Bi-Level Attention Graph Neural Networks

Roshni G. Iyer
University of California, Los Angeles
roshnigiyer@cs.ucla.edu

Wei Wang
University of California, Los Angeles
weiwang@cs.ucla.edu

Yizhou Sun
University of California, Los Angeles
yzsun@cs.ucla.edu

***Abstract*—Recent graph neural networks (GNNs) with the attention mechanism have historically been limited to small-scale homogeneous graphs (HoGs). However, GNNs handling heterogeneous graphs (HeGs), which contain several entity and relation types, all have shortcomings in handling attention. Most GNNs that learn graph attention for HeGs learn either node-level or relation-level attention, but not both, limiting their ability to predict both important entities and relations in the HeG. Even the best existing method that learns both levels of attention has the limitation of assuming graph relations are independent and that its learned attention disregards this dependency association. To effectively model both multi-relational and multi-entity large-scale HeGs, we present Bi-Level Attention Graph Neural Networks (BA-GNN), scalable neural networks (NNs) that use a novel bi-level graph attention mechanism. BA-GNN models both node-node and relation-relation interactions in a personalized way, by hierarchically attending to both types of information from local neighborhood contexts instead of the global graph context. Rigorous experiments on seven real-world HeGs show BA-GNN consistently outperforms all baselines, and demonstrate quality and transferability of its learned relation-level attention to improve performance of other GNNs.**

***Index Terms*—graph neural networks, representation learning**

I. Introduction

Highly multi-relational data are characteristic of real-world HeGs. Relational data in HeGs are defined as triples of form *(h:head entity, r:relation, t:tail entity)*, indicating that two entities are connected by a specific relation type. Figure 1 shows a HeG formed by such triples. However, even comprehensive HeGs [1] remain incomplete. Regarding HeGs completion, despite the recent years' research progress in developing GNNs for representation learning in various domains [7], [8], [14] and adapting the successful attention mechanism [17], [18], most GNNs face several challenges. They either are ill-equipped to handle HeGs [9], [18], or do handle HeGs but do not learn graph attention [4], [6], [13], [23], or learn inaccurate graph attention [5], [12], [19], [22].

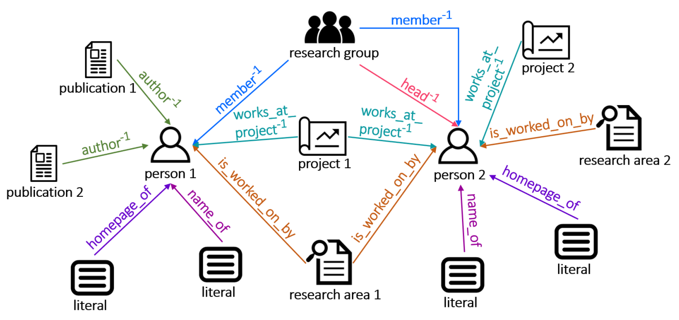


Fig. 1: Partial HeG of AIFB dataset.

Considering the GNNs that learn graph attention, their architectures are limited to only one level of attention, either for nodes or relations, but rarely for both, shown in Table I. This is problematic for modeling HeGs which contain several different entity and relation types. Bi-level attention is more powerful in learning compared to uni-level attention, where only one level of attention is learned by the model. Bi-level attention learns attention at different levels of granularity in HeGs which captures more information about graph components than a uni-level attention mechanism is capable of. HAN, one of the few models that attempts to use bi-level attention, unsurprisingly falls short of capturing the associations between the node and relation levels in the HeG. First, HAN places unnatural assumptions on the data because it treats graph relations as independent from each other, omitting most relation-relation interactions in HeGs. Second, it requires manually chosen meta paths that force many node-node and node-relation interactions to also be left out, and requires domain specific knowledge to compute. Third, HAN lacks a general framework for systematically studying bi-level attention.

To address the above challenges, in this paper, we present **Bi**-Level **A**ttention **G**raph **N**eural **N**etworks (BA-GNN) for HeGs. To summarize, our work makes the following contributions:

(1) We design a general framework for bi-level attention, and identify challenges of state-of-art NNs for HeGs.
(2) We propose BA-GNN to model both multi-relational and multi-entity large-scale HeGs. BA-GNN avoids manually chosen meta paths, and learns personalized graph properties by integrating graph entity/relation types, graph structure, and graph attention using local graph neighborhoods instead of global graph context. BA-GNN improves accuracy of state-of-art GNNs and scales to million-node/edge graphs, like the AM archaeological dataset.
(3) To our knowledge, we are the first to propose efficient bi-level attention GNNs that learn from dependency interactions of both nodes/relations and without meta paths.
(4) We rigorously experiment on seven real-world HeGs showing BA-GNN consistently outperforms major state-of-art NN groups, and also demonstrate quality and transferability of BA-GNN's attention-induced change in graph structure to enrich other GNNs.

The remainder of this paper is organized as follows. Section II examines preliminaries and related work. Section III presents a general framework for computing bi-level attention, and describes BA-GNN's architecture. Section IV presents experiment results, ablation studies, and case studies of BA-

TABLE I: Properties of GNN and attention-based models, with ✓ as advantages, and ✗ as disadvantages.

Type	Model	[A]	[B]	[C]	[D]	†[E]	[F]	[G]
–	TRANSFORMER [17]	✓	✓	✓	✗	✓	✓	–
(1)	TRANSE [2]	✓	✓	✗	✗	✗	✓	✗
	HOLE [10]	✓	✓	✗	✗	✗	✓	✗
	DISTMULT [21]	✓	✓	✗	✗	✗	✓	✗
	COMPLEX [16]	✓	✓	✗	✗	✗	✓	✗
(2)	GCN [9]	✗	✓	✗	✗	✗	✓	✗
	GAT* [18]	✗	✓	✓	✗	✓	✓	✗
(3A)	METAPATH2VEC [4]	✓	✓	✗	✗	✗	✗	✗
	HEREC [14]	✓	✓	✗	✗	✗	✗	✗
	HIN2VEC [6]	✓	✓	✗	✗	✗	✗	✗
	HEGAN [7]	✓	✓	✗	✗	✗	✓	✗
	TEMPORALGAT [5]	✗	✗	✓	✗	✓	✓	✗
	HETGNN [23]	✓	✓	✗	✗	✗	✓	✗
	R-GCN* [13]	✓	✓	✗	✗	✗	✓	✗
	HAN* [19]	✓	✓	✓	✓	✗	✗	✗
(3B)	DYSAT [12]	✓	✗	✓	✗	✓	✓	✓
	TGAT [20]	✗	✗	✓	✗	✓	✓	✓
	HGT [8]	✓	✓	✓	✗	✓	✓	✓
	GTN [22]	✓	✓	✓	✗	✓	✗	✓
–	BA-GNN (ours)	✓	✓	✓	✓	✓	✓	✓

† Personalized graph attention learns attention using the local graph neighborhood instead of the global graph context.
*Primary baseline models
Notation: [A]: Does not require HoGs; [B]: Does not require a dynamic graph; [C]: Learns attention; [D]: Bi-level attention; [E] Personalized attention; [F]: Does not require meta paths; [G]: Transformer-inspired; (1): Non-GNN-based KGE models for HeGs; (2): GNNs for HoGs; (3A): Non-TRANSFORMER-based GNNs for HeGs; (3B): TRANSFORMER-based GNNs for HeGs

GNN models, and section V concludes.

II. Preliminary and Related Work

Here, we introduce HeG concepts and discuss the achievement of various state-of-art NNs, summarized in Table I.

Definition 1: **Heterogeneous Graph:** We define HeGs as $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with nodes $v_i \in \mathcal{V}$, and edges $e_{i,j} \in \mathcal{E}$ connecting source v_i and target v_j. Nodes in the HeG are associated with entity types through an entity mapping function $\Lambda(v_i) : \mathcal{V} \rightarrow \mathcal{B}, \mathcal{B} = \{b | b \in \mathcal{B}\}$, for entity type b. Edges in the HeG are associated with relation types through a relation mapping function $\Gamma(e_{i,j}) : \mathcal{E} \rightarrow \mathcal{R}, \mathcal{R} = \{r | r \in \mathcal{R}\}$ for relation type r. For efficiency in our model, we compute entity and relation mapping functions for node and relation neighborhoods rather than globally such that $\Lambda(v_i) : \mathcal{V}_i \rightarrow \mathcal{B}_i, \mathcal{B}_i \subset \mathcal{B}$ and $\Gamma(e_{i,j}) : \mathcal{E}_i \rightarrow \mathcal{R}_i, \mathcal{R}_i \subset \mathcal{R}$. In this paper, we consider local HeG neighborhoods of a node $v_i \in \mathcal{V}$ consisting of both one-hop nodes, $\{v_j | v_j \in \mathcal{V}_i\}$ and one-hop relations, $\{r | r \in \mathcal{R}_i\}$.

Definition 2: **Meta Relation:** The meta relation for $e_{i,j}$ between source v_i and target v_j is $(\Lambda(v_i), \Gamma(e_{i,j}), \Lambda(v_j))$, and $\Gamma(e_{i,j})^{-1} = \Gamma(e_{j,i})$ is the inverse of $\Gamma(e_{i,j})$. In this paper, we loosely use the term *relation* to denote meta relation. Traditional meta paths are a sequence of such meta relations.

Definition 3: **Graph Attention:** Graph attention enables NNs to learn useful graph representations by selectively attending to different nodes and relations. Multiplicative and additive attention are state-of-art attention mechanisms used in NNs [17], [18], both of which operate on encoder states. Multiplicative attention uses an inner product or cosine similarity of encoder states while additive attention is a linear combination or concatenation of encoder states.

A. GNNs for Homogeneous Graphs

Successful models in this category, like GAT [18], use attention-based neural architectures for learning representations.

Graph Attention Networks: GAT [18] are additive attention-based GNNs that effectively leverage graph structure and sparsity to compute a node's attention. GAT models, however, are limited to HoGs and cannot handle HeGs which contain different relations that may have varying levels of importance for different nodes.

B. GNNs for Heterogeneous Graphs

Successful models (1) leverage different graph relations, like R-GCN, (2) learn bi-level attention, like HAN, and (3) learn multiplicative attention, like TRANSFORMER-based NNs.

Relational Graph Convolutional Networks: R-GCNs [13] extend GCNs and GAT, which operate on local graph neighborhoods of HoGs, to operate on multi-relational graphs by distinguishing nodes by relation type. R-GCNs, however, treat all relation-specific nodes as equally important. Further, R-GCNs do not utilize graph attention as they are limited to directly learning from weight parameters.

Heterogeneous Graph Attention Networks: To address limitations of the above models, HAN integrates bi-level attention, which learns node- and relation-level attention, with GNNs to learn node embeddings. However, HAN uses a global learnable weight vector lacking local inter-relation comparison. Besides, HAN uses pre-defined metapaths which are computationally expensive to design and compute, and result in sub-optimal graph components learned by the model.

Transformer: TRANSFORMER models [17], although successful in natural language processing for small text sequences, have limitations for multi-relational and multi-entity HeGs. This is because TRANSFORMER attends to all other tokens in the sequence, making it infeasible for large-scale input. While recent works extend TRANSFORMER-like attention to other graph domains, they have limitations, shown in Table I.

III. BA-GNN Architecture

We design a general bi-level attention framework for computing hierarchical attention and then discuss BA-GNN's architecture. Source code and data are at: https://github.com/roshnigiyer/BA-GNN. READ.md details dataset properties, data splits, and hyperparameters of BA-GNN models.

A. General Bi-Level Attention Framework

Bi-level attention in HeGs incorporates interactions between relation-specific nodes for learning lower level attention which informs the higher level attention that captures inter-relation interactions. In this way, bi-level attention jointly attends to node-node, relation-relation and node-relation interactions to collectively produce a representative node embedding. Uni-level attention models omit these critical graph interactions and ability for the two-levels of attention to jointly inform each other. Eq. 1 describes the general bi-level attention framework to compute embeddings for each v_i in the HeG:

$$\begin{aligned}\widetilde{\mathbf{h}}_i^{(l+1)} &= \mathbf{HigherAtt}\big(\mathbf{LowerAtt}(\cdot), \mathcal{R}, \widetilde{\mathbf{h}}^{(l)}\big) \qquad (1)\\ &= \mathrm{AGG}\left(\left\{\mathbf{f}_\psi\left(\left\{\mathbf{LowerAtt}(\cdot) | r \in \mathcal{R}, \widetilde{\mathbf{h}}^{(l)}\right\}\right) \middle| r \in \mathcal{R}\right\}\right),\end{aligned}$$

$$\mathbf{LowerAtt}(\cdot) = \mathrm{AGG}\Big(\{\mathbf{g}_\gamma(e_{i,j}|r, \widetilde{\mathbf{h}}^{(l)})|v_j \in N_i^r\}\Big), \quad (2)$$

$$\mathcal{R} = \{\Gamma(e_{i,j})|v_j \in \mathcal{V}\}, \quad (3)$$

where $\mathbf{h}_i$ and $\widetilde{\mathbf{h}}_i$ are the initial and projected node features respectively, $\mathcal{R}_i$ is the relation set on the edge of v_i, and $\mathbf{g}_\gamma(\cdot)$ is a vector-output function of the node-level attention, γ, that provides a relation-specific embedding summary using learned node representations from the previous layer, $\widetilde{\mathbf{h}}^{(l)}$, which is aggregated, $\mathrm{AGG}(\cdot)$, over edges $e_{i,j}$ in the relation-specific neighborhood context of $v_j \in N_i^r$. $\mathbf{f}_\psi(\cdot)$ is a vector-output function of the relation-level attention, ψ, that are attended relation-specific local context embeddings, $\mathbf{LowerAtt}(\cdot)$, which are aggregated over relations in the neighborhood context to form the layer's final node embedding, $\widetilde{\mathbf{h}}_i^{(l+1)}$.

In Sections III-B and III-C, we propose a novel semi-supervised attention-based GCN model, BA-GNN, for multi-relational and multi-entity HeGs. BA-GNN performs attention aggregation on node and *relation* levels, rather than nodes and edges. For node-level attention, attention is placed on the *edges* of neighbor nodes. For relation-level attention, attention is placed on the relations, which are formed by *grouping* edges by relation type. In this way, our model uses a hierarchical attention mechanism. The higher-order graph considers relation-type-specific edge groups, and the lower-order graph considers nodes in their local contexts. Figure 2 summarizes BA-GNN's attention mechanism. BA-GNN models use L stacked layers, each of which is defined through Eq. 1, and for further efficiency, all nodes and relations are restricted to their neighborhoods. Model input can be chosen as pre-defined features or as a unique one-hot vector for each node.

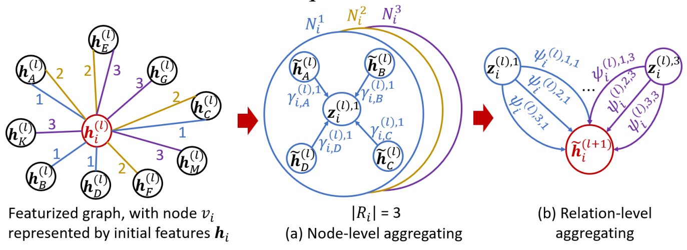

Featurized graph, with node v_i represented by initial features $\boldsymbol{h}_i$
$|R_i|$ = 3
(a) Node-level aggregating
(b) Relation-level aggregating

Fig. 2: Bi-level attention visualization. (a) Node-level aggregating: A node's features is a weighted combination of its prior layer's relation-specific embeddings, $\mathbf{z}_i^r$. (b) Relation-level aggregating: Relation-level attention is learned via multiplicative attention using neighborhood relational similarity to determine relative relation importance.

B. Node-level Attention

Node-level attention distinguishes different roles of nodes in the neighborhood context for learning relation-specific node embeddings. As node-level attentions are target-node-specific, they are different for different target nodes. Our model learns node embeddings such that it intrinsically captures graph attributes and structure in its neighborhood. In HeGs, neighbor nodes may belong to different feature spaces due to different node types, so we utilize an entity-specific type transformation matrix, $\boldsymbol{\mathcal{T}}_{\Lambda(v_i)}$, to project all node features to the same space through $\widetilde{\mathbf{h}}_i = \boldsymbol{\mathcal{T}}_{\Lambda(v_i)} \cdot \mathbf{h}_i$. $\boldsymbol{\mathcal{T}}_{\Lambda(v_i)} \in \mathbb{R}^{d \times |\mathcal{R}_i|}, \mathbf{h}_i \in \mathbb{R}^{|\mathcal{R}_i|}$ if the initial features are chosen to be a one-hot vector of dimension d, where $\widetilde{\mathbf{h}}_i$ is continuously updated to learn the final embedding.

BA-GNN's node-level attention uses additive attention inspired by GAT, discussed in Section II, but overcomes GAT's limitation by extending the attention to HeGs. GAT performs projections that do not consider the different relation types in the HeG. We address this by using a learnable relation-specific attention vector, $\mathbf{a}_r^{(l)} \in \mathbb{R}^{2d}$. For a specific relation r, the attention is shared for all node pairs, so that each node is influenced by its neighborhood context. The attention is also asymmetric since the importance of v_j to v_i may be different from the importance of v_i to v_j. We compute relation-specific node-level attention at layer l as follows, with a $\mathrm{softmax}(\cdot)$ activation applied to normalize each node-pair attention weight and where $v_j, v_k \in N_i^r$ and $\mathbf{x}^{T^{(l)}}$ is transpose of $\mathbf{x}$ at layer l:

$$\gamma_{i,j}^{(l),r} = \frac{\exp\Big(\mathrm{LeakyReLU}\Big(\mathbf{a}_r^{T^{(l)}}\Big[\widetilde{\mathbf{h}}_i^{(l)}\Big|\Big|\widetilde{\mathbf{h}}_j^{(l)}\Big]\Big)\Big)}{\sum_{v_k \in N_i^r}\exp\Big(\mathrm{LeakyReLU}\Big(\mathbf{a}_r^{T^{(l)}}\Big[\widetilde{\mathbf{h}}_i^{(l)}\Big|\Big|\widetilde{\mathbf{h}}_k^{(l)}\Big]\Big)\Big)},$$

where $\mathbf{a}_r^{(l)}$ attends over the concatenated, $||$, node features of v_i and v_j with applied $\mathrm{LeakyReLU}(\cdot)$ and $\mathrm{softmax}(\cdot)$ activations. By restricting the attention to within the relation-specific local context of nodes, sparsity structural information is injected into the model through adjacency-masked attention layers.

Node v_i's relation-specific embedding, $\mathbf{z}_i^{(l),r}$, can then be learned with $\mathrm{AGG}(\cdot)$ from Eq. 1 being a weighted summation of the neighbor's projected features as follows:

$$\begin{aligned} \mathbf{z}_i^{(l),r} &= \mathbf{LowerAtt}(\cdot) \quad (4)\\ &= \mathrm{AGG}\Big(\{\mathbf{g}_\gamma(e_{i,j}|r, \widetilde{\mathbf{h}}^{(l)})|v_j \in N_i^r\}\Big)\\ &= \textstyle\sum_{v_j \in N_i^r}\Big[\mathbf{g}_\gamma(e_{i,j}|r, \widetilde{\mathbf{h}}^{(l)})\Big]\\ &= \textstyle\sum_{v_j \in N_i^r}\Big[\gamma_{i,j}^{(l),r}\widetilde{\mathbf{h}}_j^{(l)}\Big], \end{aligned}$$

where $\mathbf{z}_i^{(l),r}$ provides a summary of relation r for v_i at layer l. We also add skip-connections for corresponding $\mathbf{LowerAtt}(\cdot)$ from the previous layer, $l-1$, to preserve learned node-level representations as the depth of the NN is extended.

C. Relation-level Attention

Relation-level attention distinguishes roles of different relations in the neighborhood context for learning more comprehensive node embeddings. In HeGs, different relations may play different roles of importance for v_i, in addition to v_i's relation-specific neighbor nodes. So, we learn relation-level attention to better fuse v_i's relation-specific node embeddings. One could design a simple node-relation attention mechanism to encode the effect of relations between nodes, but this would fail to capture relation-relation dependencies hidden in HeGs.

We address TRANSFORMER's inefficiency for large-scale HeGs through an approximation technique by sampling relation-specific node embeddings from the local graph context. Further, instead of using the same single set of projections for all words, we enable each relation-specific embedding to learn a distinct set of personalized projection weights, while maximizing parameter sharing. This technique captures unique relation-dependent characteristics such that each relation-specific node embedding is also influenced by its local relation context.

Node v_i's relation-specific TRANSFORMER-based query $\mathbf{q}_{r,i}^{(l)}$, key $\mathbf{k}_{r,i}^{(l)}$, and value $\mathbf{v}_{r,i}^{(l)}$ vectors are computed as follows: $\mathbf{q}_{r,i}^{(l)}; \mathbf{k}_{r,i}^{(l)}; \mathbf{v}_{r,i}^{(l)} = \mathbf{W}_{1,r}\mathbf{z}_i^{(l),r}; \mathbf{W}_{2,r}\mathbf{z}_i^{(l),r}; \mathbf{W}_{3,r}\mathbf{z}_i^{(l),r}$, such that $\mathbf{z}_i^{(l),r}$ is projected onto the learnable weight matrices $\mathbf{W}_{1,r}, \mathbf{W}_{2,r}, \mathbf{W}_{3,r} \in \mathbb{R}^{d \times d}$. The relation-level attention for relations (r, r') are computed by iterating over all possible relation pairs in the neighborhood context, $r, r' \in \mathcal{R}_i$, where $\mathcal{R}_i = \{\Gamma(e_{i,j}) | v_j \in \mathcal{V}_i\}$. The importance of relation r' of node v_i is as follows, with relation similarity being captured through $\psi_i^{(l),r,r'} = \mathrm{softmax}(\mathbf{q}_{r,i}^{T^{(l)}} \mathbf{k}_{r',i}^{(l)})$, where the more similar r' is to r, the greater the attention weights of r', which results in more contribution of r''s embedding to v_i's final embedding. A $\mathrm{softmax}(\cdot)$ activation is then applied to normalize each relation pair's attention weight.

A node's relation-specific embedding is then informed by a weighted summation of its similarity to other local context relations, $\psi_i^{(l),r,r'}$. To reduce information loss, we add a self-connection of a special relation type per node, which is projected onto $\mathbf{W}_i$, and aggregated to the attended relation-specific embedding, $\psi_i^{(l),r,r'} \mathbf{v}_{r',i}^{(l)}$. Lastly, a $\mathrm{ReLU}(\cdot)$ activation is applied to get the overall relation-specific embedding, $\boldsymbol{\delta}_i^{(l),r} = \mathrm{ReLU}(\sum_{r' \in \mathcal{R}_i} \psi_i^{(l),r,r'} \mathbf{v}_{r',i}^{(l)} + \mathbf{W}_i \widetilde{\mathbf{h}}_i^{(l)})$.

Node v_i's final embedding is learned with $\mathrm{AGG}(\cdot)$ in Eq. 1 being a summation of all attended relation-specific embeddings through iteration of neighborhood relations, $r \in R_i$. We also apply multi-head attention of $S = \{1, ..., K\}$ heads to allow BA-GNN to jointly attend to different representation subspaces of nodes/relations, with aggregation, $\mathrm{AGG}(\cdot)$, via averaging:

$$\begin{aligned} \widetilde{\mathbf{h}}_i^{(l+1)} &= \mathbf{HigherAtt}\big(\mathbf{LowerAtt}(\cdot), \mathcal{R}, \widetilde{\mathbf{h}}^{(l)}\big) \qquad (5) \\ &= \tfrac{1}{K} \textstyle\sum_{k=1}^{K} \sum_{r \in \mathcal{R}} \Big[\mathbf{f}_\psi \Big(\big\{ \mathbf{LowerAtt}(\cdot) \big| r \in \mathcal{R}, \widetilde{\mathbf{h}}^{(l)} \big\} \Big) \Big] \\ &= \tfrac{1}{K} \textstyle\sum_{k=1}^{K} \sum_{r \in \mathcal{R}_i} \Big[\mathbf{f}_\psi \Big(\big\{ \mathbf{LowerAtt}(\cdot) \big| r \in \mathcal{R}_i, \widetilde{\mathbf{h}}^{(l)} \big\} \Big) \Big] \\ &= \mathrm{AGG}\Big(\big\{ \textstyle\sum_{r \in \mathcal{R}_i} \Big[\boldsymbol{\delta}_i^{(l),r} \Big] \big| k \in S \big\} \Big). \end{aligned}$$

We also add skip-connections for corresponding $\mathbf{HigherAtt}(\cdot)$ from the previous layer, $l - 1$, to preserve learned higher-order relation-level representations as depth of the NN is extended.

The final representation of a node at layer $(l + 1)$ is:

$$\begin{aligned} \widetilde{\mathbf{h}}_i^{(l+1)} = \mathrm{AGG}\Bigg(\Big\{ \textstyle\sum_{r \in \mathcal{R}_i} \mathrm{ReLU}(\\ \textstyle\sum_{r' \in \mathcal{R}_i} \mathrm{softmax}(\mathbf{q}_{r,i}^{T^{(l)}} \mathbf{k}_{r',i}^{(l)}) \mathbf{v}_{r',i}^{(l)} + \mathbf{W}_i \widetilde{\mathbf{h}}_i^{(l)}) \Big| k \in S \Big\} \Bigg). \qquad (6) \end{aligned}$$

D. Analysis of Proposed Attention

We use multiplicative attention at the relation level, instead of additive attention, because learning attention through a concatenation of features does not compute feature similarity which inner product operations capture. Since relation features are characterized by single attribute relation types, a relation's scaling can be directly determined by its feature similarity to other relations in its neighborhood. This is unlike node-level attention, where node features may have several attributes, making it more difficult to learn latent similarity of nodes through direct feature comparison such as through inner product computation. Rigorous evaluation of mixed combinations of additive/multiplicative attention shown in experiments further support our bi-level attention combination choice.

IV. EXPERIMENTS

In this section, we evaluate BA-GNN on seven large-scale heterogeneous datasets (HDs). We conduct experiments on node classification and link prediction using Pytorch Geometric and Deep Graph Library frameworks on an Nvidia Tesla V100 GPU cluster, and report model test accuracies.

Datasets: We evaluate on benchmark Resource Description Framework (RDF) format datasets [11] for node classification: AIFB, MUTAG, BGS, and AM. For link prediction, we evaluate on FB15k [15], WN18 [3], and FB15k-237 [15].

A. Node Classification

Node classification is the semi-supervised classification of nodes to entity types. For evaluation consistency against primary baseline models, we implement BA-GNN with $L = 2$ and where the output of the final layer uses a $\mathrm{softmax}(\cdot)$ activation per node. Our model follows the same node classification evaluation procedure as [13], using cross-entropy loss with parameters learned from the Adam Optimizer.

Baselines: Table I summarizes our baselines. To adapt the models to our problem setting of multi-relational, static HeGs, we made the following modifications. For GAT [18] and GCN [9], we omit HeG relations. For TEMPORALGAT [5], we omit the temporal convolutional network used for temporal interactions. For HETGNN [23], we consider the neighbors to be the entire set of neighbor nodes and relations. For DYSAT [12], we omit temporal attention. For TGAT [20], we omit functional time encoding. For HGT [8], we omit relative temporal encoding.

Results: Experiment results are in Table II. Results show BA-GNN significantly and consistently outperforms all baselines for all tasks on all datasets. For example, on AIFB, MUTAG, BGS, and AM, against the most competitive NNs per category, BA-GNN achieves relative performance gains of up to 22%, 25%, 20%, and 24% respectively, and overall performance gains of up to 32%, 33%, 36%, and 35% respectively. Further, the Welch t-test of unequal variance shows that BA-GNN's relative performance compared to each model per dataset is statistically significant to be greater, with $\text{p-value} < 0.001$.

Ablation Studies: To further analyze BA-GNN's bi-level attention, we design the following BA-GNN variant models: BA-GNN-NODE, a uni-level attention model using BA-GNN's node-level attention; BA-GNN-RELATION, a uni-level attention model using BA-GNN's relation-level attention; BA-GNN(NODE)+HAN(REL.), a hybrid model using BA-GNN's node-level attention and HAN's relation-level attention.

The bi-level attention models outperform the uni-level attention models, shown in Table II. Further, simply comparing BA-GNN's higher-order relation-level attention with HAN's, shows a significant relative performance gain on all datasets. For example on MUTAG, the performance gain is nearly 11.30% with the replacement of only the relation-level attention. This

TABLE II: Node classification averaged accuracy (%) over 10 model trains on four datasets. Best results per model group are bold-faced, with overall best results per dataset underscored. We compute test accuracy improvement of BA-GNN over best NNs, all NNs, and per NN by the difference of BA-GNN's average accuracy and NNs per dataset column (e.g., rows 24-25), or the difference of BA-GNN's average accuracy and each NN across datasets (col. 7). $\mathrm{A}_{\text{NODE, REL.}}/\mathrm{M}_{\text{NODE, REL.}}$ are BA-GNN additive/mutliplicative attention at node/relation levels.

Type	Model	AIFB	MUTAG	BGS	AM	BA-GNN (% improve)
(1) Non-GNN-based KGE models for HeGs	TRANSE [2]	66.89 ± 1.26	54.41 ± 0.73	58.46 ± 0.38	61.23 ± 0.57	+[32.05, 36.29]
	HOLE [10]	67.98 ± 0.42	**63.17** ± 0.26	72.74 ± 0.22	65.25 ± 0.38	+[22.01, 31.43]
	DISTMULT [21]	73.64 ± 0.10	62.06 ± 0.18	68.19 ± 0.13	69.12 ± 0.24	+[25.3, 27.56]
	COMPLEX [16]	**77.24** ± 0.15	62.38 ± 0.22	**74.72** ± 0.18	**72.39** ± 0.16	+[20.03, 25.43]
(2) GNNs for HoGs	GCN [9]	91.99 ± 0.21	**67.02** ± 0.08	**78.74** ± 0.16	86.82 ± 0.60	+[6.95, 20.79]
	GAT [18]	**92.50** ± 0.29	66.18 ± 0.00	77.93 ± 0.17	**88.52** ± 1.65	+[6.44, 21.63]
(3A) Non-TRANSFORMER-based GNNs for HeGs	METAPATH2VEC [4]	89.52 ± 0.12	66.04 ± 0.27	78.34 ± 0.13	85.48 ± 0.11	+[9.42, 21.77]
	HEREC [14]	91.03 ± 0.15	66.96 ± 0.18	79.36 ± 0.25	85.98 ± 0.07	+[7.91, 20.85]
	HIN2VEC [6]	91.63 ± 0.17	66.29 ± 0.14	79.01 ± 0.12	86.22 ± 0.21	+[7.31, 21.52]
	HEGAN [7]	92.33 ± 0.13	68.07 ± 0.08	81.60 ± 0.27	86.79 ± 0.14	+[6.61, 19.74]
	TEMPORALGAT [5]	93.42 ± 0.11	66.88 ± 0.24	79.14 ± 0.13	89.10 ± 0.13	+[5.52, 20.93]
	HETGNN [23]	95.18 ± 0.16	75.64 ± 0.09	82.05 ± 0.25	89.67 ± 0.05	+[3.76, 12.70]
	R-GCN [13]	95.31 ± 0.62	73.23 ± 0.48	83.10 ± 0.80	89.29 ± 0.35	+[3.63, 14.58]
	HAN [19]	**96.25** ± 0.12	**76.46** ± 0.07	**86.84** ± 0.21	**90.68** ± 0.23	+[2.69, 11.35]
(3B) TRANSFORMER-based GNNs for HeGs	DYSAT [12]	92.64 ± 0.21	66.57 ± 0.05	78.02 ± 0.19	88.90 ± 1.05	+[6.30, 21.24]
	TGAT [20]	92.84 ± 0.14	67.19 ± 0.21	78.35 ± 0.15	89.43 ± 0.28	+[6.10, 20.62]
	HGT [8]	95.97 ± 0.15	**76.84** ± 0.12	**86.01** ± 0.18	90.33 ± 0.13	+[2.97, 10.97]
	GTN [22]	**96.04** ± 0.17	76.32 ± 0.12	85.38 ± 0.24	**90.56** ± 0.10	+[2.90, 11.49]
BA-GNN variants	BA-GNN-NODE	95.46 ± 0.13	73.19 ± 0.25	84.23 ± 0.22	89.45 ± 0.02	–
	BA-GNN-RELATION	95.28 ± 0.23	76.17 ± 0.22	85.43 ± 0.34	90.52 ± 0.18	–
	BA-GNN(NODE)+HAN(REL.)	96.30 ± 0.09	76.48 ± 0.04	86.80 ± 0.24	90.67 ± 0.35	–
	BA-GNN ($\mathrm{M}_{\text{NODE}}/\mathrm{A}_{\text{REL.}}$)	96.02 ± 0.13	76.20 ± 0.09	86.90 ± 0.17	90.54 ± 0.11	–
	BA-GNN ($\mathrm{A}_{\text{NODE}}/\mathrm{A}_{\text{REL.}}$)	96.38 ± 0.14	76.61 ± 0.22	87.00 ± 0.09	90.73 ± 0.05	–
	BA-GNN ($\mathrm{M}_{\text{NODE}}/\mathrm{M}_{\text{REL.}}$)	96.44 ± 0.16	77.01 ± 0.07	86.92 ± 0.12	90.78 ± 0.08	–
	BA-GNN (ours)	**98.94** ± 0.13	**87.81** ± 0.11	**94.75** ± 0.08	**96.68** ± 0.14	–
BA-GNN (% improve)	against ***best*** NNs, group 1-3	+[2.69, 21.70]	+[10.97, 24.64]	+[7.91, 20.03]	+[6.00, 24.29]	–
	against ***all*** NNs, group 1-3	+[2.69, 32.05]	+[10.97, 33.40]	+[7.91, 36.29]	+[6.00, 35.45]	–

TABLE III: Link prediction results for mean reciprocal rank (MRR), and Hits @ n metrics. For each group of models, the best results are bold-faced. The overall best results on each dataset are underscored.

	FB15k					WN18					FB15k-237				
	MRR		Hits @			MRR		Hits @			MRR		Hits @		
Model	Raw	Filtered	1	3	10	Raw	Filtered	1	3	10	Raw	Filtered	1	3	10
R-GCN	0.251	0.651	0.541	0.736	0.825	0.553	0.814	0.686	0.928	0.955	0.158	0.248	0.153	0.258	0.414
BA-GNN	**0.261**	**0.702**	**0.601**	**0.778**	**0.857**	**0.590**	**0.820**	**0.698**	**0.945**	**0.959**	**0.195**	**0.260**	**0.160**	**0.268**	**0.468**
TRANSE	0.221	0.380	0.231	0.472	0.641	0.335	0.454	0.089	0.823	0.934	0.144	0.233	0.147	0.263	0.398
R-GCN$_T$	0.252	0.651	0.543	0.738	0.828	0.554	0.815	0.681	0.928	0.956	0.161	0.258	0.159	0.274	0.421
BA-GNN$_T$	**0.264**	**0.700**	**0.649**	**0.781**	**0.858**	**0.593**	**0.822**	**0.692**	**0.943**	**0.960**	**0.200**	**0.268**	**0.168**	**0.275**	**0.493**
HOLE	0.232	0.524	0.402	0.613	0.739	0.616	0.938	0.930	0.945	0.949	0.124	0.222	0.133	0.253	0.391
R-GCN$_H$	0.257	0.659	0.556	0.744	0.839	0.667	0.937	0.935	0.951	0.966	0.159	0.257	0.156	**0.272**	0.420
BA-GNN$_H$	**0.268**	**0.720**	**0.670**	**0.787**	**0.860**	**0.670**	**0.940**	**0.942**	**0.955**	**0.979**	**0.194**	**0.266**	**0.161**	**0.272**	**0.488**
DISTMULT	0.248	0.634	0.522	0.718	0.814	0.526	0.813	0.701	0.921	0.943	0.100	0.191	0.106	0.207	0.376
R-GCN$_D$	0.262	0.696	0.601	0.760	0.842	0.561	0.819	0.697	0.929	0.964	0.156	0.249	0.151	0.264	0.417
BA-GNN$_D$	**0.272**	**0.745**	**0.688**	**0.792**	**0.868**	**0.600**	**0.825**	**0.705**	**0.934**	**0.977**	**0.190**	**0.251**	**0.155**	**0.268**	**0.483**
COMPLEX	0.242	0.692	0.599	0.759	0.840	0.587	0.941	0.936	0.945	0.947	0.109	0.201	0.112	0.213	0.388
R-GCN$_C$	0.260	0.712	0.629	0.771	0.845	0.615	0.953	0.937	0.947	0.965	0.158	0.255	0.152	0.268	0.419
BA-GNN$_C$	**0.278**	**0.788**	**0.731**	**0.867**	**0.880**	**0.671**	**0.978**	**0.981**	**0.988**	**0.992**	**0.170**	**0.262**	**0.159**	**0.270**	**0.485**

indicates that BA-GNN's relation-level attention is more effective across the different data domains and that its personalized attention to local graph contexts yields performance gain.

B. Link Prediction

Link prediction involves assigning confidence scores to HeG triples to determine how likely predicted edges belong to true relations. Our models follow the same evaluation framework as [2] and [13] using negative sampling and cross-entropy loss with parameters learned from the Adam Optimizer. We use evaluation metrics of mean reciprocal rank (MRR) and Hits @ n, in raw and filtered settings. The same number of negative samples, $w = 1$, are used to make datasets comparable.

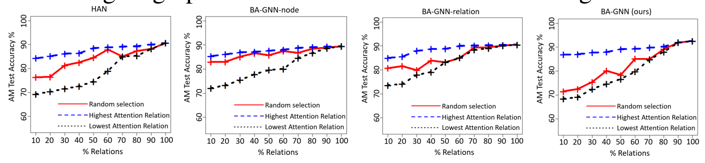
Baselines: We evaluate standalone GNNs (BA-GNN, R-GCN), KGE models, and GNN+KGE autoencoder models using the same setup procedure as [2] and [13]. The autoencoder models include: BA-GNN$_x$ and R-GCN$_y$ where x, y are TRANSE (T), HOLE (H), DISTMULT (D), and COMPLEX (C).

Results and Ablation Studies: Experiment results are in Table III. Results show that the best BA-GNN models outperform R-GCN models on all datasets for all metrics of both tasks of MRR and Hits @ n = 1, 3, 10. We observe that BA-GNN outperforms R-GCN when comparing standalone models. Further, results show that autoencoder models outperform each of GNN and KGE standalone models, showing that GNNs and KGE models can each be benefited by their joint learning. Results also show that BA-GNN autoencoders outperform R-GCN autoencoders on all datasets for all tasks and metrics.

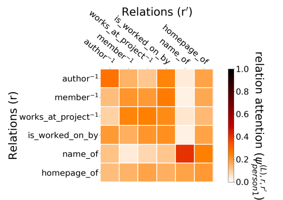
C. Case Study

We conduct experiments to determine the quality of relation-level attention and graph-structure of BA-GNN. We modify the AM dataset to contain the following relation types, each with cummulative 10% splits: (1) relations randomly selected, (2) relations with the highest relation-level attention weights from BA-GNN, and (3) relations with the lowest relation-level attention weights from BA-GNN. Experiment figures on node classification for HAN and BA-GNN models are in Figure 3(a).

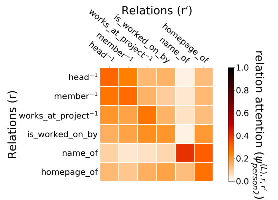
(2)'s graph structure yields the highest test accuracy on all splits of AM compared to (1) or (3), while (3) yields the lowest test accuracy. (1) is as expected in between test accuracies of (2) and (3). Models that do not learn relation-level attention (BA-GNN-NODE) still benefit from the graph structure identified by (2). This suggests that BA-GNN's relation-level attention can selectively identify important graph components and that its learning of graph structure can enhance other leading GNNs.



(a) BA-GNN's attention-induced graph structure for HAN, BA-GNN-NODE, BA-GNN-RELATION, BA-GNN



(b) pp1 heat map: BA-GNN's relation-level attention



(c) pp2 heat map: BA-GNN's relation-level attention

Fig. 3: Experiments evaluating learned relation-level attention and graph structure of BA-GNN.

D. Attention Visualization

We randomly sample two nodes belonging to entity *Persons* on AIFB and plot its learned relation-level attention weights from layer $l = L$ using heat maps, seen in Figures 3(b) and (c). The corresponding partial graphs of *person 1* and *person 2* are in Figure 1. In Figure 3(b), $author^{-1}$ and $member^{-1}$ have high attention to *is_worked_on_by* because a person is likely to have publications and research affiliations in their research area. *name_of* has high attention to *homepage_of*, observed in both Figures 3(b) and (c), because a homepage may directly contain personal identifying information. In Figure 3(c), $head^{-1}$ and $member^{-1}$ have high attention to each other, since head of a research group is a member. Further, members of the research group are likely to work on the group's projects and focus on a particular research domain, explaining why $works\_at\_project^{-1}$ has higher attention to $head^{-1}$ and $member^{-1}$, and $works\_at\_project^{-1}$ and members of the research group also have higher attention to *is_worked_on_by*.

V. CONCLUSION

We propose Bi-Level Attention Graph Neural Networks (BA-GNN) for modeling multi-entity and multi-relational large-scale heterogeneous graphs (HeGs), via entity type and meta relation information to learn graph structure and properties. Further, BA-GNN distinguishes nodes and relations using a novel smart-sampling bi-level attention mechanism to guide the model when aggregating features in graph neighborhoods. We conduct extensive experiments on seven real-world heterogeneous datasets, and show BA-GNN learns effective and efficient embeddings. We observe that BA-GNN outperforms all state-of-art GNN baselines on various information recovery tasks.

VI. ACKNOWLEDGEMENTS

This work was supported by NSF 1705169, 1829071, 1937599, 2031187, 2106859; NIH R35-HL135772, NIBIB R01-EB027650; DARPA HR00112090027; Okawa Foundation Grant; and Amazon Research Awards. We also thank Kai-Wei Chang and Yunsheng Bai for helpful discussions.

REFERENCES

[1] K. Bollacker, C. Evans, P. Paritosh, T. Sturge, and J. Taylor. Freebase: a collaboratively created graph database for structuring human knowledge. In *In SIGMOD Conference*, pages 1247–1250, 2008.

[2] A. Bordes, N. Usunier, A. Garcia-Duran, J. Weston, and O. Yakhnenko. Translating embeddings for modeling multi-relational data. NIPS 2013.

[3] T. Dettmers, P. Minervini, P. Stenetorp, and S. Riedel. Convolutional 2d knowledge graph embeddings. AAAI 2018.

[4] Y. Dong, N. V. Chawla, and A. Swami. metapath2vec: Scalable representation learning for heterogeneous networks. KDD 2017.

[5] A. Fathy and K. Li. TemporalGAT: Attention-based dynamic graph representation learning. PAKDD 2020.

[6] T. Fu, W. Lee, and Z. Lei. HIN2Vec: Explore meta-paths in heterogeneous information networks for representation learning. CIKM 2017.

[7] B. Hu, Y. Fang, and C. Shi. Adversarial learning on heterogeneous information networks. KDD 2019.

[8] Z. Hu, Y. Dong, K. Wang, and Y. Sun. Heterogeneous graph transformer. WWW 2020.

[9] Q. Li, Z. Han, and X. ming Wu. Deeper insights into graph convolutional networks for semi-supervised learning. AAAI 2018.

[10] M. Nickel, L. Rosasco, and T. Poggio. Holographic embeddings of knowledge graphs. https://arxiv.org/abs/1510.04935, 2015.

[11] P. Ristoski and H. Paulheim. Rdf2vec: Rdf graph embeddings for data mining. In *International Semantic Web Conference*. Springer, 2016.

[12] A. Sankar, Y. Wu, L. Gou, W. Zhang, and H. Yang. Dynamic graph representation learning via self-attention networks. ICLR 2019.

[13] M. Schlichtkrull, T. N. Kipf, P. Bloem, R. van den Berg, I. Titov, and M. Welling. Modeling relational data with graph convolutional networks. https://arxiv.org/abs/1703.06103, 2017.

[14] C. Shi, B. Hu, and W. X. Zhao. Heterogeneous information network embedding for recommendation. IEEE 2018.

[15] K. Toutanova and D. Chen. Observed versus latent features for knowledge base and text inference. In *Proceedings of the 3rd Workshop on Continuous Vector Space Models and their Compositionality*, 2015.

[16] T. Trouillon, J. Welbl, S. Riedel, E. Gaussier, and G. Bouchard. Complex embeddings for simple link prediction. ICML 2016.

[17] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Łukasz Kaiser, and I. Polosukhin. Attention is all you need. NIPS 2017.

[18] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio. Graph attention networks. ICLR 2018.

[19] X. Wang, H. Ji, C. Shi, B. Wang, P. Cui, P. Yu, and Y. Ye. Heterogeneous graph attention network. WWW 2019.

[20] D. Xu, C. Ruan, E. Korpeoglu, S. Kumar, and K. Achan. Inductive representation learning on temporal graphs. ICLR 2020.

[21] B. Yang, W. Yih, X. He, J. Gao, and L. Deng. Embedding Entities and Relations for Learning and Inference in Knowledge Bases. ICLR 2015.

[22] S. Yun, M. Jeong, R. Kim, J. Kang, and H. J. Kim. Graph Transformer Networks. NeurIPS 2019.

[23] C. Zhang, D. Song, C. Huang, A. Swami, and N. V. Chawla. HetGNN: Heterogeneous Graph Neural Network. KDD 2019.