Bi-Level Attention Neural Architectures for Relational Data

Roshni G. Iyer [1] **Wei Wang** [1] **Yizhou Sun** [1]

1. Abstract

We present Bi-Level Attention-Based Relational Graph Convolutional Networks (BR-GCN), unique neural network architectures that utilize masked self-attentional layers with relational graph convolutions, to effectively operate on highly multi-relational data. BR-GCN models use bi-level attention to learn node embeddings through (1) node-level attention, and (2) relation-level attention. BR-GCN's node-level self-attentional layers use intra-relational graph interactions to learn relation-specific node embeddings using a weighted aggregation of neighborhood features in a sparse subgraph region. BR-GCN's relation-level self-attentional layers use inter-relational graph interactions to learn the final node embeddings using a weighted aggregation of relation-specific node embeddings. BR-GCN's bi-level attention mechanism extends Transformer-based multiplicative attention from the natural language processing (NLP) domain, and Graph Attention Networks (GAT)-based attention, to large-scale heterogeneous graphs (HGs). On node classification, BR-GCN outperforms baselines from 0.29% to 14.95% as a stand-alone model, and on link prediction, BR-GCN outperforms baselines from 0.02% to 7.40% as an auto-encoder model. We also conduct ablation studies to evaluate the quality of BR-GCN's relation-level attention and discuss how its learning of graph structure may be transferred to enrich other Graph Neural Networks (GNNs). Through various experiments, we show that BR-GCN's attention mechanism is both scalable and more effective in learning compared to state-of-the-art GNNs.

2. BR-GCN Architecture

We define directed and labeled HGs as utilized in this work as $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{R})$ where nodes are $v_i \in \mathcal{V}$ and belong to possibly different entities, and edges are $(v_i, r, v_j) \in \mathcal{E}$ with $r \in \mathcal{R}$ and belong to possibly different relation types.

[1]University of California, Los Angeles, USA. Correspondence to: Roshni G. Iyer <roshnigiyer@cs.ucla.edu>, Wei Wang <weiwang@cs.ucla.edu>, Yizhou Sun <yzsun@cs.ucla.edu>.

Workshop on Graph Representation Learning and Beyond (GRL+) of *International Conference on Machine Learning*, 2020.

2.1. Generalized Framework for Bi-Level Attention

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 the HG which thereby captures more information about graph components than a uni-level attention mechanism is capable of. Eq. 1 describes the generalized bi-level attention framework to compute embeddings for node i in the $(l+1)$-th layer:

$$\mathbf{h}_i^{(l+1)} = \mathrm{AGG}(\Big\{\mathbf{f}(a(r))^T \mathrm{AGG}(\Big\{ \mathbf{g}(a(\mathrm{edge}_{i,j}|r))|j \in N_i^r \Big\})|r \in R_i \Big\}) \quad (1)$$

where $\mathbf{g}(a(\mathrm{edge}_{i,j}|r))$ is a vector-output function of the node-level attention that provides a relation-specific embedding summary which is aggregated, $\mathrm{AGG}(\cdot)$, over edges $\mathrm{edge}_{i,j}$ that belong to the neighborhood context of nodes $j \in N_i^r$, and $\mathbf{f}(a(r))$ is a vector-output function of the relation-level attention that are weighted relation-specific embeddings which are aggregated over relations in the neighborhood context to form the final node embedding. See Table 1 for explanations of variables.

In Sections 2.2 and 2.3, we propose a novel semi-supervised attention-based GCN model, BR-GCN, for multi-relational HGs. BR-GCN models use bi-level attention to learn (1) node-level attention, followed by (2) relation-level attention. BR-GCN's attention mechanism is summarized in Figure 1. BR-GCN models use L stacked layers, each of which is defined through Eq. 1, where the previous layer's output is input to the next layer. The initialized input can be chosen as a unique one-hot vector for each node if no other features are present. The model also supports pre-defined features. BR-GCN's source code, pseudo-code, and a walkthrough example of its attention mechanism is in the Appendix.

2.2. Node-level Attention

Node-level attention distinguishes the 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. In HGs, neighbor nodes may belong to different feature spaces, so the features of all nodes are projected to the same feature space to enable node-level attention to

Table 1. Variables (Var) and Explanations. Table on the left column corresponds to node-level attention. Table on the right column corresponds to relation-level attention.

Var	Explanation	Var	Explanation
r	Relation on node edge	$\mathbf{W}_{1,r}$	Projection weight matrix for z_i^r
R_i	Set of relations on edge of node i	$\mathbf{W}_{2,r}$	Projection weight matrix for z_i^r
$\mathbf{h}_i^{(l)}$	Node i features at layer l	$\mathbf{W}_{3,r}$	Projection weight matrix for z_i^r
$e_{i,j}^r$	GAT-based attention for (i,j)	$\mathbf{q}_{r,i} \in Q_r$	Transformer-based query matrix row
$\text{edge}_{i,j}$	Edge between node i and node j	$\mathbf{k}_{r,i} \in K_r$	Transformer-based key matrix row
$\mathbf{a}_r$	Relation-specific attention vector	$\mathbf{v}_{r,i} \in V_r$	Transformer-based value matrix row
$\gamma_{i,j}^r$	Relation-specific weight for (i,j)	$\mathbf{W}_i$	Weight matrix for $h_i^{(l)}$
N_i^r	Set of relation-specific node neighbors	ψ_i^r	Importance of relation r for node i
$\mathbf{z}_i^r$	Relation-specific node embedding	$\boldsymbol{\delta}_i^r$	Attended relation-specific embedding

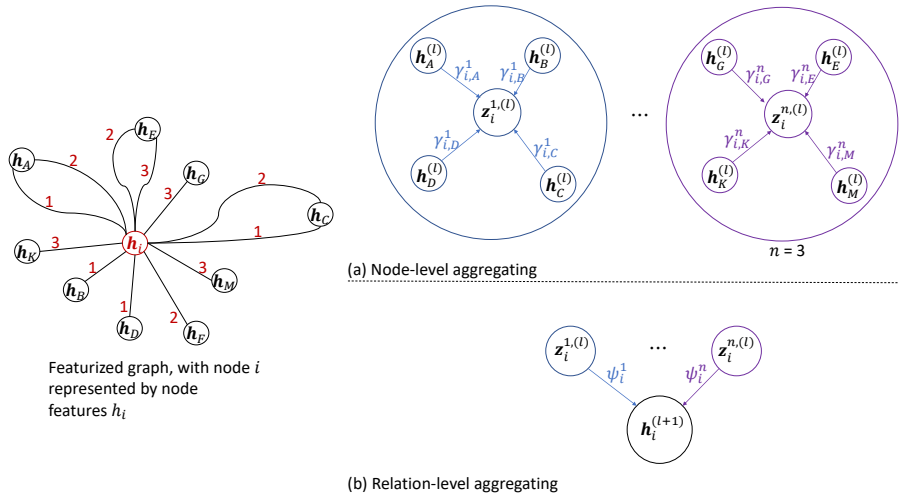

Figure 1. Bi-level attention visualization. (a) Node-level aggregating: A node's features in the $(l+1)$-th layer is a weighted combination of relation-specific embeddings of the node, $\mathbf{z}_i^r$. (b) Relation-level aggregating: Relation-level attention is learned through multiplicative attention using neighborhood relational similarity to other relations to determine the relation's relative importance.

handle arbitrary node types. BR-GCN's node-level attention uses additive attention similar to GAT (11), but overcomes GAT's limitation by extending the attention to HGs. The self-attention for pair (i,j) for node j's importance to node i for relation r is defined as:

$$e_{i,j}^r = a(\text{edge}_{i,j}|r) = att_{node}(\mathbf{h}_i^{(l)}, \mathbf{h}_j^{(l)}, r) \tag{2}$$

$$= \text{LeakyReLU}(\mathbf{a}_r^{T^{(l)}} \left[\mathbf{h}_i^{(l)} \parallel \mathbf{h}_j^{(l)}\right]) \tag{3}$$

For a specific relation r, $att_{node}(\cdot)$ is shared for all node pairs, so that each node is influenced by its neighborhood context. $e_{i,j}^r$ is asymmetric since the importance of node j to node i may be different from the importance of node i to node j. $\mathbf{a}_r^T$ attends over the concatenated, $||$, node features of nodes i and j with an applied $\text{LeakyReLU}(\cdot)$ activation.

By restricting the attention to within the relation-specific neighborhood context of nodes $j \in N_i^r$, sparsity structural information is injected into the model through masked self-attentional layers. A $\text{softmax}(\cdot)$ activation is then applied to normalize each node-pair attention weight:

$$\gamma_{i,j}^r = \text{softmax}(e_{i,j}^r) \tag{4}$$

$$= \frac{\exp(\text{LeakyReLU}(\mathbf{a}_r^{T^{(l)}} \left[\mathbf{h}_i^{(l)} \parallel \mathbf{h}_j^{(l)}\right]))}{\sum_{k \in N_i^r} \exp(\text{LeakyReLU}(\mathbf{a}_r^{T^{(l)}} \left[\mathbf{h}_i^{(l)} \parallel \mathbf{h}_k^{(l)}\right]))} \tag{5}$$

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

$$\mathbf{z}_i^r = \sum_{j \in N_i^r} [\mathbf{g}(a(\text{edge}_{i,j}|r))] = \sum_{j \in N_i^r} [\gamma_{i,j}^r \mathbf{h}_j^{(l)}] \tag{6}$$

where $\mathbf{z}_i^r$ serves as a summary of relation r for node i.

2.3. Relation-level Attention

Relation-level attention distinguishes the different roles of relations in the neighborhood context for learning more comprehensive node embeddings. In HGs, different relations may play different roles of importance for a node i, in addition to its relation-specific neighbor nodes. As such, we learn relation-level attention to better fuse node i's relation-specific node embeddings. We extend Transformer-based multiplicative attention to the HG domain to learn relation-level attention by capturing the importance of a relation r based on how similar it is to the other relations in the local neighborhood context. By restricting the set of relations to the local neighborhood, $r \in R_i$, we utilize multiplicative attention. Node i's relation-specific Transformer-based query vector $\mathbf{q}_{r,i}$, key vector $\mathbf{k}_{r,i}$, and value vector $\mathbf{v}_{r,i}$ are computed as follows:

$$\mathbf{q}_{r,i}; \mathbf{k}_{r,i}; \mathbf{v}_{r,i} = \mathbf{W}_{1,r}\mathbf{z}_i^r; \mathbf{W}_{2,r}\mathbf{z}_i^r; \mathbf{W}_{3,r}\mathbf{z}_i^r \tag{7}$$

where $\mathbf{z}_i^r$ is projected onto the learnable weight matrices $\mathbf{W}_{1,r}, \mathbf{W}_{2,r}, \mathbf{W}_{3,r}$.

The relation-level attention for relation pairs (r, r') are computed by iterating over relations in the neighborhood context, $r' \in R_i$. The importance of relation r' of node i is denoted as follows with Eq. 8 capturing relation similarity:

$$\psi_i^r = a(r) = att_{relation}(\mathbf{z}_i^r, r) = \sum_{r' \in R_i} \mathbf{q}_{r,i}^T \mathbf{k}_{r',i} \tag{8}$$

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 node i's final embedding.

Similar to Relational Graph Convolutional Networks (R-GCN) (8), to enable the representation of a node to be informed by its representation in previous layers, we add a self-connection of a special relation type to each node, which is projected onto $\mathbf{W}_i$, and aggregated to the attended relation-specific embedding, $\psi_i^r \mathbf{v}_{r',i}$. A $\mathrm{softmax}(\cdot)$ activation is then applied to normalize these embeddings.

$$\boldsymbol{\delta}_i^r = \mathrm{softmax}(\sum_{r' \in R_i} \psi_i^r \mathbf{v}_{r',i} + \boldsymbol{W}_i \boldsymbol{h}_i^{(l)}) \tag{9}$$

The final node embedding for node i is learned with $\mathrm{AGG}(\cdot)$ in Eq. 1 being a weighted summation of the relation-specific embeddings:

$$\boldsymbol{h}_i^{(l+1)} = \sum_{r \in R_i} [\mathbf{f}(a(r))] = \sum_{r \in R_i} [\boldsymbol{\delta}_i^r] \tag{10}$$

Finally, putting the above equation components together, the final node embedding is learned by:

$$h_i^{(l+1)} = \sum_{r \in R_i} \mathrm{softmax}(\sum_{r' \in R_i} \mathbf{q}_{r,i}^T \mathbf{k}_{r',i} \mathbf{v}_{r',i} + \boldsymbol{W}_i \boldsymbol{h}_i^{(l)}) \tag{11}$$

2.4. Justification for BR-GCN's Bi-Level Attention Mechanism

Our approach uses GAT-based attention to learn node-level attention because it is an effective additive attention mechanism, and using multiplicative attention requires the strong assumption that the importance of a node is a function of its similarity to other nodes in its context. There may be relation-specific nodes that are highly similar to each other but require different node-level attentions to be learned. Transformer-based attention is used to learn relation-level attention because it is an effective multiplicative attention mechanism, and since concatenation is not enough to capture relational importance. Relation features are characterized simply by their relation types, whereas node features may have several attributes. As such, the assumption of multiplicative attention is applicable to the relation-level unlike the node-level where it is more difficult to learn the similarity of nodes when many attribute factors are at play. By hierarchically learning node-level and relation-level attention of HGs, BR-GCN models address the limitations of R-GCN and GAT, since bi-level attention captures more information than uni-level attention. By considering the entire HG instead of subgraphs from pre-selected meta-paths as in Heterogeneous Graph Attention Networks (HAN) (13), BR-GCN comprehensively learns different aspects of nodes through the entire set of nodes and relations. Furthermore, by learning relation-level attention using information from the neighborhood context instead of a generic global vector as in HAN, BR-GCN learns a more personalized attention for that relation to construct the final node embedding.

3. Experiments

In this section, we present experiments on node classification and link prediction using benchmark datasets: AIFB (7), MUTAG (7), BGS (7), AM (7), FB15k (1), WN18 (3), and FB15k-237 (9). The models evaluated are: BR-GCN and variant models, HAN, R-GCN and variant models, GAT, and Graph Isomorphism Networks (GIN) (14). Link prediction experiments, which are based on the splits from (2), are detailed in the Appendix. The Appendix also discusses experiments on ablation studies to evaluate the quality of BR-GCN's relation-level attention and explains how its learning of graph structure may be transferred to enrich other GNNs. Furthermore, the Appendix discusses the computational complexity of BR-GCN and primary baseline models, which suggest that BR-GCN is a scalable architecture.

3.1. Node Classification

Node classification is the semi-supervised classification of nodes to entity types. For evaluation consistency against R-GCN and GAT, BR-GCN architectures are implemented using two convolutional layers with the final layer using

Table 2. Node classification test accuracy %. Results are averaged over 10 runs and with benchmark splits from (8). Baseline models are HAN (13), R-GCN (8), GAT (11), and GIN (14). BR-GCN-node and BR-GCN-relation are uni-level attention models such that BR-GCN-node uses BR-GCN's node-level attention, and BR-GCN-relation uses BR-GCN's relation-level attention. BR-GCN's model is described in Section 2. Experiments are run using the PyTorch Geometric framework (4) on an NVIDIA Tesla V100 GPU cluster.

Model	AIFB	MUTAG	BGS	AM
HAN	96.68 ± 0.04	78.46 ± 0.07	86.84 ± 0.21	90.68 ± 0.23
R-GCN	95.83 ± 0.62	73.23 ± 0.48	83.10 ± 0.80	89.29 ± 0.35
GAT	92.50 ± 0.29	66.18 ± 0.00	77.93 ± 0.17	88.52 ± 1.65
GIN	96.18 ± 0.53	78.89 ± 0.09	86.42 ± 0.35	91.33 ± 0.16
BR-GCN-node	96.46 ± 0.13	73.19 ± 0.25	84.23 ± 0.22	89.45 ± 0.02
BR-GCN-relation	95.28 ± 0.23	76.17 ± 0.22	87.53 ± 0.34	90.52 ± 0.18
BR-GCN (ours)	**96.97** ± 0.08	**81.13** ± 0.61	**88.30** ± 0.04	**92.57** ± 0.15

a $\mathrm{softmax}(\cdot)$ activation per node. We optimize BR-GCN using cross-entropy loss for labeled nodes with parameters learned through the Adam Optimizer. We minimize the cross-entropy loss below:

$$L = -\sum_{i \in Y} \sum_{k=1}^{K} t_{ik} \ln h_{ik}^{(L)} \tag{12}$$

with Y being the indices of labeled nodes, $h_{ik}^{(L)}$ being the k-th entry of the i-th labeled node for the L-th layer, and t_{ik} being h_{ik}'s corresponding ground-truth label.

Datasets: To ensure fair comparison against reported results of baseline models, we evaluate BR-GCN on the commonly used node classification heterogeneous datasets in the Resource Description Framework (RDF) format (6; 7): AIFB, MUTAG, BGS, and AM. Consistent with (8), relations used to create entity labels: *employs* and *affiliation* (AIFB), *isMutagenic* (MUTAG), *hasLithogenesis* (BGS), and *objectCategory* and *material* (AM) are removed. The reader is referred to the Appendix for dataset details.

Results: The experimental results of Table 2 are reported by using the benchmark splits from (8). We evaluate against state-of-the-art GNNs with two hidden layers for fairness of comparison: HAN (13), R-GCN (8), GAT (11), and GIN (14), in addition to BR-GCN variant models: BR-GCN-node being a uni-level attention model using BR-GCN's node-level attention, BR-GCN-relation being a uni-level attention model using BR-GCN's relation-level attention, and BR-GCN is our model, described in Section 2. We use the graph defined by all meta-paths for evaluating HAN, since the authors do not indicate how the pre-defined meta-paths are selected. BR-GCN outperforms prior state-of-the-art models on the benchmark datasets and in comparison to BR-GCN-node and BR-GCN-relation. Furthermore, HAN, another bi-level attention model, achieves test accuracy that is generally higher on all datasets compared to the uni-level attention models of GAT, BR-GCN-node, and BR-GCN-relation. This suggests that bi-level attention may be more effective than uni-level attention. While there is insufficient evidence to determine whether relation-level or node-level attention is more important for learning embeddings, bi-level attention seems to leverage the information of both uni-level attentions using an effective weighted aggregation. Hyperparameter values for BR-GCN models are reported in the Appendix.

4. Conclusions and Future Work

We present a generalized framework for computing bi-level attention and discuss our novel bi-level attention model, BR-GCN. Our best BR-GCN model effectively leverages attention and graph sparsity as suggested by experiment results against state-of-the-art baseline models of HAN, R-GCN, GAT, GIN, BR-GCN-node, and BR-GCN-relation. On node classification, BR-GCN outperforms baselines from 0.29% to 14.95%, and on link prediction, BR-GCN outperforms baselines from 0.02% to 7.40%. As GNNs and attention-based neural architectures have been widely applied to numerous domains, by advancing these architectures, BR-GCN also has the potential to further advance knowledge discovery in these domains. Domain applications of BR-GCN include social networks, medical informatics, software development, and natural language processing. For future work, we plan to investigate applying BR-GCN to enhance the inference tasks of question-answering for text and software, graph similarity detection, as well as to benefit domain-specific tasks such as code recommendation and bug detection in software development, and disease prediction and prognosis in medical informatics.

Acknowledgements We would like to thank Yunsheng Bai for helpful discussions, comments, and corrections. This project is supported by the National Science Foundation (NSF) Research Traineeship program, through the NSF **M**od**E**ling and u**N**ders**T**anding human behavi**OR** (MENTOR) Fellowship (Grant No. #1829071).

References

[1] A. Bordes, N. Usunier, A. Garcia-Duran, J. Weston, and O. Yakhnenko. Translating embeddings for modeling multi-relational data. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26*, pages 2787–2795. Curran Associates, Inc., 2013.

[2] A. Bordes, N. Usunier, A. Garcia-Duran, J. Weston, and O. Yakhnenko. Translating embeddings for modeling multi-relational data. In *Advances in Neural Information Processing Systems*, NIPS 2013, 2013.

[3] T. Dettmers, P. Minervini, P. Stenetorp, and S. Riedel. Convolutional 2d knowledge graph embeddings. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[4] M. Fey and J. E. Lenssen. Fast graph representation learning with PyTorch Geometric. In *ICLR Workshop on Representation Learning on Graphs and Manifolds*, 2019.

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

[6] P. Ristoski, G. K. D. de Vries, and H. Paulheim. A collection of benchmark datasets for systematic evaluations of machine learning on the semantic web. In *In International Semantic Web Conference*, Springer, page 186–194, 2016.

[7] P. Ristoski and H. Paulheim. Rdf2vec: Rdf graph embeddings for data mining. In *In International Semantic Web Conference*, page 498–514. Springer, 2016.

[8] 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.

[9] 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*, pages 57–66, 2015.

[10] T. Trouillon, J. Welbl, S. Riedel, E. Gaussier, and G. Bouchard. Complex embeddings for simple link prediction. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*, ICML'16, page 2071–2080. JMLR.org, 2016.

[11] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio. Graph attention networks. In *International Conference on Learning Representations*, ICLR'18, 2018.

[12] M. Wang, L. Yu, D. Zheng, Q. Gan, Y. Gai, Z. Ye, M. Li, J. Zhou, Q. Huang, C. Ma, Z. Huang, Q. Guo, H. Zhang, H. Lin, J. Zhao, J. Li, A. J. Smola, and Z. Zhang. Deep graph library: Towards efficient and scalable deep learning on graphs. *ICLR Workshop on Representation Learning on Graphs and Manifolds*, 2019.

[13] X. Wang, H. Ji, C. Shi, B. Wang, P. Cui, P. Yu, and Y. Ye. Heterogeneous graph attention network. In *Proceedings of the 30th International Conference on World Wide Web*, WWW '19. ACM, 2019.

[14] K. Xu, W. Hu, J. Leskovec, and S. Jegelka. How powerful are graph neural networks? In *International Conference on Learning Representations*, ICLR'19, 2019.

[15] B. Yang, W. tau Yih, X. He, J. Gao, and L. Deng. Embedding entities and relations for learning and inference in knowledge bases. *ICLR2015*, abs/1412.6575, 2014.

APPENDIX

A. Source Code and Environment

The source code for our work can be found at: https://github.com/roshnigiyer/BR-GCN

Details for running the source code are present in the README.md file, and environment details are in the requirements.txt file.

The overall process of BR-GCN is described in Algorithm 1. A walkthrough example of BR-GCN's architecture is detailed in Figure 2.

B. Dataset Description

We evaluate node classification on four datasets: AIFB, MUTAG, BGS, and AM. We evaluate link prediction on three datasets: FB15k, WN18, and FB15k-237. We evaluate ablation studies on the AM dataset. The description of these datasets are found below.

AIFB The AIFB dataset is a social network dataset pertaining to the Institute for Applied Informatics and Formal Description Methods at the Karlsruhe Institute of Technology. It includes the relationships between persons (e.g., Professors, Students), research topics, projects, publications etc. AIFB has 8,285 entities, 45 relations, and 29,043 edges. 176 of the entities have labels and are to be classified into 4 classes.

MUTAG The MUTAG dataset is a biological dataset that contains information about molecules that are potentially carcinogenic. MUTAG has 23,644 entities, 23 relations, and 74,227 edges. 340 of the entities have labels and are to be classified into 2 classes.

BGS The BGS dataset is a geological dataset that contains information about rock units. BGS has 333,845 entities, 103 relations, and 916,199 edges. 146 of the entities have labels and are to be classified into 2 classes.

AM The AM dataset contains information about artifacts from the Amsterdam Museum. AM has 1,666,764 entities, 133 relations, and 5,988,321 edges. 1,000 of the entities have labels and are to be classified into 11 classes.

FB15k The FB15k dataset is a subset of Freebase, a highly multi-relational collaborative HG. FB15k has 14,951 entities, and 1,345 relations. 483,142 edges are used for training, 50,000 edges are used for validation, and 59,071 edges are used for testing.

WN18 The WN18 dataset is a subset of WordNet, a multi-relational lexical HG for the English language. WN18 has 40,943 entities, and 18 relations. 141,442 edges are used for training, 5,000 edges are used for validation, and 5,000 edges are used for testing.

FB15k-237 The FB15k-237 dataset is a reduced dataset of FB15k with inverse triplet pairs removed. Triplet and inverse triplet pairs are denoted as: $t = (e_1, r, e_2)$ and $t^{-1} = (e_2, r^{-1}, e_1)$. FB15k-237 has 14,541 entities, and 237 relations. 272,115 edges are used for training, 17,535 edges are used for validation, and 20,466 edges are used for testing.

C. Experiments

In this section we describe further experimental details for BR-GCN and baseline models for node classification, link prediction, and ablation studies.

C.1. Model Hyperparameters

We proceed to summarize the hyperparameter values used for BR-GCN models for node classification experiments. The hyperparameters used for BR-GCN models for link prediction experiments and ablation studies are the same as the hyperparameter values for BR-GCN models for the AM dataset.

The learning rate for BR-GCN-node for the datasets of AIFB, MUTAG, BGS, and AM are $\{0.010, 0.001, 0.001, 0.001\}$ respectively, with the $l2$ penalty as 0 and the # hidden units as 16 for all datasets. The # basis functions are $\{6, 1, 0, 2\}$ respectively, and the # epochs are $\{70, 90, 70, 80\}$ respectively. The dropout rate is $\{0.6, 0.4, 0.0, 0.6\}$ respectively. The negative slope for $\mathrm{LeakyReLU}(\cdot)$ activation is $\{0.6, 0.8, 0.4, 0.8\}$ respectively.

The learning rate for BR-GCN-relation for the datasets of AIFB, MUTAG, BGS, and AM are $\{0.010, 0.010, 0.050, 0.001\}$ respectively, with the $l2$ penalty as $\{0, 0, 0, 5 \times 10^{-4}\}$ respectively. The # hidden units are 16 for all datasets. The # basis functions are $\{2, 0, 4, 2\}$ respectively, and the # epochs are $\{70, 75, 85, 85\}$ respectively.

The learning rate for BR-GCN for the datasets of AIFB, MUTAG, BGS, and AM are $\{0.050, 0.010, 0.005, 0.010\}$ respectively, with the $l2$ penalty being $\{0, 5 \times 10^{-4}, 0, 0\}$ respectively. The # hidden units are 16 for all datasets. The # basis functions are $\{0, 0, 1, 0\}$ respectively, and the # epochs are $\{85, 90, 95, 100\}$ respectively. The dropout rate is $\{0.4, 0.2, 0.6, 0.6\}$ respectively. The negative slope for $\mathrm{LeakyReLU}(\cdot)$ activation is $\{0.2, 0, 0.4, 0\}$ respectively.

C.2. Link Prediction

Denote $\mathcal{E}'$ to be the incomplete subset of edges $\mathcal{E}$ in the HG. Link prediction involves assigning confidence scores α to *(h,r,t)* to determine how likely those predicted edges belong to $\mathcal{E}$, or the true relations. We construct graph auto-encoder models, with BR-GCN used as the encoder, and HG embedding models used as the decoder, for this task. We

Algorithm 1: The BR-GCN model.

Input : The labeled heterogeneous graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{R})$,
The nodes $v_i \in \mathcal{V}$, belonging to possibly different entities,
The edges $(v_i, r, v_j) \in \mathcal{E}$,
The relations $r \in \mathcal{R}$, belonging to possibly different relation types.

Output : The final embedding $\mathbf{h}_i^{(l+1)}$,
The node-level attention weight γ ,
The semantic-level attention weight ψ .

for $r \in R_i$ **do**
  for $v_i \in \mathcal{V}$ **do**
    Featurized type-specific transformation $\mathbf{h}_i^{(l)} \leftarrow v_i$;
    Find the relation-specific node neighbors $\mathcal{N}_i^r$;
    for $j \in \mathcal{N}_i^r$ **do**
      Calculate the weight coefficient $\gamma_{i,j}^r$;
    end
    Calculate the semantic-specific node embedding $\mathbf{z}_i^r \leftarrow \sum_{j \in N_i^r} [\gamma_{i,j}^r \mathbf{h}_j^{(l)}]$;
  end
  Calculate the relation weight ψ_i^r;
  Calculate the attended relation-specific embedding $\boldsymbol{\delta}_i^r$;
  Fuse the semantic-specific embedding $\mathbf{h}_i^{(l+1)} \leftarrow \sum_{r \in R_i} [\boldsymbol{\delta}_i^r]$;
end
Calculate Cross-Entropy $L = -\sum_{i \in Y} \sum_{k=1}^{K} t_{ik} \ln h_{ik}^{(L)}$;
Back propagation and update parameters in BR-GCN;
return $\mathbf{h}_i^{(l+1)}, \gamma, \psi$.

use the approach of (8) to train and evaluate our model. As such, our models use negative sampling, with ω being the number of negative samples per observed example. Negative sampling is constructed by randomly corrupting entities of the observed example. We use cross-entropy for the loss function, where observable triples are scored higher than negative triples, with parameters learned using the Adam Optimizer. The loss function is calculated as:

$$L = c \times \sum_{(h,r,t,y) \in T} y \log l(\alpha) + (1-y) \log(1 - l(\alpha)) \quad (13)$$

$$c = -\frac{1}{(1+\omega)|\mathcal{E}'|} \quad (14)$$

$$y = \begin{cases} 1 & \text{if positive triples} \\ 0 & \text{if negative triples} \end{cases} \quad (15)$$

with T being the total set of real and corrupted triples, α being the confidence score, $\mathcal{E}'$ being the incomplete subset of edges in the HG, ω being the number of negative samples per observed example, y being an indicator variable defined in Eq. 15, and $l(\cdot)$ being the logistic sigmoid activation.

Datasets: To ensure fair comparison against reported results of baseline models, we evaluate BR-GCN on the commonly used link prediction heterogeneous datasets of FB15k (1), WN18 (3), and FB15k-237 (9), a reduced version of FB15k. Results are based on splits from (2). See Section B for dataset descriptions.

Results: We use mean reciprocal rank (MRR) and Hits @ n as evaluation metrics, computed in a raw and filtered setting. The reader is referred to (2) for details. The same number of negative samples, $w = 1$, are utilized to make the datasets comparable. We evaluate BR-GCN and R-GCN as standalone models and as autoencoder models as in (8), using the following HG embedding models as decoders: DistMult (D) (15), TransE (T) (2), HolE (H) (5), and ComplEx (C) (10). BR-GCN$_{embedding}$ is an ensemble model with a trained BR-GCN model and a separately trained embedding model: $\alpha_{\text{BR-GCN}_{embedding}} = \beta \times \alpha_{\text{BR-GCN}} + (1-\beta) \times \alpha_{embedding}$, $\beta = 0.4$. Similarly, for R-GCN$_{embedding}$. See Table 3 for results on FB15k and WN18, and Table 4 for results on FB15k-237. The best BR-GCN models outperform R-GCN models. In general, the ComplEx model achieves promising results perhaps since it explicitly models asymmetry in relations. The FB15k and WN18 datasets are well-connected and therefore nodes can learn important information from their neighborhood contexts. FB15k-237, however, does not contain as much important local information, so HG embedding models are not as useful when being coupled with R-GCN and BR-GCN

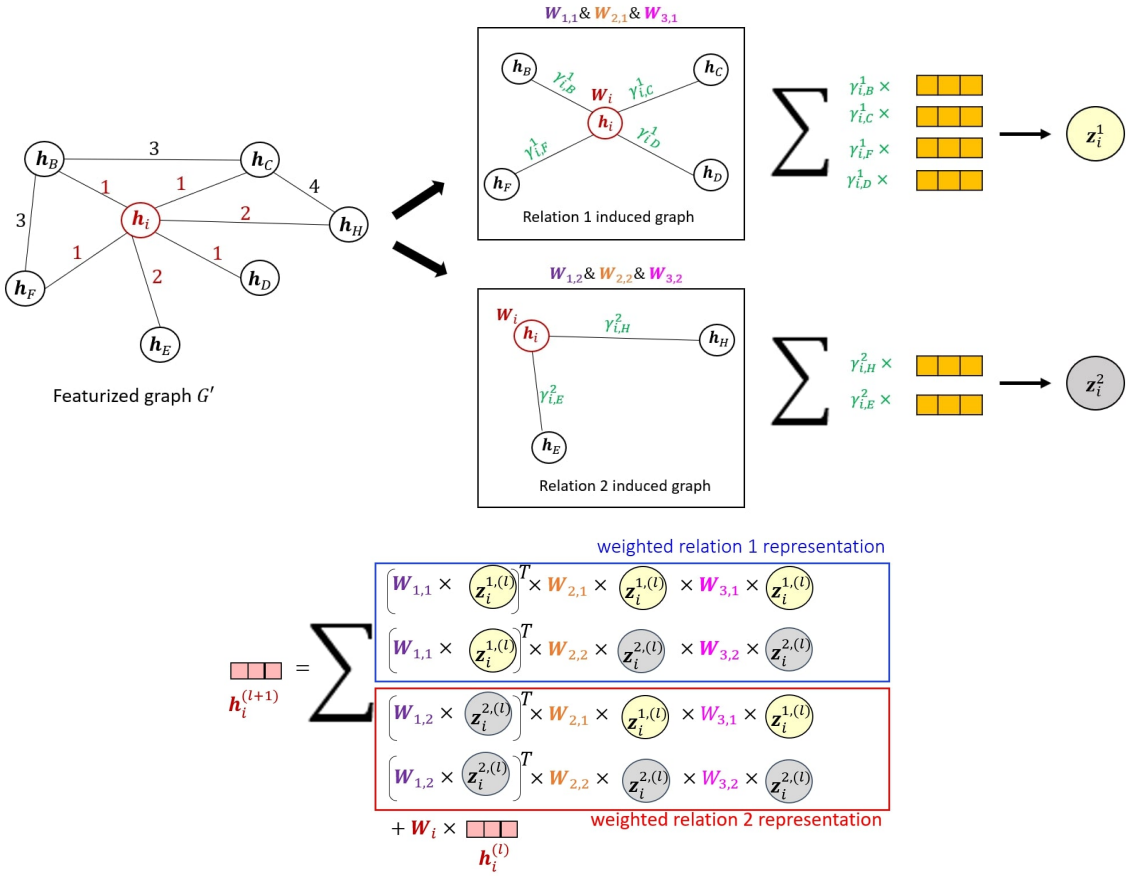

Figure 2. Bi-level attention visualization walkthrough example. The featurized graph G' of node i denotes relations in the neighborhood context for node i in red. An induced graph for each relation in the neighborhood context is created and node-level attention for each induced graph is learned to form relation-specific embeddings $\mathbf{z}_i^1$, and $\mathbf{z}_i^2$ for relations 1 and 2. The relation-level attention mechanism to learn the final node embedding $\mathbf{h}_i^{(l+1)}$ and the corresponding weighted relation representations of relations 1 and 2 are shown.

Table 3. Link prediction results on the FB15k and WN18 datasets for mean reciprocal rank (MRR), and Hits @ n metrics. We evaluate BR-GCN and R-GCN as standalone models and as autoencoder models using HG embedding models as decoders: DistMult (D) (15), TransE (T) (2), HolE (H) (5), and ComplEx (C) (10). BR-GCN$_{embedding}$ is an ensemble model with a trained BR-GCN model and a separately trained embedding model. Similarly for R-GCN$_{embedding}$. Experiments are run using the Deep Graph Library (DGL) (12), and the PyTorch Geometric framework (4), on an NVIDIA Tesla V100 GPU cluster. Averages are reported for 10 runs.

	FB15k					WN18				
	MRR		Hits @			MRR		Hits @		
Model	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
BR-GCN	0.255	0.662	0.564	0.748	0.829	0.557	0.814	0.691	0.928	0.956
R-GCN$_D$	0.262	0.696	0.601	0.760	0.842	0.561	0.819	0.697	0.929	0.964
R-GCN$_T$	0.252	0.651	0.543	0.738	0.828	0.554	0.815	0.681	0.928	0.956
R-GCN$_H$	0.257	0.659	0.556	0.744	0.839	**0.567**	0.822	0.699	0.933	0.966
R-GCN$_C$	0.260	0.712	0.629	0.771	0.845	0.565	0.822	0.701	0.933	0.965
BR-GCN$_D$	**0.265**	0.703	0.646	0.782	**0.851**	0.564	0.825	0.700	**0.934**	0.966
BR-GCN$_T$	0.254	0.655	0.544	0.740	0.829	0.559	0.815	0.684	0.928	0.960
BR-GCN$_H$	0.258	0.661	0.560	0.748	0.840	**0.567**	0.823	**0.702**	**0.934**	**0.969**
BR-GCN$_C$	0.264	**0.725**	**0.668**	**0.785**	**0.851**	0.566	**0.829**	**0.702**	**0.934**	0.966

Table 4. Link prediction results on the FB15k-237 dataset, a reduced version of FB15k with problematic inverse relation pairs removed for mean reciprocal rank (MRR), and Hits @ n metrics. We evaluate BR-GCN auto-encoder models against R-GCN auto-encoder models (8) using the following HG embedding models as decoders: DistMult (D) (15), TransE (T) (2), HolE (H) (5), and ComplEx (C) (10). R-GCN$_{embedding}$ denotes R-GCN as the encoder combined with the specific embedding model as the decoder. Similarly, for BR-GCN$_{embedding}$. Experiments are run using the Deep Graph Library (DGL) (12), and the PyTorch Geometric framework (4).

	MRR		Hits @		
Model	Raw	Filtered	1	3	10
R-GCN	0.158	0.248	0.153	0.258	0.414
BR-GCN	0.160	0.249	0.160	0.261	0.418
R-GCN$_D$	0.156	0.249	0.151	0.264	0.417
R-GCN$_T$	0.161	0.258	0.159	0.274	0.421
R-GCN$_H$	0.159	0.257	0.156	0.272	0.420
R-GCN$_C$	0.158	0.255	0.152	0.268	0.419
BR-GCN$_D$	0.157	0.251	0.255	0.265	0.419
BR-GCN$_T$	**0.163**	**0.261**	**0.164**	**0.275**	**0.423**
BR-GCN$_H$	0.161	0.258	0.158	0.272	0.422
BR-GCN$_C$	0.160	0.259	0.153	0.268	0.420

compared to in the FB15k and WN18 datasets. The TransE embedding model seems to be most effective in FB15k-237 perhaps because the embedding relationship between the head and tail entities are linear. Refer to Section C.1 for details on model hyperparameters.

C.3. Ablation Studies

We conduct experiments to determine the quality of relation-level attention and graph-structure learned from BR-GCN. We modify the AM dataset to contain the following types of relations, each with cummulative 10% splits: (1) relations considered by random selection, (2) relations with the highest corresponding relation-level attention weights learned by BR-GCN, and (3) relations with the lowest corresponding relation-level attention weights learned by BR-GCN. The attention of relations for BR-GCN that we utilize is a function of the learned relation-level attention matrix ψ^r, which is computed by averaging the matrix elements: $average(\psi^r)$. The experiment figures for the primary baseline models of HAN, R-GCN, GAT, and BR-GCN models are detailed in the Figure 3.

As shown in Figure 3, the AM graph produced by the highest attention relation results in the highest test accuracy compared to the other metrics in all models. The AM graph produced by the lowest attention relation results in the lowest test accuracy compared to the other metrics in all models.

Table 5. Test Accuracy (%) achieved from the top $x\%$ attention relations for the AM dataset. The attention of relations is a function of the learned relation-level attention matrix ψ^r computed by BR-GCN, defined as $average(\psi^r)$.

Model	Top 10%	Top 50%	Top 100%
HAN	84.32	88.55	90.68
R-GCN	83.61	88.47	89.29
GAT	75.25	81.90	88.52
BR-GCN-node	85.38	87.63	89.45
BR-GCN-relation	84.97	88.89	90.52
BR-GCN (ours)	**85.91**	**89.19**	**92.55**

The test accuracy for the AM graph produced by relations randomly selected are as expected in between the test accuracies of the other two metrics. Since models that do not learn relation-level attention (R-GCN, GAT, BR-GCN-node) still benefit from the graph structure identified by the highest attention relations, this shows that the learning of graph structure identified by relation-level attention may be transferable. Furthermore, as larger % of relations are considered, all models benefit from learning better graph structure and some models may also learn better attention, resulting in even higher test accuracies.

Table 5 summarizes the model test accuracies achieved from the AM graph produced by the highest attention relation metric for the top 10%, 50%, and 100% of relations. BR-GCN produces the highest test accuracy in all categories, showing that it may learn the best from graph structure and relation-level attention compared to the primary baseline models.

D. Theoretical Analysis

Table 6. Runtime complexity for BR-GCN and primary baseline models for computing $\mathbf{h}_i^{(l+1)}$.

Model	Runtime Complexity
HAN	$\mathrm{O}(P \cdot d^{(l)}) + \mathrm{O}(\lvert V_P\rvert \cdot d^{3(l)})$
R-GCN	$O(\lvert R_i\rvert \cdot \lvert N_i^r\rvert^2 \cdot d^{(l+1)} \cdot d^{(l)})$
GAT	$O(K \cdot \lvert N_i^r\rvert \cdot d^{2(l)})$
BR-GCN-node	$O(\lvert R_i\rvert \cdot \lvert N_i^r\rvert \cdot d^{(l+1)} \cdot d^{(l)})$
BR-GCN-relation	$O(\lvert R_i\rvert \cdot \lvert N_i^r\rvert \cdot d^{(l+1)} \cdot d^{(l)})$
BR-GCN (ours)	$O(d^{2(l)}) + O(\lvert R_i\rvert \cdot \lvert N_i^r\rvert \cdot d^{(l)})$

Table 6 shows the runtime complexity for BR-GCN and the primary baseline models, and Table 7 shows the memory complexity for BR-GCN and the primary baseline models. $d^{(l)}$ and $d^{(l+1)}$ are the node dimensions at the l-th and $(l+1)$-th layers respectively, R_i is the set of relations in the neighborhood context, R is the set of relations in the HG, N_i^r is the set of nodes in the neighborhood context, K is the number of attention heads, P is the set of pre-defined meta-paths used in HAN, and V_p is the set of nodes contained in the pre-defined meta-paths. The reader is referred to (8; 11; 13) for descriptions of the baseline GNN models. BR-GCN is comparable in runtime and memory complexity to the state-of-the-art neural architectures of HAN, R-GCN, and GAT, suggesting that it is a scalable architecture.

Table 7. Memory complexity for BR-GCN and primary baseline models for computing $\mathbf{h}_i^{(l+1)}$.

Model	Memory Complexity
HAN	$O(d^{(l)} \times (K + \lvert R\rvert))$
R-GCN	$O(\lvert R_i\rvert \cdot d^{(l+1)} \cdot d^{(l)})$
GAT	$O(d^{(l)} \times (K + \lvert R_i\rvert))$
BR-GCN-node	$O(\lvert R_i\rvert \cdot d^{(l+1)} \cdot d^{(l)})$
BR-GCN-relation	$O(d^{(l+1)})$
BR-GCN (ours)	$O(\lvert R_i\rvert \cdot d^{(l)})$

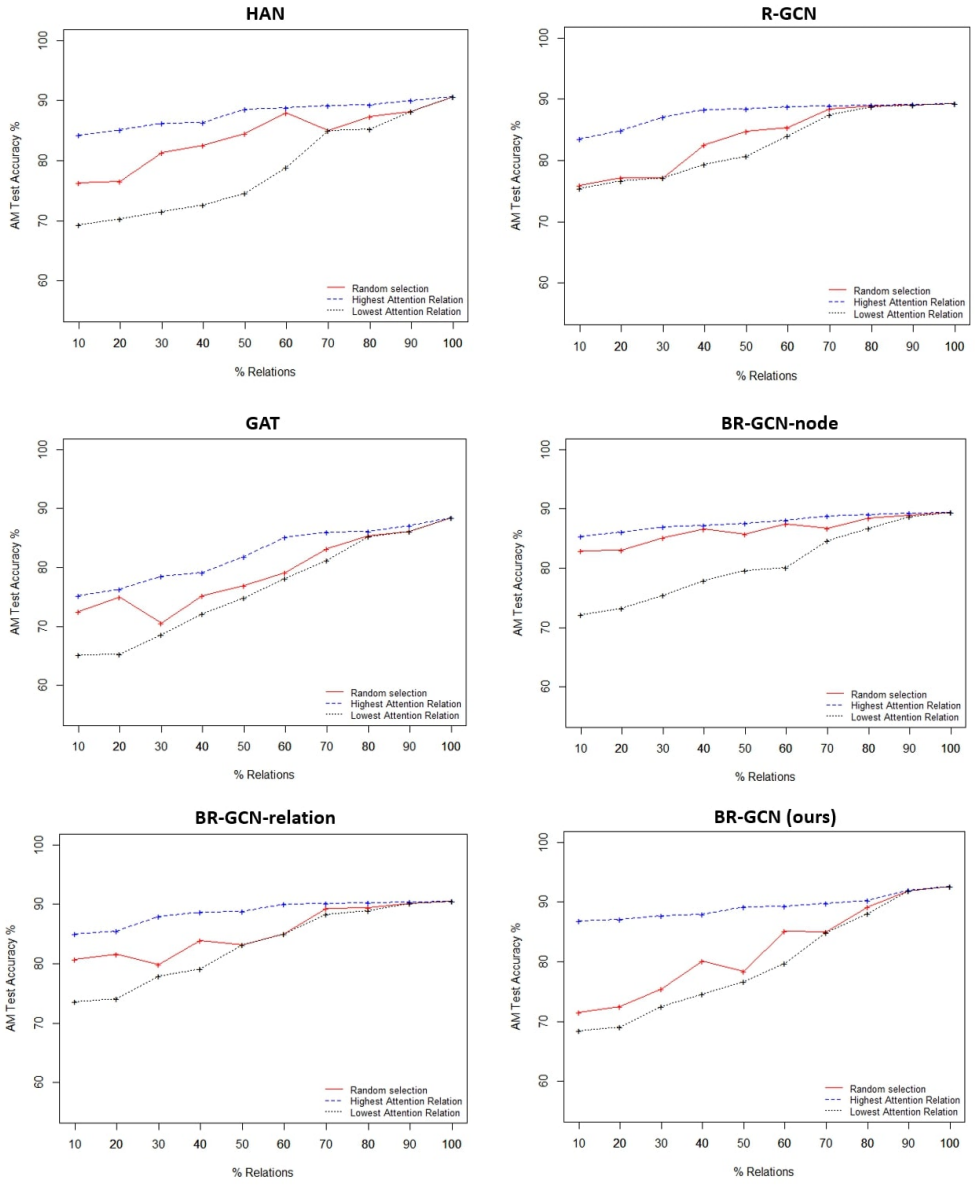

Figure 3. Ablation study experiments on primary baseline models of HAN, R-GCN, GAT, and BR-GCN models to evaluate learned relation-level attention and graph structure of BR-GCN.