3], [6]. Broadly, GNNs are a generalization of convolutional operations to arbitrarily connected graph-structured data. Researchers have typically categorized GNNs following two distinct paradigms, spatial and spectral, but recent works in geometric deep learning have argued that such a divide is improper and that the spatial and spectral interpretations of GNNs are rather complementary interpretations describing the same underlying principles [19].

### A. Graph Neural Networks

Graph neural networks can generally be described as using the input graph structure as the computation graph defining a local diffusion process over neighborhoods in the network. Early GNN approaches developed approximations of spectral convolutions over graphs by using neural networks to parameterize functions over the eigenvalues of the graph Laplacian. Directly learning the eigenvalues of a graph Laplacian is computationally expensive and often fails to generalize, so instead, researchers have approximated spectral convolutions using a function such as a Chebyshev polynomial [18] or single layer perceptron [3]. Very often, successful spectral GNNs can be "spatialized" by recognizing that their functional approximations can be reinterpreted as message passing functions applied directly over the graph adjacency structure [3], [20].

Such a spatial-spectral dual interpretation is present in well-known graph algorithms such as PageRank [21] and HITS [22], and the more sophisticated diffusion processes modeled by GNNs similarly can be described both by operating over the adjacency structure of the graph and by the spectral properties of the associated graph Laplacian [19].

Spatial graph neural networks instead parameterize functions directly over the graph adjacency structure by employing a message passing [20] paradigm to extract, aggregate, and transform information from a node's local neighborhood. By stacking multiple GNN layers, such approaches can model higher order neighborhoods by incorporating information from a node's $k$-hop neighborhood. Following the early successes of GCN approaches like Kipf et al. [3], works such as graphSAGE [2] further extended ideas of message passing on graphs by incorporating random neighbor sampling and alternative permutation invariant aggregators, such as *mean*, *max*, or an LSTM. Further developments by Velikovic et al. [6] introduced attention mechanisms to dynamically compute aggregation weights for node-neighbor pairs during the message passing process.

### B. Heterogeneous GNNs

Convolutional and message passing approaches on graphs were originally limited to homogeneous networks, but following the successes of GNNs on many graph learning tasks, researchers have sought to extend GNN-based embedding frameworks to model dynamics in heterogeneous networks. R-GCN [7] extends the convolutional operator defined by Kipf et al. [3] to knowledge graphs by applying separate convolutions over relations in the networks and taking the mean of each node's representations to compute a node's representation. MNE [11] applies a similar approach that utilizes random walks over relation sub-networks in conjunction with word2vec to compute a base node embedding augmented by edge type information aggregated using a fixed aggregation weight. GATNE [8] extends MNE by applying separate node-type transformations with a graphSAGE-style convolution operator applied over edge types in a heterogeneous network for link prediction in recommender systems. GATNE replaces the static uniform edge type aggregation of MNE with edge-type attention for aggregation.

Alongside node and edge type views of heterogeneous networks, a number of approaches have relied on manually specific meta-paths to decompose heterogeneous networks. Approaches such as Metapath Aggregated Neural Network (MAGNN) [10] and Heterogeneous Attention Network (HAN) [9] extend the

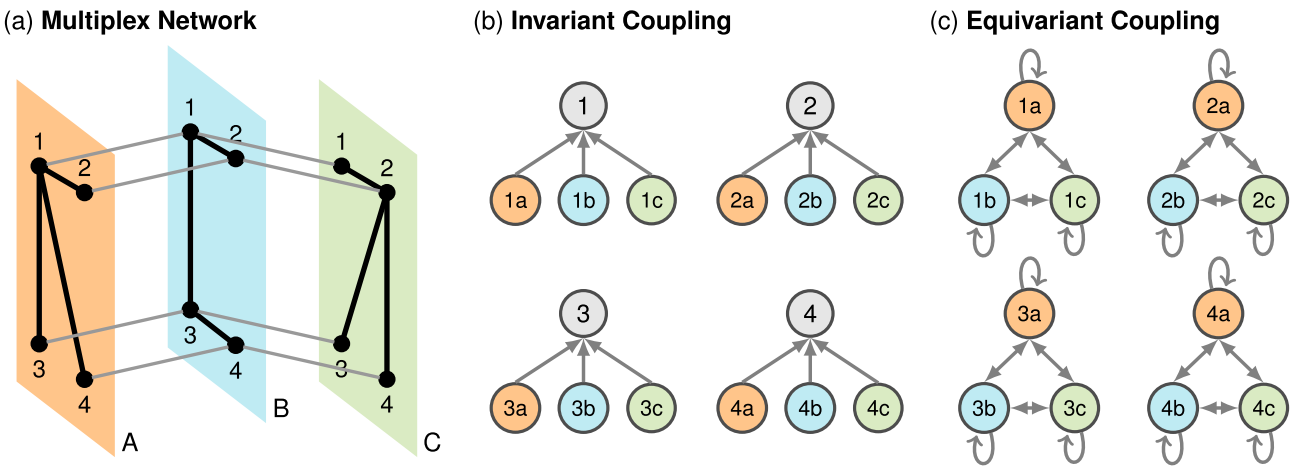


Fig. 1. Schematic of a multilayer representation of a heterogeneous graph. (a) A multiplex network consisting of relation-specific layer graphs with separate node instantiations coupled across relations. (b) Node invariant coupling graph structure in which a node's relation-specific representations are directly coupled to the supra-node to produce a singular representation of a node in a heterogeneous graph. (c) Node equivariant coupling graph structure in which node-relation instantiations are coupled with one another to produce a set of representations characterizing a node in a heterogeneous graph.

applications of attention to meta-path based GNNs by adapting GAT-style convolution operators on meta-path defined subgraphs. In all, most heterogeneous embedding frameworks listed above either rely on only node-type or edge-type alone, treat such heterogeneity separately, or require manually-specified metapaths. Additionally, most methods produce embeddings suitable for either node-level or link-level tasks, but not both types. Although there have been improvements in heterogeneous graph embedding methods, there is still room for improvement by developing flexible approaches suitable for multiple downstream tasks and by more comprehensively exploiting the information present in heterogeneous networks.

Most all heterogeneous graph neural networks decompose the process of learning representations on heterogeneous graphs in order to reduce model size and to limit the amount of data required in memory during training and inference. Both GATNE and MNE decompose heterogeneous connections separately along node type and edge type. MAGNN and HAN derive multiple representations of heterogeneous graph structures along manually specified metapaths. R-GCN and HGT take a relational view to heterogeneous graphs, which avoids the need to explicitly define metapaths and better captures the heterogeneity of networks than maintaining separate node type and edge type mechanisms [7], [23]. Decomposing heterogeneous networks into multiple views is consistent both with how we intuitively understand these networks and is supported by empirical results and smaller model sizes. Despite the almost universal tendency to decompose heterogeneous graphs for representation learning, few works have explored why such an approach works when applying graph neural networks to heterogeneous graphs, letting the intuitive nature of such approaches and the empirical results to stand for themselves [7], [8], [9], [23]. In this work, we endeavor to provide a theoretical explanation for why a relational decomposition of a heterogeneous graph is particularly powerful for heterogeneous graph representation learning by examining the spectral properties of the multiplex and drawing on the connections between GNNs and graph spectral filters. On the basis of this understanding, we develop muxGNN as a flexible and extensible heterogeneous GNN framework capable of both node invariant and equivariant representations suitable for a broad range of tasks on heterogeneous graphs.

## III. Preliminaries

Heterogeneous graphs and multiplex networks are important abstractions for modeling relational data in many real-world complex systems [14], [24]. An illustration of a multiplex network is shown in Fig. 1. Formally, they are defined as:

*Definition 1.* **Heterogeneous Graph:** A heterogeneous graph is defined as a graph $G = (\mathcal{V}, \mathcal{E}, T_v, T_e)$ where $\mathcal{V}$ is the set of nodes and $\mathcal{E}$ is the set of edges. Each node $v \in \mathcal{V}$ is associated with a type mapping function $\phi(v) : \mathcal{V} \to T_v$ and each edge $e \in \mathcal{E}$ is associated with a mapping $\psi(e) : \mathcal{E} \to T_e$, where $T_v$ and $T_e$ are the sets of node types and edge types, respectively.

*Definition 2.* **Relation:** For an edge $e = (u, v)$ in a heterogeneous graph, we define the relation $r$ as $\langle \phi(u), \psi(e), \phi(v) \rangle$. Each edge is associated with a mapping $\text{rel}(e) : \mathcal{E} \to R$, where $R$ is the set of all relations present in the graph. We can assume that there may exist multiple relations between different types of nodes in many real-world heterogeneous graphs.

*Definition 3.* **Multiplex Network:** A multiplex network is a multi-layer representation of a graph defined as a quadruple $\boldsymbol{M} = (\mathcal{V}, \mathcal{R}, B, \mathbb{M})$. $\mathcal{V}$ corresponds to the node set $\mathcal{V}$ in the heterogeneous graph and represents the set of *supra-nodes* that are separately instantiated in each layer in the multiplex system. The set of relations is represented by $\mathcal{R}$, which we call the relation layer set, as the relations in the heterogeneous graph will form the basis for each layer graph in the multiplex network. The separate instantiations of each supra-node $v \in \mathcal{V}$ across each relation $r \in \mathcal{R}$ is given by a binary relation on $\mathcal{V}$ and $\mathcal{R}$, such that $B = (\mathcal{V}, \mathcal{R}, \beta),\ \beta \subseteq \mathcal{V} \times \mathcal{R}$. In other words, the statement $(v, r) \in \beta$ reads *node v participates in relation r*.

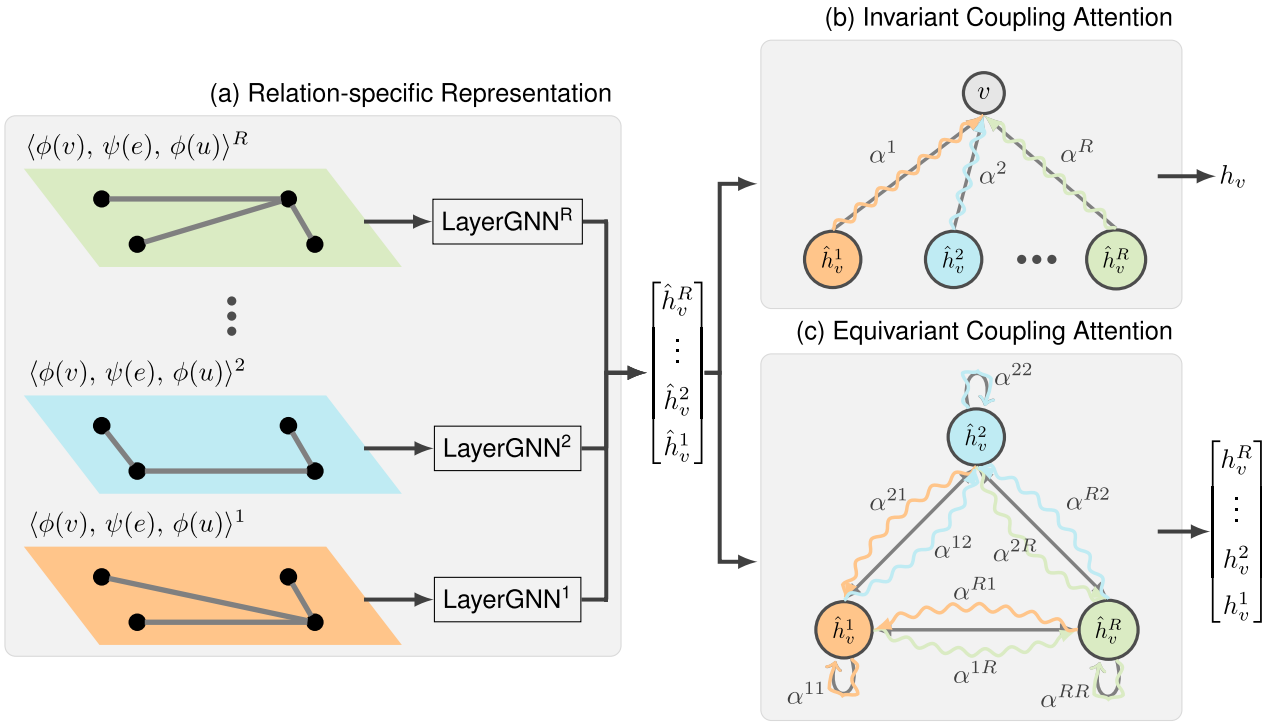


Fig. 2. Overall framework of the proposed muxGNN model. (a) The relation-specific representation of a node generated by local neighborhood aggregation across each relation in which a node participates. (b) The node-relation pair representations of a supra-node are fused through coupling weights learned via node invariant coupling attention (c) Node equivariant attention learns sets of coupling weights for a node with each relation viewed as the dominant layer.

The set of layer-graphs $\mathbb{M} = \{G^r\}_{r\in R}$ defines the relation specific graphs on node-layer pairs. Each layer graph $G^r$'s node set is subset of the node-layer pairs such that $\beta^r = \{(v,i) \subset \beta \,|\, i = r\}$, and its edge set is the subset of $\mathcal{E}$ such that $\mathcal{E}^r = \{e \in \mathcal{E} \,|\, \text{rel}(e) = r\}$, i.e. the relation triple of edge $e$ corresponds to $r$.

Lastly, we may define the interlayer coupling graph $G^C$ on $\beta$ in which there is an edge between two node-layer pairs $(v,i)$ and $(u,j)$ only if $v = u$. In other words, the graph $G^C$ defines the coupling between the same node $v \in \mathcal{V}$ across different layers. These four components together define a multiplex system encoding both intra- and inter-relation interactions in a heterogeneous graph.

## IV. muxGNN

In this section, we present the proposed muxGNN framework, illustrated in Fig. 2. The idea in muxGNN is to decompose the multiplex representation of a heterogeneous graph into two components defined by the layer graphs and the coupling graph. This allows muxGNN to define a convolution over the entire multiplex by separately parameterizing weight matrices for the specific instantiations of nodes in each layer graph and parameterizing a dynamic coupling mechanism over the coupling graph.

### A. Multiplex Spectral Convolution

Let $\boldsymbol{M} = (\mathcal{V}, \mathcal{R}, B, \mathbb{M})$ be a multiplex network with nodes $\mathcal{V} = \{1,\ldots,N\}$ and $\mathcal{R} = \{1,\ldots,R\}$ layers. Interactions between node-layer pairs are encoded by the set of layer graphs $\mathbb{M} = \{G^r\}$, and the coupling between separate instantiations of the same node across pairs of layers is encoded by the coupling graph $G^C$.

We can represent the system of layer graphs as the set of adjacency matrices $\{\boldsymbol{A}^r\}$ such that $A^r_{ij} = 1$ only if $(i,j) \in G^r$ and $A^r_{ij} = 0$ otherwise. If we restrict coupling to only between a node in one layer and that same node in other layers, the multiplex network is "diagonally" coupled, and we can define an $N \times N$ coupling adjacency matrix $\boldsymbol{C}$ such that $C_{ij} = 1$ only if $((i,n),(j,m)) \in G^C$, i.e. $i$ and $j$ correspond to the same node in different layers [14], [16]. We can then construct the *supra-adjacency* block diagonal matrix of the multiplex graph as:

$$\mathcal{A} = \bigoplus_{r\in R} \boldsymbol{A}^r + \boldsymbol{C} = \begin{pmatrix} \boldsymbol{A}^1 & \boldsymbol{C} & \ldots & \boldsymbol{C} \\ \boldsymbol{C} & \boldsymbol{A}^2 & \ldots & \boldsymbol{C} \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{C} & \boldsymbol{C} & \ldots & \boldsymbol{A}^R \end{pmatrix} \tag{1}$$

We may also consider the multiplex *supra-Laplacian* derived from the Laplacians of the layer graphs as $\boldsymbol{L}^r = \boldsymbol{D}^r - \boldsymbol{A}^r$, where $\boldsymbol{D}^r = \text{diag}(d^r_1,\ldots,d^r_N)$ is the diagonal degree matrix for layer graph $G^r$, and $\boldsymbol{L}^C = \boldsymbol{D}^C - \boldsymbol{C}$, where $\boldsymbol{D}^C = \text{diag}(d^c_1,\ldots,d^c_N)$ is the multiplexity degree matrix containing the number of layer in which a node participates along the diagonal. Furthermore, by considering quotient graphs over the multiplex network, we may decompose the supra-Laplacian of the whole multiplex into two components defined by the supra-Laplacian of the layer graphs and the supra-Laplacian of

the coupling graph [15], [16].

$$\boldsymbol{\mathcal{L}} = \boldsymbol{\mathcal{L}}^R + \boldsymbol{\mathcal{L}}^C, \tag{2}$$

where $\boldsymbol{\mathcal{L}}^R$ is the supra-Laplacian of the independent layer graphs and $\boldsymbol{\mathcal{L}}^C$ is the supra-Laplacian of the interlayer coupling graph. The layer supra-Laplacian may be expressed as:

$$\boldsymbol{\mathcal{L}}^R = \bigoplus_{r \in R} \boldsymbol{L}^r = \begin{pmatrix} \boldsymbol{L}^1 & 0 & \dots & 0 \\ 0 & \boldsymbol{L}^2 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \boldsymbol{L}^R \end{pmatrix} \tag{3}$$

where $\boldsymbol{L}^r$ represents the Laplacian matrix of graph $G^r$. As a block diagonal matrix with independent entries, the spectrum of the layer supra-Laplacian $\boldsymbol{\mathcal{L}}^R$ is given by the union of the spectra of the individual layer graph Laplacians:

$$\Lambda^R = \bigcup_r \Lambda^r$$
$$s.t. \quad \lambda^R_{max} = \max_r \lambda_r. \tag{4}$$

The interlayer coupling supra-Laplacian may be expressed as the Kronecker product of the coupling graph Laplacian, $\boldsymbol{L}^C$ and the $N \times N$ identity matrix $I$:

$$\boldsymbol{\mathcal{L}}^C = \boldsymbol{L}^C \otimes I. \tag{5}$$

The spectrum of the Kronecker product of two matrices is formed by the product of the eigenvalues of the individual matrices. Since the identity matrix only has eigenvalues equal to one, the spectrum of the coupling supra-Laplacian is spectrum of $\boldsymbol{L}^C$ with a multiplicity $N$.

If we consider the eigendecomposition of the multiplex supra-Laplacian, $\boldsymbol{\mathcal{L}} = \boldsymbol{U}\Lambda\boldsymbol{U}^T$ where $\boldsymbol{U}$ is the eigenvector matrix and $\Lambda$ is the diagonal eigenvalue matrix of $\boldsymbol{\mathcal{L}}$, it has been shown that the eigenvalues of the layer supra-Laplacian and coupling supra-Laplacian are also eigenvalues of the parent multiplex supra-Laplacian [16]. In particular, if we let $\boldsymbol{u}$ be any of the eigenvectors of $\boldsymbol{\mathcal{L}}^C$ with corresponding eigenvalue $\lambda$, then $\boldsymbol{u}$ is also an eigenvector of $\boldsymbol{\mathcal{L}}^R$ with zero eigenvalue, and thus,

$$\boldsymbol{\mathcal{L}}\boldsymbol{u} = \left(\boldsymbol{\mathcal{L}}^R + \boldsymbol{\mathcal{L}}^C\right)\boldsymbol{u} = \lambda\boldsymbol{u}. \tag{6}$$

We can alternatively characterize the spectrum of the multiplex supra-Laplacian by considering the coupling supra-Laplacian as a perturbation matrix acting on the layer supra-Laplacian, such that:

$$\bar{\lambda}^R = \lambda^R + \Delta\lambda, \tag{7}$$

where $\lambda^R$ is an eigenvalue of the layer graph supra-Laplacian, and $\Delta\lambda$ is the perturbation due to the interlayer coupling [14].

We may therefore exploit the shared eigenbasis and the relationship between the eigenvalues of the multiplex with the spectra of the layer graphs and coupling graph to define a multiplex graph convolution as:

$$(g_\theta, h_\theta) \star x_i = \boldsymbol{U}\left(g_\theta(\lambda^R) + h_\theta(\lambda^C)\right)\boldsymbol{U}^T x_i \tag{8}$$

where $g_\theta$ and $h_\theta$ are functions on the eigenvalues of the independent layer supra-Laplacian $\boldsymbol{\mathcal{L}}^R$ and the coupling supra-Laplacian $\boldsymbol{\mathcal{L}}^C$, respectively. Therefore, a convolution over a multiplex representation of a heterogeneous network can be parameterized using graph neural networks applied hierarchically over the relation graphs and the coupling graph.

### B. Layer Graph Representation

We have shown that a convolution over a multiplex system may be decomposed as two operators over the relation-specific layer graphs and over the coupling graph. Furthermore, (4) shows that the spectrum of the layer graph supra-Laplacian is itself composed of the spectra of the independent layer graphs, so we may separately parameterize convolution over each independent layer graph. Directly learning the eigenvalues of the Laplacian is generally inappropriate due to the fact that they are expensive to compute, non-localized, and for the most part do not generalize well. Instead, a common solution has been to "spatialize" the spectral operation by making the eigenvalues related to the eigenvalues of the Laplacian through a localized spatial function with coefficients defined by $c_{vu} = (f(\boldsymbol{L}))_{vu}$ [3], [19]. We are therefore able to approximate the convolution, $g_\theta(\Lambda^R)$, over the layer graph supra-Laplacian by applying a graph neural network over each relation-specific layer graph in the multiplex.

We consider three implementations from two prominent families of graph neural networks—convolutional and attentional—as the layer-wise operator in muxGNN as seen in Fig. 2(a). The neighbor coefficients may be fixed weights as in convolutional GNNs or may be computed implicitly, as in attentional GNNs. Specifically, we evaluate GCN [3] and GIN [4] as examples of convolutional GNNs and GAT [6] as an example attentional GNN.

For each layer graph $G^r \in \mathbb{M}$ in the multiplex, we compute the layer-specific representations of a node-layer pair $(v, r) \in G^r$ using a layer-wise GCN operator as:

$$\hat{\boldsymbol{h}}^{(k)}_{v,r} = \sigma\left(\mathbf{W}^r \sum_{u \in \tilde{\mathcal{N}}(v,r)} \frac{1}{\sqrt{d^r_v d^r_u}} \boldsymbol{h}^{(k-1)}_{u,r}\right) \tag{9}$$

where $\tilde{\mathcal{N}}(v, r)$ is the neighborhood of node $v$ in layer $r$ with added self-loop, $\mathbf{W}^r$ is a relation-specific trainable weight matrix, $\sigma$ is a non-linear activation function, and $d^r_v$ and $d^r_u$ are the degree of nodes $v$ and $u$ in layer $r$. This function corresponds to a first-order approximation of a spectral convolution over the symmetrically normalized relation layer graph Laplacian.

We also consider the layer-wise GIN operator is defined as:

$$\hat{\boldsymbol{h}}^{(k)}_{v,r} = \text{MLP}^r\left((1 + \epsilon^r) \cdot \boldsymbol{h}^{(k-1)}_{v,r} + \sum_{u \in \mathcal{N}(v,r)} \boldsymbol{h}^{(k-1)}_{u,r}\right) \tag{10}$$

Compared with the GCN operator, GIN notably replaces the single layer transformation with a multi-layer perceptron (MLP) and with a simple unnormalized summation function applied directly over the adjacency structure of the relation layer graph. The sum aggregator's ability to represent the entirety of a multiset is particularly important for multiplex networks, as we see in Section V.

Thirdly, we also consider a GAT layer operator, defined as:

$$\hat{\boldsymbol{h}}_{v,r}^{(k)} = \sigma\left(\mathbf{W}^r \sum_{u \in \tilde{\mathcal{N}}(v,r)} a_{vu}^r \boldsymbol{h}_{u,r}^{(k-1)}\right), \tag{11}$$

where aggregation coefficient $a_{vu}^r$ is computed dynamically in a node-dependent manner via edge softmax attention [6]. In this case, the GAT function is a first-order approximation of a convolution applied over the weighted adjacency matrix, where the edge weights are computed via the attention mechanism.

### C. Coupling Graph Attention

The second component of muxGNN, and the essential component for encoding the interlayer coupling dynamics that distinguish a multiplex system from a set of independent graphs, is the coupling graph attention mechanism. From the construction of the multiplex supra-Laplacian defined in Section IV-A, we may further define a weighted supra-Laplacian with weight parameter $\alpha$ as:

$$\bar{\boldsymbol{\mathcal{L}}} = \bigoplus_{r \in R} \boldsymbol{L}^r + \alpha \boldsymbol{L}^C I. \tag{12}$$

We can then approximate a convolution over the weighted coupling supra-Laplacian by parameterizing an attentional graph neural network over the coupling graph. By limiting coupling to within the instantiations of the same supra-node, the edge structure of the coupling graph is formed separately by all the node-layer pairs representing the same supra-node. Each supra-node $v \in \mathcal{V}$ is represented in the coupling graph by the set of feature vectors of its separate node-layer instantiations computed by the layer graph GNNs:

$$\hat{\boldsymbol{h}}_v^{(k)} = \left\{\hat{\boldsymbol{h}}_{v,r}^{(k)} \,|\, r \in R\right\}. \tag{13}$$

We can define two edge arrangements that arise naturally from the construction of multiplex system from the heterogeneous graph, illustrated in Fig. 2(b)–(c). In the *node invariant* case, the node-layer pairs are all directly connected to a synthetic supra-node. Such a coupling structure is appropriate for node-level and graph-level tasks. To compute the node invariant coupling of a node based on its node-relation instantiations, we define an attentional GNN with coefficients computed via an attention mechanism that can capture various types of semantic information to learn the importance of each relation for a give node [8], [9], [25]. The representation of a supra-node $v$ is given as the weighted sum of the learned layer representations for each of its associated node-relation instantiations:

$$\boldsymbol{h}_v^{(k)} = \sum_{r \in R} \alpha_v^r \cdot \hat{\boldsymbol{h}}_{v,r}^{(k)}. \tag{14}$$

The coupling attention measures how important a give node-relation instantiation is for the overall supra-node. First, we project the relation-specific embedding of a node through a non-linear transformation, and then we compute the importance as the similarity of the transformed representation with a relation-level attention vector $\mathbf{q}_r$ as:

$$a_v^r = \mathbf{q}_r^{\mathrm{T}} \cdot \tanh(\mathbf{W}^r \cdot \hat{\boldsymbol{h}}_{v,r}). \tag{15}$$

The coupling weight of relation $r$ for node $v$ can be obtained by normalizing the importances of each relation in which a supra-node participates using the softmax function:

$$\alpha_v^r = \frac{\exp(a_v^r)}{\sum_{r \in R} \exp(a_v^r)}. \tag{16}$$

In the node invariant coupling case, the attention mechanism applied over the composite supra-node learns the "natural" dominant layer and relative perturbations due to participation in other relations for each node in the multiplex. This supra-node level embedding computed by muxGNN with node invariant coupling is well-suited for node-level and graph-level tasks on heterogeneous networks.

We can further extend the coupling attention to operate in a *node equivariant* manner that is more suitable for link-level tasks on heterogeneous graphs. By employing multiple attention heads, each associated with a particular relation, we can enforce that the coupling attention learns multiple coupling relationships for a particular supra-node from the perspective of each relation as the "dominant" layer. Rather than a single semantic attention vector, each coupling attention head is associated with a set of relation attention vectors. The attentional GNN is applied to each node-relation instantiation in the coupling clique associated with a supra-node:

$$\left(\boldsymbol{h}_v^1, \ldots, \boldsymbol{h}_v^R\right)^{(k)} = \Big\|_{r \in R} \mathrm{attn}^r\left(\hat{\boldsymbol{h}}_v^{(k)}\right). \tag{17}$$

In this way, muxGNN with node equivariant coupling produces a set of embeddings associated with a particular supra-node representing multiple views of the node from the perspective of each relation in which the node participates.

The hierarchical application of relation-layer neighborhood aggregation with cross relation coupling attention allows muxGNN to learn node representations that incorporate soft meta-paths across multiple relations by stacking muxGNN layers over the $k$-hop neighborhoods of node $v$. The final representation, or set of representations in the equivariant case, for a node $v$ after $K$ muxGNN layers is:

$$\boldsymbol{z}_v = \boldsymbol{h}_v^{(K)}. \tag{18}$$

For graph-level tasks, we apply a permutation invariant readout function the set of node embeddings for a given graph. In our experiments we use a *sum* readout as:

$$\boldsymbol{z}_{\boldsymbol{M}} = \sum_{v \in \mathcal{V}} \boldsymbol{z}_v. \tag{19}$$

### D. Model Optimization

In this section, we describe the semi-supervised training process for muxGNN. We use random walks over the multiplex graph to generate node sequences and optimize the model to learn node representations that maximize the similarity of co-occurring node pairs [8], [26]. Specifically, we conduct random walks along each layer graph $G^r$, such that the transition probability at each step is:

$$p(u \,|\, v, G^r) = \begin{cases} \frac{1}{|\mathcal{N}(v,r)|} & (v,u) \in G^r \\ 0 & (v,u) \notin G^r \end{cases} \tag{20}$$

where $\mathcal{N}(v, r)$ denotes the neighborhood of node $v$ in layer $r$. The random walker conducts a set of walks for each node along every layer in which the node participates, capturing the heterogeneous contexts of a node in the multiplex network. A random walk of length $l$ in $G^r$ defines a path $W^r = (v_1, v_2, \ldots, v_l)$. Given a context sliding window size $c$, we define the layer context of $v$ in $G^r$ as:

$$C = \{v_j \,|\, v_i \in W^r,\ |i - j| \leq c,\ j \neq i\} \tag{21}$$

The objective for a node $v$ with a context $C$ along a path is to minimize the negative log-likelihood:

$$-\log P_\theta\left(\{u \,|\, u \in C\} \,|\, v\right) = \sum_{u \in W} \log P_\theta(u \,|\, v) \tag{22}$$

where $\theta$ represents the model parameters. We utilize the softmax function as the probability of a node $u$ given $v$:

$$P_\theta(u \,|\, v) = \frac{\exp(\boldsymbol{c}_u^T \cdot \boldsymbol{z}_v^r)}{\sum_{w \in G^r} \exp(\boldsymbol{c}_w^T \cdot \boldsymbol{z}_v^r)} \tag{23}$$

where $u, v \in G^r$, $\boldsymbol{c}_u$ is the context embedding of node $u$, and $\boldsymbol{z}_v^r$ is the embedding for node $v$ on layer $r$. To approximate the objective function, we use negative sampling for each context triplet $(v, u, r)$ as:

$$E = -\log \sigma(\boldsymbol{c}_u^T \cdot \boldsymbol{z}_v^r) - \sum_{n=1}^{N} \mathbb{E}_{w \sim P_{(G^r)}}[\log \sigma(-\boldsymbol{c}_w^T \cdot \boldsymbol{z}_v^r)], \tag{24}$$

where $\sigma$ is the sigmoid function. $N$ corresponds to the number of negative samples drawn for each positive sample, and $w$ is a node drawn randomly from noise distribution $P_{(G^r)}$ defined on the corresponding node set of layer graph $G^r$. $P_{(G^r)}$ may be either a uniform distribution or a log-uniform distribution ordered by node degree. As many real-world networks follow a power law degree distribution, a log-uniform distribution weights the probability of a node being sampled proportional to the node's degree.

## V. Experiments

In this section, we evaluate our proposed method on two common graph tasks: link prediction and graph classification. For link prediction results, we employ area under the curve (AUC) of the receiver operating characteristic (ROC) and precision-recall (PR) curves and F1 score as evaluation metrics to measure the performance of different models. For graph classification, we present accuracy, precision, recall, and F1 score for comparison. The code and datasets used in this work are available at https://github.com/NASCL/muxGNN.

### A. Datasets and Baseline Models

In this work, we conduct link prediction evaluation using our proposed muxGNN with equivariant coupling on four different datasets: Amazon,[1] Twitter,[2] YouTube,[3] Tissue Protein-Protein Interaction (PPI).[4] For graph classification, we apply the proposed embedding framework with permutation invariant coupling and a graph-level readnout function on two data provenance graph datasets, StreamSpot [27], [28] and Unicorn [29], and compare against the associated data provenance graph classification frameworks proposed in these works. To demonstrate that our method can work well in different data domains and with graphs of different scales, datasets are collected from several distinct domains and contain a range of numbers of nodes and edges. The details of the datasets are shown in Table I.

TABLE I
Summary of Datasets Used in Our Experiments. For the Graph Classification Datasets, the Average Number of Nodes, Edges, and Relation Layers is Reported

| Name | # Graph | # Node | # Edge | # Layer | Type |
|---|---|---|---|---|---|
| Amazon | | 10,099 | 135,761 | 2 | Product |
| Twitter | | 28,473 | 91,726 | 3 | Social Network |
| YouTube | | 2,000 | 1,310,544 | 5 | Social Network |
| Tissue PPI | | 4,360 | 527,850 | 10 | Biology |
| StreamSpot | 600 | 8,411 | 149,618 | 19 | Data Prov. |
| Unicorn-SC1 | 150 | 261,133 | 969,141 | 48 | Data Prov. |

We compare our proposed method with five state-of-the-art heterogeneous GNN frameworks, including MNE [11], R-GCN [7], and R-graphSAGE as examples of convolutional graph neural network models, and GATNE [8], HAN [9] and HGT [23] as examples of attentional GNN models. In addition, we select two random walk-based graph embedding methods, node2vec [30] and DeepWalk [31], for comparison. To adapt these methods to heterogeneous networks, we conduct the random walks independently over each layer graph. To explore the performance of different GNN models as the basis for the layer graph representation, we present results for muxGNN with GCN [3], GAT [6], and GIN [4] as the message passing function in the layer graph component.

### B. Link Prediction

*Experimental Setup.* To evaluate the performance of our proposed method on link prediction, we conduct one set of experiments in a transductive setting—where all nodes in the test graph are also present in the training graph—and one set of experiments in an inductive context—where some nodes in the test graph are not present in the training graph. We utilize the train/val/test splits for the Amazon and YouTube datasets provided by [8] with 85% training links and 5% and 10% positive validation and positive testing links, respectively. The Twitter dataset is similarly constructed. For the Tissue Protein-Protein Interaction dataset, we use five-fold cross validation with 80% training links and 5% and 15% positive validation and test links, respectively. For all validation and test sets, an equivalent number of non-existent links are randomly selected from the graph as negative samples to augment the positive links.

For the inductive experiment, we randomly mask 15% of nodes from the original Tissue PPI network. From the reduced

1.http://jmcauley.ucsd.edu/data/amazon/
2.https://snap.stanford.edu/data/higgs-twitter.html
3.http://socialcomputing.asu.edu/datasets/YouTube
4.http://snap.stanford.edu/ohmnet/

TABLE II
MEAN PERFORMANCE (ROC-AUC %, F1 % AND PR-AUC %) ON LINK PREDICTION IN A TRANSDUCTIVE CONTEXT. OOT: OUT OF TIME (24 HRS)

| | Amazon | | | Twitter | | | YouTube | | | Tissue PPI | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | ROC-AUC | F1 | PR-AUC | ROC-AUC | F1 | PR-AUC | ROC-AUC | F1 | PR-AUC | ROC-AUC | F1 | PR-AUC |
| node2vec | 94.47 | 87.88 | 94.30 | 72.58 | 71.94 | 82.23 | 71.21 | 65.36 | 70.32 | 51.30 | 64.04 | 66.40 |
| DeepWalk | 94.20 | 87.38 | 94.03 | 76.88 | 72.42 | 83.10 | 71.11 | 65.52 | 70.04 | 58.48 | 67.16 | 69.45 |
| MNE | 90.28 | 83.25 | 83.25 | OOT | OOT | OOT | 82.30 | 75.03 | 75.03 | OOT | OOT | OOT |
| R-graphSAGE | 94.88 | 89.39 | 93.75 | 74.31 | 70.77 | 77.77 | 81.02 | 73.91 | 77.58 | 71.15 | 65.46 | 70.71 |
| R-GCN | 94.96 | 90.08 | 93.33 | 92.76 | 85.85 | 92.88 | 80.21 | 73.36 | 75.52 | 84.19 | 75.98 | 83.08 |
| GATNE-I | 96.25 | 91.36 | 91.36 | 92.94 | 86.20 | 93.20 | 84.47 | 76.83 | 76.83 | 79.83 | 71.78 | 77.81 |
| HAN | 95.28 | 90.43 | 93.26 | 92.76 | 85.70 | 92.47 | 80.43 | 73.43 | 76.84 | 93.05 | 85.98 | 93.02 |
| HGT | 95.74 | 91.18 | 93.70 | 92.30 | 85.24 | 92.83 | 82.82 | 75.51 | 78.68 | 90.95 | 82.88 | 90.66 |
| MuxGNN$_{GCN}$ | 96.02 | 91.17 | 94.27 | 88.95 | 81.61 | 88.97 | 82.93 | 75.56 | 79.82 | 84.58 | 78.17 | 78.56 |
| MuxGNN$_{GAT}$ | 96.54 | 92.10 | 94.90 | 90.23 | 83.37 | 90.40 | 83.11 | 75.50 | 80.01 | 88.40 | 83.15 | 79.99 |
| MuxGNN$_{GIN}$ | **96.83** | **92.35** | **95.48** | **93.74** | **86.85** | **94.18** | **85.14** | **77.36** | **82.40** | **94.38** | **87.69** | **93.22** |

graph, we mask an additional 20% positive links as a validation set, along with an equivalent number of negative links. The training graph therefore has 85% of nodes and 80% training links of the training node set. In the test graph, 50% of links incident on the masked nodes are re-added with the remaining 50% of links as positive test samples, augmented by an equivalent number of non-existent links incident on the masked nodes as negative samples.

The embedding size of all models is set to 200 dimensions with all GNN models using two graph convolution layers corresponding to the 2-hop neighborhood for all nodes. For all random walks, we use 20 walks of length 10 steps with a sliding window context of size 5. The number of edge attention heads in HAN and the GAT variant of muxGNN is set to two, and the heterogeneous attention of GATNE, HAN, and the coupling graph attention size for all muxGNN variants is set to 16 dimensions. For HGT, the number of transformer attention heads is set to four. All models are trained for a maximum of 50 epochs with early stopping based on the validation ROC-AUC.

*Transductive Results and Analysis.* The link prediction results in a transductive setting are presented in Table II. It can be seen that the random walk based methods cannot achieve satisfactory performance on all datasets. Their performance notably degrades as the number of different node and edge types increases in the heterogeneous graph. In general, the attentional GNN models GATNE, HAN, and HGT outperform the convolutional graph neural networks MNE, R-graphSAGE, and R-GCN. The notable exception in the performance of GATNE on the Tissue PPI dataset. The node attributes in this dataset are a binary vector indicating biological functions in which the protein participates. These feature vectors are highly sparse compared with the dense features in the other datasets, and we hypothesize that this fact may be responsible for the relative low performance of GATNE on this dataset when compared with the other datasets and baseline models. Frameworks like GATNE, HAN, and our proposed muxGNN utilize a two-stage encoding process whereby GNN modules learn latent representations of graph structures based on the network structure that are then pipelined to an attention mechanism to aggregate and combine. By contrast, HGT integrates a transformer-style query-key-value attention mechanism into their message propagation resulting in a framework where the message propagation and attention are intertwined. Our experiments indicate that this transformer-style module does not provide any significant benefits over the GNN then attention two-stage approach. Compared with transformer-style attention, our semantic attention mechanism requires fewer parameters—requiring two transformation matrices compared with three–and can successfully operate with low dimensional attention representations as illustrated in Fig. 3. By leveraging a multiplex representation of heterogeneous graphs, muxGNN can flexibly substitute in different GNN modules with minimal alteration to the overall framework, allowing the framework to easily adapt to particular data sources and problems where a certain flavor of GNN is best suited. Additionally, such flexibility enables muxGNN to readily adapt to new GNN architectures as they are developed, allowing novel frameworks to be extended to heterogeneous graphs.

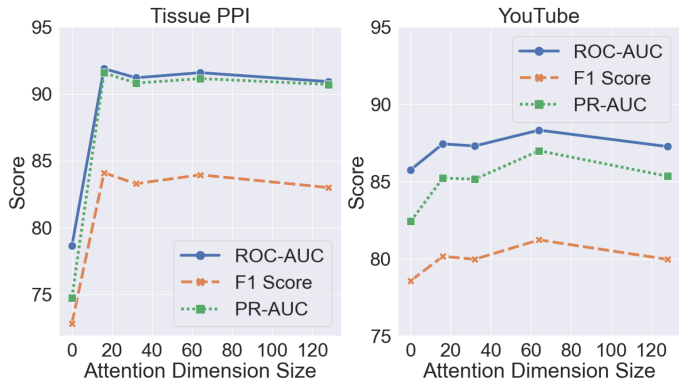


Fig. 3. Ablation study and dimension analysis of muxGNN's coupling graph attention.

The graph convolution function proposed by [4] uses the sum operation as the permutation invariant aggregation function because the sum captures the full multiset of node neighbors. In a multiplex network, it is especially important to retain the full multiset of neighbors in each layer graph. In most heterogeneous graph datasets, there is only a single supra-node feature representation that is repeated across each relation layer. Additionally, the construction of a multilayer heterogeneous graph representations increases the likelihood of repeated or highly

TABLE III
MEAN PERFORMANCE (ROC-AUC %, F1 % AND PR-AUC %) ON LINK PREDICTION IN AN INDUCTIVE CONTEXT

| | Tissue PPI | | |
|---|---|---|---|
| | ROC-AUC | F1 | PR-AUC |
| R-graphSAGE | 66.10 | 61.59 | 63.15 |
| R-GCN | 78.08 | 70.98 | 76.81 |
| GATNE-I | 71.36 | 67.64 | 70.63 |
| HAN | 87.53 | 79.59 | 88.12 |
| HGT | 84.68 | 76.73 | 84.77 |
| MuxGNN$_{GCN}$ | 82.47 | 75.02 | 82.98 |
| MuxGNN$_{GAT}$ | 87.48 | 79.62 | 88.06 |
| MuxGNN$_{GIN}$ | **90.44** | **82.74** | **90.97** |

similar local neighborhood topologies across the relation layer graphs. All layer-specific representations together encompass a supra-node's structural role in the heterogeneous network. A particular supra-node is likely to appear multiple times across the set of layer-specific representations, resulting in a high multiplicity for a given supra-node across neighborhood sets in each layer. Compared with a *sum* aggregator as the permutation invariant function in the layer graph neural network function, a *mean* aggregator only captures the distribution of elements and the *max* ignores multiplicities altogether and reduces a multiset to a simple set [4]. As such, the layer graph representations generated by GIN retain the full multiset of a given supra-node's representations in each layer graph. This allows the coupling graph attention layer to operate over the full multiset of layer-specific representations for a given supra-node across the entire multiplex graph. The experimental results show that our proposed method can learn better features to represent a target link for prediction in heterogeneous networks from multiple domains. In addition, our method is more stable than other baseline methods even as the heterogeneity of the networks increases.

*Inductive Results and Analysis.* The link prediction results in an inductive setting are presented in Table III. For these experiments, we include all baseline models that learn an inductive transformation function and are thus capable of link prediction on nodes previously unseen in training. As in the transductive setting, the GIN variant of muxGNN outperforms all baselines. The GAT variant also performs well, equaling the performance of the strongest baseline competitor in HAN. We observe that muxGNN$_{GAT}$ experiences the smallest drop in performance between the inductive and transductive settings. GATNE experiences an approximately 8% reduction in performance in the inductive setting compared with the transductive setting, which we may attribute to the highly sparse node features in this dataset. The other models in our experiments see an approximately 3-5% reduction in performance across all three metrics in the inductive setting compared with the transductive setting. By contrast muxGNN$_{GAT}$ achieves a roughly equivalent performance regardless of whether all nodes in the test set were present during training. Despite this, muxGNN$_{GIN}$ still achieves the best performance on the inductive link prediction task, outperforming all baselines and other muxGNN variants.

### C. Graph Classification

*Experimental Setup.* To evaluate our proposed model on the task of heterogeneous graph classification, we conduct a set of experiments to classify data provenance graphs as generated from benign execution patterns or from attack executions. We utilize two advanced persistent threat (APT) datasets and accompanying baseline models from StreamSpot [27], [28] and Unicorn [29]. The Streamspot model uses Locality Sensitive Hashing (LSH) to compute vector representations of graphs that preserve similarity. The StreamSpot dataset consists of data provenance graphs generated from normal web browsing activity, video gaming, and software downloading as a benign class. The positive class consists of graphs generated from a drive-by-download attack on the host machine via a malicious URL. The Unicorn model uses the Weisfeiler-Lehman (WL) subtree kernel to embed graphs while preserving similarity, and its associated dataset provides data provenance graphs generated from CamFlow [1], a whole-system data provenance monitoring tool. Benign system executions compose the negative class, and the positive class is composed of a set of graphs generated from Trojan horse and remote code execution attacks via a malicious URL. Both models rely on graph summarization techniques to generate representations of graphs. To compute a graph level embedding, a *sum* readout function is applied to the set supra-node embeddings of a graph. We set the graph embedding dimension to 64 and use a single hop neighborhood for neighbor aggregation. An MLP classifier is jointly trained with the muxGNN embedding model using cross entropy loss. For both datasets, we use five-fold cross validation, such that our model is trained on 80% of the available graphs with 20% of graphs held out as an unseen test set. We report the mean classification accuracy, precision, recall, and F1 score across all test folds.

*Results and Analysis.* The graph classification results are presented in Table IV. MuxGNN achieves near perfect performance on the StreamSpot dataset, whose isolated scenarios capture'' simplified workloads and is not representative of typical workloads in modern systems. That said, such graphs provide a standard for comparing different threat detection models and are a similar design to many of today's microarchitecture services. Compared with StreamSpot's baseline model, both the graph kernel based method and muxGNN's graph neural network approach realize a significant increase in performance. When compared with Unicorn's graph kernel approach, considering both the single-hop and three-hop neighborhoods of nodes, we see that muxGNN outperforms both variants when considering only a one-hop neighborhood of nodes.

Methods based on the Weisfeiler-Lehman isomorphism test and subtree kernel are well-known for their discriminative power and have many connections to graph neural network methods. Graph neural networks can be seen as an extension of Weisfeiler-Lehman graph isopmorhism test with non-linear neural networks employed in place of the hashing function [2], [4]. These improvements provide GNNs with greater expressive capability when transforming graph signals over the local topology of the network.

TABLE IV
MEAN PERFORMANCE (ACCURACY, PRECISION, RECALL, AND F1) OF DIFFERENT METHODS ON DATA PROVENANCE GRAPH CLASSIFICATION. ONLY TWO DIGITS OF PRECISION REPORTED BY [29]. RECALL AND F-SCORE NOT REPORTED BY [27]

| | StreamSpot | | | | Unicorn-SC1 | | | |
|---|---|---|---|---|---|---|---|---|
| | Accuracy | Precision | Recall | F1 | Accuracy | Precision | Recall | F1 |
| StreamSpot | 0.66 | 0.74 | - | - | - | - | - | - |
| Unicorn (k = 1) | 0.51 | 1.0 | 0.60 | 0.68 | - | - | - | - |
| Unicorn (k = 3) | 0.98 | 0.93 | **0.96** | 0.94 | 0.90 | 0.85 | **0.96** | **0.90** |
| MuxGNN$_{GCN}$ (k=1) | 0.9916 | 1.0 | 0.9470 | 0.9727 | 0.9402 | 0.9167 | 0.7229 | 0.8078 |
| MuxGNN$_{GAT}$ (k=1) | 0.9916 | 1.0 | 0.9470 | 0.9727 | 0.9649 | 1.0 | 0.7667 | 0.8677 |
| MuxGNN$_{GIN}$ (k=1) | **0.9919** | **1.0** | 0.9516 | **0.9752** | **0.9674** | **1.0** | 0.7967 | 0.8866 |

The Unicorn dataset consists of only 150 total graphs, which is a very limited size for training neural network models. The small, unbalanced dataset along with the heterogeneity and large individual size of the provenance graphs presents a challenging dataset for GNNs optimized via gradient descent both in terms of convergence and avoiding overfitting. Despite these challenges, the one-hop muxGNN model is competitive with the corresponding three-hop Unicorn model. MuxGNN achieves a significant improvement in overall accuracy and in precision, while Unicorn maintains strong performance in recall, with both models overall F1-score approximately equivalent. In addition, the increased computation and memory complexity of Unicorn requires summarization of the provenance graphs prior to embedding generation. In such a way, muxGNN better captures the vertical heterogeneity of a node across the multiple activity types through its coupling attention. By leveraging the advantages of GNNs, such as node batching and weight sharing, muxGNN is able to operate over the entire provenance graphs when computing graph embeddings.

### D. Coupling Graph Attention Analysis

The core component of muxGNN is the coupling graph attention layer. To further analyze the effects of inter-layer coupling, we conduct an ablation study by removing coupling attention from muxGNN. We also investigate the layer coupling attention weight distributions for protein-tissue pairs in the Tissue PPI dataset and show that our proposed coupling attention recovers expected biological connections between tissue layers.

*Ablation Study:* In the ablation study, the coupling graph attention layer is removed from muxGNN. Each layer graph convolution is performed independently, and representations of a node in a given layer encode only it's local neighborhood information within that layer. Fig. 3 illustrates the performance of muxGNN$_{GIN}$ on the Tissue PPI and YouTube datasets. As can be seen, removing the coupling graph attention layer from muxGNN reduces overall performance on both datasets; though, the extent of performance decrease varies between datasets. The layers in the YouTube dataset correspond to different user actions on the social network-retweet, quote, or reply. Our link prediction experiments indicate that user interactions of a particular type are largely independent of a user's behavior of other types. That is to say, whether a user is likely to retweet another user is only marginally influenced by that user's reply and quoting behavior. What our experiments indicate is that there is only weak inter-layer coupling in the YouTube dataset, and there is only a marginal perturbation of a node's representation in one layer by the other node-layer pairs. By contrast in the Tissue PPI dataset, inclusion of the coupling graph attention layer is essential to accurately capture a protein's role across multiple tissue types. Here we observe strong inter-layer coupling that significantly impacts the likelihood of two proteins interacting in any particular tissue. We further explore the coupling attention in the Tissue PPI dataset in the following section.

*Tissue PPI Layer Coupling:* The coupling graph attention is critical for capturing the multi-faceted context of nodes in heterogeneous networks. In addition to improving the performance of muxGNN on downstream graph learning staks, we hypothesize that our coupling attention discovers interpretable connections in the network, which provides a level of explainability to muxGNN that is not present in other heterogeneous graph embedding frameworks. We examine the learned attention weights for proteins in each tissue layer in the Tissue PPI dataset. For our experiments, we extract the ten largest layers in network corresponding to the blood, brain, central nervous system (CNS), fetus, heart, kidney, leukocyte, lung, nervous system (NS), and testis. Fig. 4 illustrates the distribution of attention values for every protein node in the network with respect to each tissue type in the dataset. It can be seen that the muxGNN's coupling attention identifies the importance of biologically-related tissues in predicting protein interactions. Fig. 4(a) shows the attention distribution for the brain layer where we note that both the CNS and nervous system layers are attended to by the model when learning latent representation of proteins in the brain. Similarly, Fig. 4(b) shows the attention distributions for the CNS layer, where both the brain and NS layers are important. By contrast, Fig. 4(d) shows that the more general nervous system layer does not attend as highly over the brain and CNS layers compared with other tissue layers. These findings indicate that the many proteins roles in the brain and CNS are tightly coupled, which is reflected in the perturbation of layer-specific representations of such protein nodes. Lastly, Fig. 4(c) shows that the attention values for the leukocyte layer, where we see that white blood cell proteins attend to other proteins expressed in the blood. In all, our analysis of the attention weight distributions learned by muxGNN illustrate that the coupling graph attention produces interpretable and biologically consistent importance weights across the tissue types present in the protein-protein interaction network.

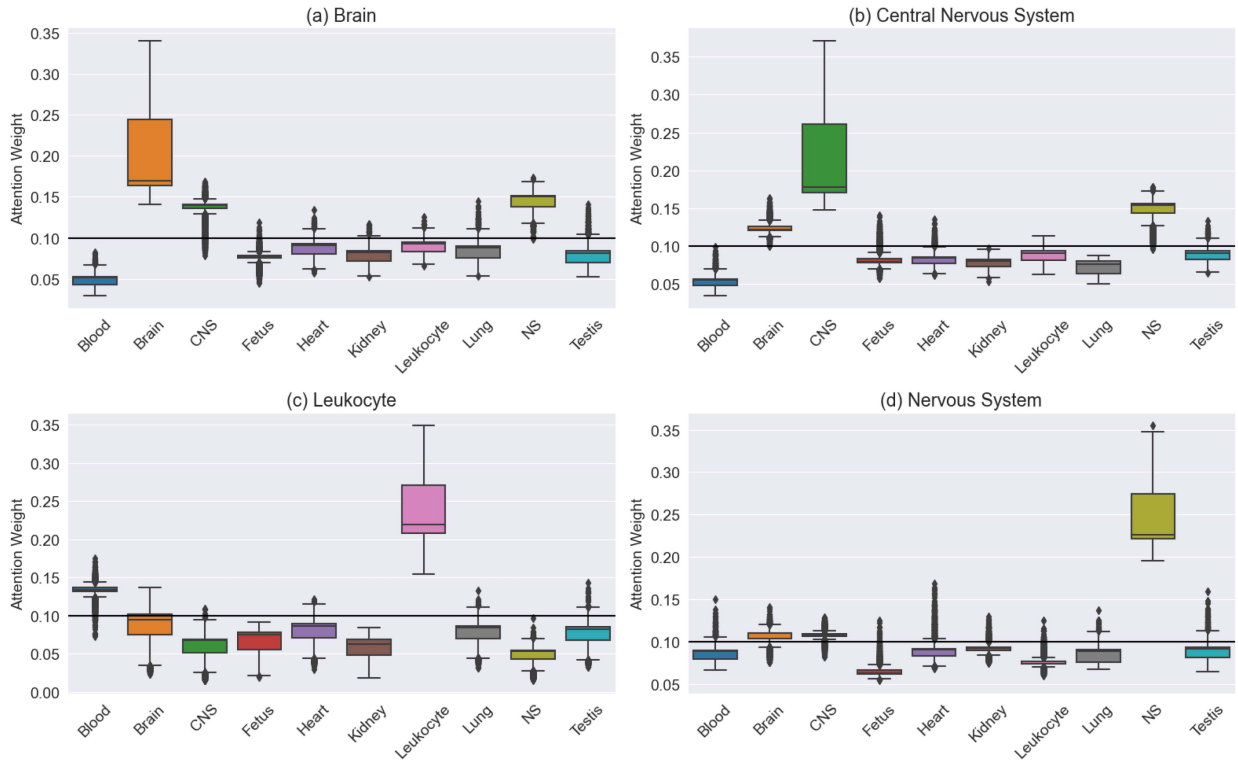


Fig. 4. Distributions of coupling attention values for each protein in the (a) Brain, (b) CNS, (c) Leukocyte, and (d) NS layers in the Tissue-PPI dataset.

## VI. Conclusion

In this work, we introduce muxGNN, a multiplex graph neural network for heterogeneous graphs. Our framework models heterogeneity in networks by characterizing nodes and edges by constructing a multiplex representation of heterogeneous graphs. The multiplex representation consists of a set of relation-specific graphs that captures the local topology of nodes with respect to each edge type in the network and a coupling graph that links the individual instantiations of nodes across the different relation layers. We demonstrate that due to the shared eigenbasis of a multiplex and its component graphs, a convolution over the entire multiplex can be decomposed by separately parameterizing convolutions over each relation graph and over the coupling graphs. We develop a novel coupling attention mechanism to dynamically capture the multi-faceted semantic contexts of nodes in heterogeneous graphs, and we further extend this method to characterize both node invariant and node equivariant attention allowing muxGNN to be applied for node-level, link-level, or graph-level downstream tasks. We conduct extensive experiments on graph classification and on link prediction in both transductive and inductive contexts, demonstrating that the proposed muxGNN model can capture both network heterogeneity and provides superior performance compared with state-of-the-art heterogeneous GNN models. A case study of our coupling attention on a protein-protein interaction dataset further illustrates that muxGNN discovers interpretable connections between node contexts in heterogeneous graphs.

In the future, we will explore the use of the multiplex construct for temporal graph analysis, the development of novel coupling graph structures, and the incorporation of higher-order structures such as graph motifs as a means to better model the myriad contexts in heterogeneous graphs.

## Acknowledgments

The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Army Research Office or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation herein.

## References

[1] T. Pasquier et al., "Practical whole-system provenance capture," in *Proc. Symp. Cloud Comput.*, 2017, pp. 405–418.
[2] W. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2017, pp. 1024–1034.
[3] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *Proc. Int. Conf. Learn. Representations*, 2017.
[4] K. Xu, W. Hu, J. Leskovec, and S. Jegelka, "How powerful are graph neural networks?," in *Proc. Int. Conf. Learn. Representations*, 2019.
[5] G. Cui, J. Zhou, C. Yang, and Z. Liu, "Adaptive graph encoder for attributed graph embedding," in *Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2020, pp. 976–985.
[6] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, "Graph attention networks," in *Proc. Int. Conf. Learn. Representations*, 2018.

[7] M. Schlichtkrull et al., "Modeling relational data with graph convolutional networks," in *The Semantic Web*. Cham, Switzerland: Springer, 2018, pp. 593–607.
[8] Y. Cen, X. Zou, J. Zhang, H. Yang, J. Zhou, and J. Tang, "Representation learning for attributed multiplex heterogeneous network," in *Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2019, pp. 1358–1368.
[9] X. Wang et al., "Heterogeneous graph attention network," in *Proc. World Wide Web Conf.*, 2019, pp. 2022–2032.
[10] X. Fu, J. Zhang, Z. Meng, and I. King, "MAGNN: Metapath aggregated graph neural network for heterogeneous graph embedding," in *Proc. Web Conf.*, New York, NY, USA: Association for Computing Machinery, 2020, pp. 2331–2341.
[11] H. Zhang, L. Qiu, L. Yi, and Y. Song, "Scalable multiplex network embedding," in *Proc. Int. Joint Conf. Artif. Intell.*, 2018, pp. 3082–3088.
[12] C. Yang et al., "MultiSage: Empowering GCN with contextualized multi-embeddings on web-scale multipartite networks," in *Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2020, pp. 2434–2443.
[13] D. Taylor, M. A. Porter, and P. J. Mucha, "Tunable eigenvector-based centralities for multiplex and temporal networks," *Multiscale Model. Simul.*, vol. 19, no. 1, pp. 113–147, 2021.
[14] E. Cozzo, G. F. D. Arruda, F. A. Rodrigues, and Y. Moreno, "Multilayer networks: Metrics and spectral properties," in *Interconnected Networks*. Berlin, Germany: Springer, 2016, pp. 17–35.
[15] R. Sánchez-García, E. Cozzo, and Y. Moreno, "Dimensionality reduction and spectral properties of multiplex networks," *Phys. Rev. E*, vol. 89, no. 5, May 2014, Art. no. 052815, doi: 10.1103/PhysRevE.89.052815.
[16] A. Solé-Ribalta, M. De Domenico, N. E. Kouvaris, A. Diaz-Guilera, S. Gómez, and A. Arenas, "Spectral properties of the laplacian of multiplex networks," *Phys. Rev. E*, vol. 88, Sep. 2013, Art. no. 032807.
[17] J. Bruna, W. Zaremba, A. Szlam, and Y. LeCun, "Spectral networks and locally connected networks on graphs," 2013, *arXiv:1312.6203*.
[18] M. Defferrard, X. Bresson, and P. Vandergheynst, "Convolutional neural networks on graphs with fast localized spectral filtering," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2016, pp. 3844–3852.
[19] M. M. Bronstein, J. Bruna, T. Cohen, and P. Velickovic, "Geometric deep learning: Grids, groups, graphs, geodesics, and gauges," *CoRR*, vol. abs/2104.13478, pp. 77–80, 2021.
[20] J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals, and G. E. Dahl, "Neural message passing for quantum chemistry," in *Proc. Int. Conf. Mach. Learn.*, 2017, pp. 1263–1272.
[21] L. Page, S. Brin, R. Motwani, and T. Winograd, "The PageRank citation ranking: Bringing order to the web," Stanford InfoLab, 1999.
[22] J. M. Kleinberg, "Authoritative sources in a hyperlinked environment," *J. ACM*, vol. 46, no. 5, pp. 604–632, Sep. 1999.
[23] Z. Hu, Y. Dong, K. Wang, and Y. Sun, "Heterogeneous graph transformer," in *Proc. Web Conf.*, 2020, pp. 2704–2710, doi: 10.1145/3366423.3380027.
[24] M. Kivelä, A. Arenas, M. Barthelemy, J. P. Gleeson, Y. Moreno, and M. A. Porter, "Multilayer networks," *J. Complex Netw.*, vol. 2, no. 3, pp. 203–271, 2014.
[25] Z. Lin et al., "A structured self-attentive sentence embedding," 2017, *arXiv:1703.03130*.
[26] Y. Dong, N. V. Chawla, and A. Swami, "metapath2vec: Scalable representation learning for heterogeneous networks," in *Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2017, pp. 135–144.
[27] E. Manzoor, S. Momeni, V. Venkatakrishnan, and L. Akoglu, "Fast memory-efficient anomaly detection in streaming heterogeneous graphs," in *Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2016, pp. 1035–1044.
[28] X. Han, "Streamspot dataset," 2018. [Online]. Available: https://doi.org/10.7910/DVN/83KYJY
[29] X. Han, T. F. J. Pasquier, A. Bates, J. Mickens, and M. I. Seltzer, "UNICORN: Runtime provenance-based detector for advanced persistent threats," in *Proc. Netw. Distrib. Syst. Secur. Symp.*, 2020.
[30] A. Grover and J. Leskovec, "node2vec: Scalable feature learning for networks," in *Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2016, pp. 855–864.
[31] B. Perozzi, R. Al-Rfou, and S. Skiena, "Deepwalk: Online learning of social representations," in *Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2014, pp. 701–710.

**Joshua Melton** received the BS degree in biology, the BA degree in philosophy from UNC-Chapel Hill, in 2014, and the BS degree in computer science from UNC-Charlotte, in 2020. He is currently working toward the PhD degree in computer science with UNC-Charlotte. His main research interests include graph neural networks for heterogeneous and multilayer networks, computational social science, and data provenance analysis for cybersecurity applications.

**Siddharth Krishnan** received the BS/MS degrees in mathematics from Sri Sathya Sai University, in India, the MS degree in computer science from Florida State University, and the PhD degree in computer science from Virginia Polytechnic Institute. He is an assistant professor with the Department of Computer Science, UNC-Charlotte. At UNC-Charlotte, he directs the Network Analytics and Social Computing Lab. He has published in several data science venues including WSDM, WebSci, AAAI, and TKDD. His research interests are in web-mining, data analytics, computational social science, and applied machine learning with a primary emphasis on analyzing, characterizing, and forecasting information (news, rumors, memes, advertisements, etc.) dynamics on online social networks and social media.