# An Attention-Based Graph Neural Network for Heterogeneous Structural Learning

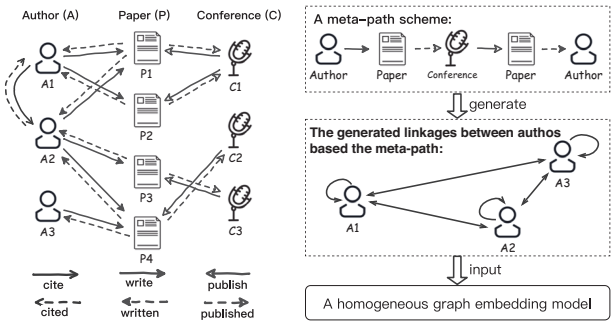
**Huiting Hong,**[1] **Hantao Guo,**[2*] **Yucheng Lin,**[1] **Xiaoqing Yang,**[1] **Zang Li,**[1†] **Jieping Ye**[1]
[1]AI Labs, Didi Chuxing, Beijing, China, [2]Peking University, Beijing, China
{honghuiting, yangxiaoqing, lizang, yejieping, linyucheng}@didiglobal.com, guohantao@pku.edu.cn

## Abstract

In this paper, we focus on graph representation learning of heterogeneous information network (HIN), in which various types of vertices are connected by various types of relations. Most of the existing methods conducted on HIN revise homogeneous graph embedding models via meta-paths to learn low-dimensional vector space of HIN. In this paper, we propose a novel Heterogeneous Graph Structural Attention Neural Network (HetSANN) to directly encode structural information of HIN without meta-path and achieve more informative representations. With this method, domain experts will not be needed to design meta-path schemes and the heterogeneous information can be processed automatically by our proposed model. Specifically, we implicitly represent heterogeneous information using the following two methods: 1) we model the transformation between heterogeneous vertices through a projection in low-dimensional entity spaces; 2) afterwards, we apply the graph neural network to aggregate multi-relational information of projected neighborhood by means of attention mechanism. We also present three extensions of HetSANN, i.e., voices-sharing product attention for the pairwise relationships in HIN, cycle-consistency loss to retain the transformation between heterogeneous entity spaces, and multi-task learning with full use of information. The experiments conducted on three public datasets demonstrate that our proposed models achieve significant and consistent improvements compared to state-of-the-art solutions.

## 1 Introduction

Graph embedding pursues informative numerical representations of graphs, which facilitates various applications on graphs such as classification, link prediction and entity alignment. Most of existing methods perform graph embedding on homogeneous graphs, where all nodes and relationships (a.k.a. linkages or edges) are of the same type. For example, DeepWalk (Perozzi, Al-Rfou, and Skiena 2014) minimize the distance between the node and its neighboring nodes in the low-dimensional vector space, to preserve the structural information of the homogeneous graph. However, the real-world data tends to be presented as a heterogeneous graph, which combines different aspects of information together.




(a) (b)

Figure 1: A toy example of the heterogeneous graph and the meta-path. (a) A heterogeneous academic graph. (b) Previous meta-path-based method.

**Heterogeneous Information Network (HIN).** A HIN (a.k.a. heterogeneous graph) comprises more than two types of nodes or edges. Fig. 1a illustrates a toy example of HIN, including three types of nodes (author, paper, and conference) and six types of edges (cite/cited, write/written, and publish/published). Note that here we regard the relationship between vertices in HIN as the directed edge, and we set reverse relationships (e.g., written) for the directed relationships (e.g., write) in HIN. Compared with the homogeneous graph, HIN suffers from two major challenges:

- **C1:** how to model the entity space of multiple types of nodes? In a homogeneous graph, all nodes are embedded into the same low-dimensional entity space. In contrast, various types of nodes in HIN are naturally modeled in distinct spaces. However, a vertex may connect to multiple types of nodes, e.g., a paper is written by the author and it will be published by an academic conference. It is imperative to design the way of interaction between vertices in different type-specific entity space.
- **C2:** how to preserve the semantic of different relation-

*This work was done when the second author was an intern at AI Labs, Didi Chuxing, Beijing, China.

†Corresponding author.

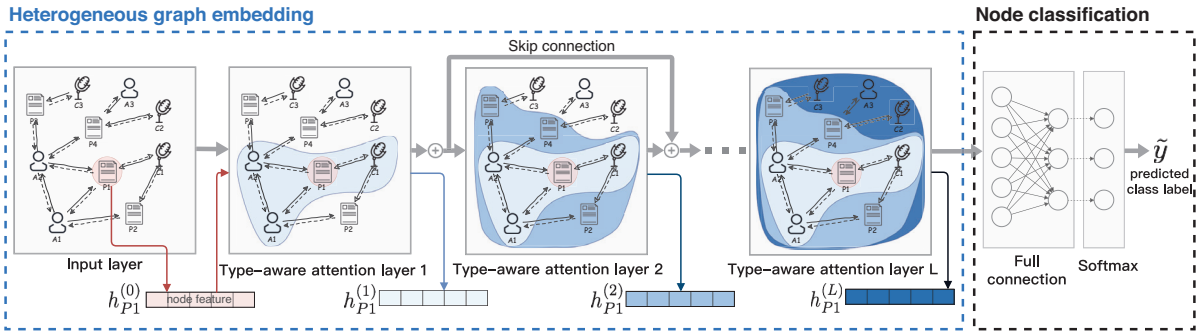



Figure 2: The overall framework of our proposed model HetSANN.

ships between nodes? For a HIN, there exist a variety of relations between both different node pairs and the same node pair. In the case of academic graph, an author can cite another author and meanwhile they can be the co-authors of some paper. The various relationships draw different semantic contents of the vertex. Thus the characterization of neighboring vertices with different relations to a vertex determines the performance of learned low-dimensional representation space.

Most of contemporary researches in HIN embedding focus on adapting HIN to homogeneous representation learning algorithms via the meta-path (Shi et al. 2016). As shown in Fig. 1b, the linkages between authors can be generated based on the designed meta-path scheme *APCPA*, and a representation learning algorithm for homogeneous graphs, e.g., DeepWalk (Perozzi, Al-Rfou, and Skiena 2014) adopted in metapath2vec (Dong, Chawla, and Swami 2017) or GNN (Gori, Monfardini, and Scarselli 2005) used in HAN (Wang et al. 2019), is implemented to the generated graph. See Section 4 for more details on meta-path-based methods.

Despite the success of meta-path-based heterogeneous graph embedding methods, these solutions employ handcrafted meta-path schemes to find homogeneous node neighbors, making them suffer from two predominant problems: 1) the scheme of meta-path relies on experts, and it is hard to exhaustively enumerate and select valuable meta-path schemes by hand; 2) the information passing by the meta-path, such as features of heterogeneous nodes or edges, is lost in the process of generating meta-path based node pairs, and it may even lead to an inferior embedding performance.

In this paper, we cast the meta-path aside and propose a novel method to learn the low-dimensional vector space preserving both structural and semantics information in HIN. Specifically, we take advantage of graph neural network (GNN) to conduct the structural information of HIN and we train the model by the task-guided objective function (node classification loss in this paper). To tackle the challenges of HIN mentioned above, we design a dedicated Type-aware Attention Layer instead of the convolutional layer in the conventional GNN. For each type-aware attention layer, a transformation operation that projects vertices from different entity space to the same low-dimensional target space is defined for the interaction between heterogeneous nodes **(C1)**, and the attention strategies focusing on different types of edges are applied for the aggregation of neighboring vertices with different semantics **(C2)**. Moreover, we develop two kinds of attention scoring functions of proposed type-aware attention layer including *concat product* and *voices-sharing product*[1]. To better model the interaction between heterogeneous nodes, we further introduce a restriction to the transformation operation. Finally, we perform multi-task learning in our proposed model which generally benefits the robustness of representations. To sum up, the main contributions of this paper are as follows:

- We propose **Het**erogeneous Graph **S**tructural **A**ttention **N**eural **N**etwork (HetSANN). Unlike previous meta-path-based solutions, HetSANN directly leverages and explores the structures in the heterogeneous graph to achieve more informative representations.
- We present three extensions of HetSANN: (E1) Enhance the extent of sharing information with multi-task learning. (E2) Take the pairwise relationship between the directed edge and the reversed edge into account (*voices-sharing product*). (E3) Introduce a constraint to the transformation operation to keep cycle consistent.
- We evaluate the proposed HetSANN with the node classification task on three heterogeneous graph datasets. The experimental results demonstrate the superiority of HetSANN compared to various state-of-the-arts. In addition, an ablation study about the three extensions of HetSANN is conducted and the result shows that all three extensions achieve improvement upon the vanilla of HetSANN.

## 2 Heterogeneous Graph Structural Attention Neural Network (HetSANN)

A heterogeneous graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ consists of a set of vertices $\mathcal{V}$ and a set of edges $\mathcal{E}$. There is a set of node types $\mathcal{A}$, and each vertex $v \in \mathcal{V}$ belongs to one of the node types, denoted by $\phi(v) = p \in \mathcal{A}$, where $\phi$ is the mapping function from $\mathcal{V}$ to $\mathcal{A}$. We represent an edge $e \in \mathcal{E}$ from the vertex $i \in \mathcal{V}$ to $j \in \mathcal{V}$ with a relation type $r$ as a triplet $e = (i, j, r)$, where $r \in \mathcal{R}$ and $\mathcal{R}$ is the set of relation types. For a directed

[1]The voice is the concept of English grammar including active voice and passive voice. Here we refer the active voice to the directed edge (cite, write, etc.) and refer the passive voice to the reversed edge (cited, written, etc.).

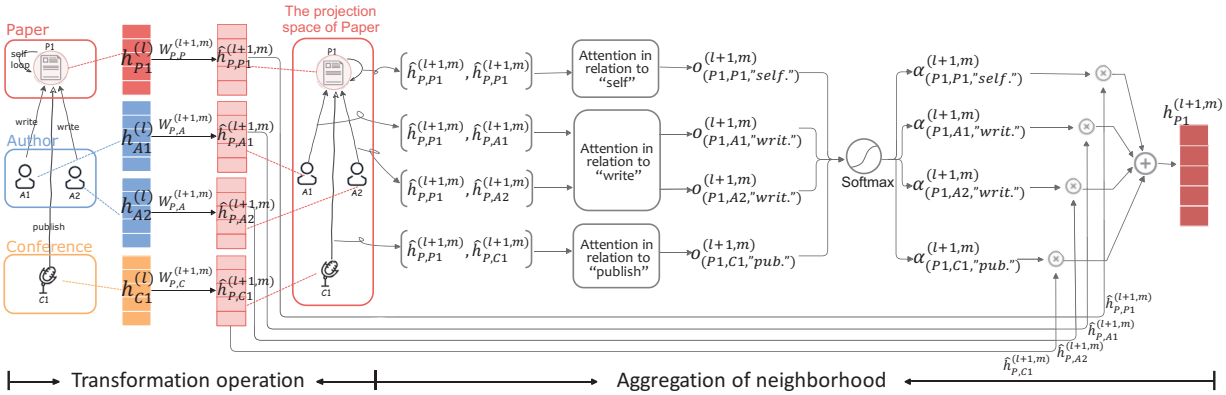



Figure 3: The dataflow of each head of the type-aware attention layer in HetSANN.

edge $e = (i, j, r)$ in canonical direction, we consider its reversed edge as $\tilde{e} = (j, i, \tilde{r})$ where $\tilde{r} \in \mathcal{R}$ is different from $r$. For a vertex $j$, the set of linkages with its neighboring nodes is defined as $\mathcal{E}_j = \{(i, j, r) \in \mathcal{E} \mid i \in \mathcal{V}, r \in \mathcal{R}\}$.

In this paper, we aim to learn the low-dimensional representation $\mathbf{h}_i \in \mathbb{R}^{n_{\phi(i)}}$, where $n_{\phi(i)}$ is the dimension of embedding space for node type $\phi(i)$, for each vertex $i$ in the heterogeneous graph and apply it to the downstream node classification task. Note that various relationship types can occur simultaneously when the vertex $i$ links to $j$, which would be a challenge of the heterogeneous graph embedding. To tackle these challenges, we propose a task-guided heterogeneous graph embedding method, namely HetSANN. As shown in Fig. 2, the key component of HetSANN framework is the type-aware attention layer presented as follows.

## 2.1 Type-aware Attention Layer (TAL)

The TAL is primarily motivated as an adaptation layer of GNNs, which performs convolution operation on local graph neighborhoods. Before conducting the embedding procedure, we connect each vertex $i$ to itself with the self-loop relation about $\phi(i)$. And we have the cold start state $\mathbf{h}_i^{(0)} \in \mathbb{R}^{n^{(0)}_{\phi(i)}}$ for per node $i \in \mathcal{V}$. The cold start state can be either the attribute features of nodes, or the dummy features (zero vector/one-hot vector) for the nodes without attributes.

Each TAL employs multi-head attention mechanism (Vaswani et al. 2017), which has been proved that it is helpful to stabilize the learning process of attention mechanism and enrich the model capacity (Velickovic et al. 2018). The dataflow of each head of the TAL is illustrated in Fig. 3. Consider a vertex $j \in \mathcal{V}$ presented as $\mathbf{h}_j^{(l)}$ in the $l$-th layer. An attention head $m$ in the $(l+1)$-th TAL outputs the corresponding hidden state $\mathbf{h}_j^{(l+1,m)}$ by the following two operations: the transformation operation and the aggregation of the neighborhood in the in-degree distribution of vertex $j$.

**Transformation Operation (C1)** We first apply a linear transformations $W^{(l+1,m)}_{\phi(j),\phi(i)} \in \mathbb{R}^{n^{(l+1,m)}_{\phi(j)} \times n^{(l)}_{\phi(i)}}$ to each neighboring vertex $i$ of vertex $j$:

$$\hat{\mathbf{h}}^{(l+1,m)}_{\phi(j),i} = W^{(l+1,m)}_{\phi(j),\phi(i)} \mathbf{h}^{(l)}_i \tag{1}$$

where $\hat{\mathbf{h}}^{(l+1,m)}_{\phi(j),i} \in \mathbb{R}^{n^{(l+1,m)}_{\phi(j)}}$ is the projection from previous hidden state in the space of type $\phi(i)$ to the hidden space of node type $\phi(j)$ in $m$-th head of layer $l+1$. That is, we transform the neighboring nodes of vertex $j$ to the same low-dimensional vector space of the node type $\phi(j)$, intended for the neighborhood aggregation.

**Aggregation of Neighborhood (C2)** To preserve the semantic of different types of relationship between nodes, we utilize $|\mathcal{R}|$ attention scoring functions to match different relation patterns, i.e., $\mathcal{F}^{(l+1,m)} = \{f_r^{(l+1,m)} \mid r \in \mathcal{R}\}$. For a vertex $j$, an attention coefficient is computed for each link edge $e = (i, j, r) \in \mathcal{E}_j$ in the form as:

$$o_e^{(l+1,m)} = \sigma\left(f_r^{(l+1,m)}\left(\hat{\mathbf{h}}^{(l+1,m)}_{\phi(j),j}, \hat{\mathbf{h}}^{(l+1,m)}_{\phi(j),i}\right)\right) \tag{2}$$

where $\sigma$ is an activation function implemented by LeakyReLU$(\cdot)$ (Maas, Hannun, and Ng 2013). The attention coefficient $o_e$ indicates the importance of edge $e$ to the target vertex $j$. In principle, the attention scoring functions can be different forms to capture various link types. For simplicity, we adopt the same form of attention mechanism for all linkage types but different in the parameters. A natural form of the attention scoring function is the *concat product*, which is adopted in GAT (Velickovic et al. 2018), defined as:

$$f_r^{(l+1,m)}(e) = \left[\hat{\mathbf{h}}^{(l+1,m)\mathsf{T}}_{\phi(j),j} \| \hat{\mathbf{h}}^{(l+1,m)\mathsf{T}}_{\phi(j),i}\right] \mathbf{a}_r^{(l+1,m)} \tag{3}$$

where $\|$ denotes the concatenation operation, and $\mathbf{a}_r^{(l+1,m)} \in \mathbb{R}^{2n^{(l+1,m)}_{\phi(j)}}$ is the trainable attention parameter shared by the same edge type $r$. Different with the HAN (Wang et al. 2019) which employs the hierarchical attention mechanism based on the meta-path schemes, we utilize the attention mechanism directly to the raw heterogeneous links. Thereby, the softmax is applied over the neighborhood linkages of vertex $j$ for the normalization of the attention coefficient:

$$\alpha_e^{(l+1,m)} = \exp\left(o_e^{(l+1,m)}\right) / \sum_{k \in \mathcal{E}_j} \exp\left(o_k^{(l+1,m)}\right). \tag{4}$$

Now we have the hidden states of neighboring nodes in the same low-dimensional space of the target node $j$, and

weights of each linkage associated with vertex $j$. Then the neighborhood aggregation for vertex $j$ can be performed as:

$$\mathbf{h}_j^{(l+1,m)} = \sigma\Big( \sum_{e=(i,j,r)\in\mathcal{E}_j} \alpha_e^{(l+1,m)} \hat{\mathbf{h}}_{\phi(j),i}^{(l+1,m)} \Big). \quad (5)$$

In our proposed model, the node pair of edge and the relation type of edge are used together to identify edges. When vertex $i$ links to $j$ with multiple types of relationship, the hidden state $\hat{\mathbf{h}}_{\phi(j),i}^{(l+1,m)}$ is propagated to vertex $j$ multiple times with the corresponding weight $\alpha_{(i,j,r)}^{(l+1,m)}$.

With $M$ attention heads executing the procedure of Eq. (5), we concatenate the low-dimensional vectors of attention heads and output the representation of each node in the type-aware attention layer $l+1$:

$$\mathbf{h}_j^{(l+1)} = \Big\Vert_{m=1}^{M} \mathbf{h}_j^{(l+1,m)}, \quad (6)$$

where $\mathbf{h}_j^{(l+1)} \in \mathbb{R}^{\sum_{m=1}^{M} n_{\phi(j)}^{(l+1,m)}}$. The aggregation of HetSANN is conducted on the raw links instead of the generated links based on meta-paths. That is, a vertex $i$ can be propagated to vertex $j$ within one layer of GNN for the meta-path-based links, while more layers are needed for the raw links. Thus, a deeper model is used in HetSANN to capture the high-order proximity information. To facilitate training, we adopt the residue mechanism, which is first introduced by (He et al. 2016), and we revise Eq. (6) as following:

$$\mathbf{h}_j^{(l+1)} = \mathbf{h}_j^{(l)} + \Big\Vert_{m=1}^{M} \mathbf{h}_j^{(l+1,m)}. \quad (7)$$

## 2.2 Model Training and Three Extensions

The last type-aware attention layer outputs the low-dimensional representations for each vertex in the heterogeneous graph, i.e. $\mathbf{h}_j = \mathbf{h}_j^{(L)}$. To optimize the representations toward the target task, such as node classification in this paper, we integrate the representations of nodes into a node classifier (implemented with a full connection layer with softmax function) to infer the label of classification. With the guide of labeled data, we minimize the cross-entropy loss:

$$\mathcal{J}_{\text{class}} = -\sum_{i\in\mathcal{V}_p} y_i \log \tilde{y}_i, \quad (8)$$

where $\mathcal{V}_p$ is the set of labeled vertices belonging to the node type $p$. $y_i$ and $\tilde{y}_i$ are the ground truth and the predicted class label for vertex $i$, respectively.

**E1: *Multi-task Learning*** We can further employ several node classifiers for different types of nodes. The parameters of all type-aware attention layers are shared and trained by multiple classifiers. The multi-task learning via uniting all classifiers greatly reduces the risk of overfitting and benefits the robustness of representations (Baxter 1997).

**E2: *Voices-sharing Product*** The *concat product* scoring function considers the directed edge (e.g., write) and the reversed edge (e.g., written) as independent types of relationships. Intuitively, vertex $i$ will link to vertex $j$ with the "written" relation when vertex $j$ link to vertex $i$ with the "write" relation. To formulate the pairwise relationship between the directed edge and the reversed edge, we share the parameters of attention mechanism between the pairwise edge types $r$ and $\tilde{r}$, where $\tilde{r}$ is the type of reversed edge of edges with type $r$. Technically, we enforce $\mathbf{a}_{\tilde{r}}^{(l,m)} = -\mathbf{a}_r^{(l,m)} \in \mathbb{R}^{n_{\phi(j)}^{(l,m)}}$ and adapt the attention scoring function $f_r^{(l,m)}$ as follow (called *voices-sharing product*):

$$f_r^{(l,m)}(e) = \hat{\mathbf{h}}_{\phi(j),j}^{(l,m)\,\mathsf{T}} \left( \hat{\mathbf{h}}_{\phi(j),i}^{(l,m)} + \mathbf{a}_r^{(l,m)} \right). \quad (9)$$

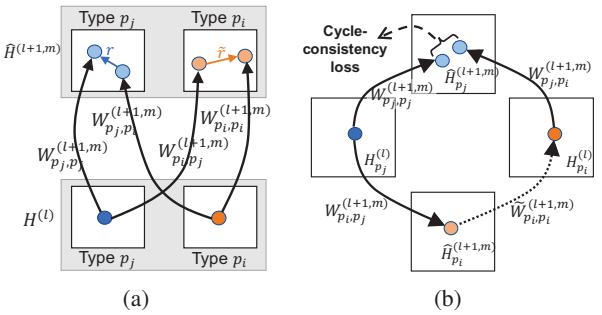



Figure 4: An illustration for the transformation operation. (a) There are two forms of transformation between node type $p_i$ and $p_j$, i.e. $p_i \to p_j$ and $p_j \to p_i$. (b) Cycle-consistency loss.

**E3: *Cycle-consistency Loss*** In natural language processing, "back translation and reconciliation" has been a popular trick to verify and improve the performance on translation (Brislin 1970). Referring back to the transformation operation between heterogeneous nodes in Eq. (1), we have a transformation from node type $p_i \in \mathcal{A}$ to $p_j \in \mathcal{A}$ and another transformation from $p_j$ to $p_i$. Particularly, a self-transformation is applied to per type of node, i.e. $p_i \to p_i$. The transformation operation between $p_i$ and $p_j$ is illustrated in Fig. 4a, which is intuitive that a vertex should return to the starting position after a cycle. Therefore, we introduce a cycle-consistency restriction to the transformation operation:

$$W_{p_j,p_i}^{(l+1,m)} \left( W_{p_i,p_i}^{(l+1,m)} \right)^{-1} W_{p_i,p_j}^{(l+1,m)} h_j^{(l)} \approx W_{p_j,p_j}^{(l+1,m)} h_j^{(l)} \quad (10)$$

where ${W_{p_i,p_i}^{(l+1,m)}}^{-1}$ is the inverse of $W_{p_i,p_i}^{(l+1,m)}$. However, the solution of matrix inversion is a notoriously time-consuming problem. To reduce the computational complexity, we adopt a trainable matrix $\widetilde{W}_{p_i,p_i}^{(l+1,m)}$ instead of the inverse of the matrix $W_{p_i,p_i}^{(l+1,m)}$ and restrain it as follows:

$$\widetilde{W}_{p_i,p_i}^{(l+1,m)} W_{p_i,p_i}^{(l+1,m)} \approx W_{p_i,p_i}^{(l+1,m)} \widetilde{W}_{p_i,p_i}^{(l+1,m)} \approx I, \quad (11)$$

where $I$ is the identity matrix. These constraints are integrated as *cycle-consistency loss* (as shown in Fig. 4b):

$$\begin{aligned} &\mathcal{J}_{\text{cycle}} = \\ &\beta_1 \mathop{\mathbb{E}}_{p_i,p_j\in\mathcal{A}} \left[ \left( W_{p_j,p_i}^{(l,m)} \widetilde{W}_{p_i,p_i}^{(l,m)} W_{p_i,p_j}^{(l,m)} h_j^{(l-1)} - W_{p_j,p_j}^{(l,m)} h_j^{(l-1)} \right)^2 \right] \\ &+ \beta_2 \mathop{\mathbb{E}}_{p\in\mathcal{A}} \left[ \left( \widetilde{W}_{p,p}^{(l,m)} W_{p,p}^{(l,m)} - I \right)^2 + \left( W_{p,p}^{(l,m)} \widetilde{W}_{p,p}^{(l,m)} - I \right)^2 \right] \end{aligned} \quad (12)$$

where $\beta_1$ and $\beta_2$ are the weighting factors. The objective function of our model therefore is derived as:

$$\min \mathcal{J} = \mathcal{J}_{\text{class}} + \mathcal{J}_{\text{cycle}}. \quad (13)$$

## 3 Experiments

**Comparative Models** The list of models in comparison includes:

**1) Variants of our proposed model[2]:** We denote HetSANN as the proposed vanilla version, i.e. without aforementioned three extensions in Section 2.2. Three suffixes "$.M$", "$.R$" and "$.V$" indicate *multi-task learning* to optimize the parameters, *voices-sharing product* in relations attention mechanism and *cycle-consistency loss* to retain the transformation between vertices, respectively. And HetSANN$.M.R.V$ refers to the full version of our proposed model.

All variations employ 3-layer HetSANN and each TAL consists of 8 attention heads. The output dimensions of each attention head are consistent to 8. The parameters are optimized via Adam solver (Kingma and Ba 2015) with a learning rate 0.001 for IMDB and 0.005 for other datasets. A regularization weight 0.0005 is applied to all trainable parameters. A dropout rate 0.6 (Srivastava et al. 2014) is implanted between hidden layers to stabilize our model training procedure. For the variant of HetSANN with suffix "$.V$", the weight coefficients $\beta_1 = 10^{-3}$ and $\beta_2 = 10^{-5}$.

**2) Baseline models:** We compare with the state-of-art baselines of which codes is publicly available at the website, including DeepWalk (Perozzi, Al-Rfou, and Skiena 2014), metapath2vec (Dong, Chawla, and Swami 2017), HERec (Shi et al. 2018), HAN (Wang et al. 2019), GCN (Kipf and Welling 2017), R-GCN (Schlichtkrull et al. 2018) and GAT (Velickovic et al. 2018). All of baseline models are introduced in Section 4, and the implementation of them is detailed as follows.

We turn the homogeneous graph embedding methods (DeepWalk, GCN and GAT) into the model for the heterogeneous graph embedding learning by ignoring the type of nodes and linkages. R-GCN is implemented by regarding all nodes as the same type node but distinguishing different types of relations in the graph. We follow most of parameters setting recommended in the published papers, and tune a few parameters to adapt to the dataset in our experiments. Specifically, we set the walk length to 50 and 100 walks/node for metapath2vec, HERec and DeepWalk, and the learned embeddings by them are used to train a 2-layer MLP (Rumelhart and Mcclelland 1988) classifier. For the graph neural network based methods, the number of layers is set to 3 for GCN, R-GCN and GAT. We use 8 attention heads in each layer of the attention-based models, i.e., HAN and GAT. To enable comparison, we design some meta-path schemes for each dataset and evaluate all meta-paths schemes for metapath2vec and HAN, and report the best performance.

All networks were trained from scratch until convergence. The dimension of the embedding is unanimously set to 64 for all models. And the model settings for all dataset are the same except for special instructions.

Table 1: The statistics of datasets in our experiments.

| Dataset | Nodes | Linkages | isLabel |
|---|---|---|---|
| IMDB | movie(5043)<br>actor(2357)<br>director(894) | movie-actor(11188)<br>movie-director(3435) | movie |
| DBLP | author(14475)<br>paper(14376)<br>venue(20) | author-paper(41794)<br>paper-venue(14376) | author<br>paper |
| AMiner | author(8052)<br>paper(20201) | author-author(31224)<br>paper-paper(44551)<br>author-paper(32029) | author<br>paper |

**Datasets** We collected a movie graph from IMDB site[3] and constructed two academic networks from DBLP (Ji et al. 2010) and AMiner (Tang et al. 2008) datasets respectively. Each of these three data sets, the statistics of which are tabulated in Table 1, is a heterogeneous graph, consisting of more than two types of nodes or edges.

- **IMDB** The IMDB records the Actors and Directors of the Movies. The movies are divided into three groups according the genre label: Action, Comedy or Drama. We utilize the keywords about plot of the movie as the attribute feature of movie vertex by the way of bag-of-words. For meta-path-based models, we set two meta-path schemes, i.e., *MAM* and *MDM*.
- **DBLP** In an academic network, Authors published their Papers in the Venues. The DBLP dataset we constructed consists of 20 venues from four different research fields: database, data mining, machine learning, and information retrieval. Each paper is labeled according to the research field of the venue where the paper is published, and each paper is characterized by the bag-of-words of keywords. Each author is labeled based on the research fields of her/his publications, and we sum up the bag-of-words of the papers published by this author as the feature of the author. Again, we set *PAP* and *PVP*, *APA* and *APVPA* as the meta-path schemes for meta-path-based models in paper classification task and author classification task respectively.
- **AMiner** We cast aside venue nodes from AMiner academic network, raising a harder classification task. In addition to the publishing relationship between the paper and the author, we also introduce the citation relationship between papers and the collaboration relationship between authors. Similar to DBLP, each paper in AMiner is characterized by the bag-of-words of keywords, and papers and authors are labeled into four research fields: database, data mining, natural language processing and computer vision. The attribute features of each author is

[2] Available at https://github.com/didi/hetsann

[3] https://www.imdb.com

Table 2: Comparison results for node classification in datasets.

| Dataset | IMDB | | DBLP | | | | AMiner | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| Target node | Movie | | Author | | Paper | | Author | | Paper | |
| Metrics (%) | Mic F1 | Mac F1 | Mic F1 | Mac F1 | Mic F1 | Mac F1 | Mic F1 | Mac F1 | Mic F1 | Mac F1 |
| DeepWalk | 63.53 | 54.91 | 92.71 | 92.01 | 99.41 | 99.30 | 84.70 | 84.99 | 87.71 | 87.70 |
| metapath2vec | 60.83 | 50.27 | 66.75 | 67.07 | 70.35 | 72.89 | 61.79 | 61.72 | 71.93 | 71.58 |
| HERec | 62.54 | 53.62 | 90.49 | 89.94 | 99.93 | 99.93 | 68.74 | 68.77 | 78.75 | 78.61 |
| HAN | 61.91 | 57.87 | 88.35 | 87.67 | **100.00** | **100.00** | 33.24 | 31.92 | 91.47 | 91.73 |
| GCN | 63.78 | 51.13 | 87.09 | 86.60 | 91.31 | 89.87 | 81.24 | 81.69 | 90.55 | 90.76 |
| R-GCN | 67.13 | 62.58 | 87.70 | 86.96 | 81.92 | 79.37 | 85.11 | 85.31 | 90.84 | 91.05 |
| GAT | 65.19 | 60.43 | 89.11 | 88.57 | 89.14 | 87.51 | 86.86 | 87.44 | 90.79 | 90.96 |
| HetSANN | 73.11 | 71.20 | 94.89 | 94.56 | 99.72 | 99.67 | 86.97 | 87.48 | 91.20 | 91.37 |
| HetSANN.$M$ | | | 95.43 | 95.21 | 99.08 | 98.89 | 91.55 | 91.99 | 92.56 | 92.77 |
| HetSANN.$M.R$ | 73.20 | 71.38 | 95.51 | 95.28 | 99.11 | 98.95 | 92.43 | 92.87 | **93.75** | **93.93** |
| HetSANN.$M.R.V$ | **73.86** | **72.00** | **95.63** | **95.38** | 98.69 | 98.43 | **92.47** | **92.91** | 93.73 | 93.92 |

provided with five indices[4] indicating the academic authority of the author. Again, *PAP* and *PAAP*, and *APA* and *APPA* are set as meta-path schemes.

**Evaluation Metrics** The whole labeled dataset is randomly split into training set, validation set and test set by a ratio of 0.8:0.1:0.1. And we select the best one in the validation set for each comparative model, then evaluate them by Micro F1 and Macro F1 on the test set. For each model, we report the average performance on 10 repeated processes.

### 3.1 Ablation Study

In this section, we employ the vanilla HetSANN and its variations including HetSANN.$M$, HetSANN.$M.R$ and our full method HetSANN.$M.R.V$ to perform an ablation study. To enable the multi-task learning, we introduce the Paper classification as the auxiliary task to the main Author classification task. And the *multi-task learning* (with suffix "$.M$") can not be conducted on the IMDB dataset which contains only single type of labeled node. The test results are shown in the bottom half of Table 2: (1) HetSANN.$M$ performs better than HetSANN both in Author and Paper classification tasks in AMiner. For DBLP dataset, HetSANN.$M$ still brings improvement in the task of Author classification, although it is slightly inferior in the performance of the Paper classification compared to HetSANN that is trained toward only one single task. We believe that the introduced *multi-task learning* guides our model to find an optimum representation that captures all of tasks, even if sometime it involves losing accuracy of one task in return for gaining performance in overall (Baxter 1997); (2) Benefiting from the substitution of *concat product* with *voices-sharing product*, HetSANN.$M.R$ improves the classification performance over all datasets compared with HetSANN.$M$; (3) HetSANN.$M.R.V$ achieves the best performance of variants of our models in the Author and Movie classification tasks. However, the gain of HetSANN.$M.R.V$ is relatively constrained as seen from the comparison between HetSANN.$M.R$ and HetSANN.$M.R.V$. One clear reason is the replacement of the analytical expression of the inverse matrix by the trainable matrix in the cycle-consistency loss [Eq. (12)], which is left to future work.

[4]Five indices (Stallings et al. 2013) for each author: the count of published papers; the total number of citations; the H-index; the P-index with equal A-index; the P-index with unequal A-index.

### 3.2 Comparison Results

Table 2 also shows the comparison results of our models with other baselines. Obviously, our models are superior to other models in the classification tasks of all datasets excluding Paper classification on DBLP. Note that we label papers according to the research field of the venue where the paper is published. The venue nodes connected to papers in DBLP enable HAN to establish the neighborhood of papers published in the same venue via the meta-path scheme *PVP*, resulting in a perfect Paper classification performance on DBLP. Without the venue vertices in AMiner, it is not easy for HAN to capture the category information of Paper via *PAP* and *PAAP* schemes, leading to worst Author classification results in AMiner. Different from the methods which require well-designed solutions of the meta-path, the meta-path-free methods achieve obvious performance gains and robustness results. With a further distinction between multiple types of nodes and linkages, our models outperform other baselines and our full method HetSANN.$M.R.V$ improves Micro F1 and Macro F1 by $3\%\sim13\%$ and $3\%\sim19\%$ respectively over the most competitive model GAT on three datasets.

Fig. 5 details the comparison performance results of main tasks (Movie and Author classification) and the auxiliary tasks (Paper classification) on the three datasets, varying the value of training ratio in $\{0.2,0.4,0.6,0.8\}$. Consistently, we have HetSANN.$M.R.V$>HetSANN>baselines in terms of Micro F1 score of classification. Besides, both HetSANN and HetSANN.$M.R.V$ still maintain the advantage in a weakly-supervised manner.

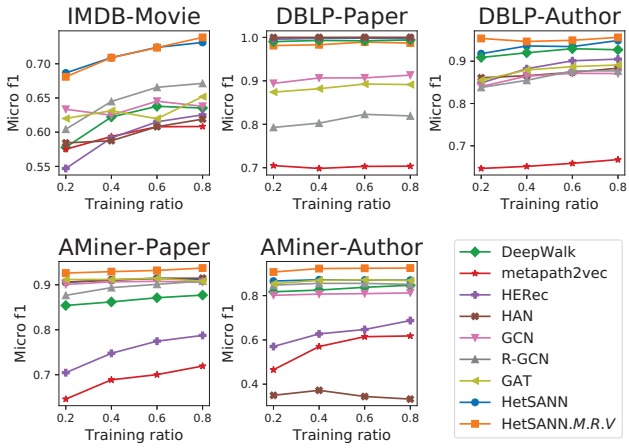



Figure 5: Micro F1 vs. Training ratio.

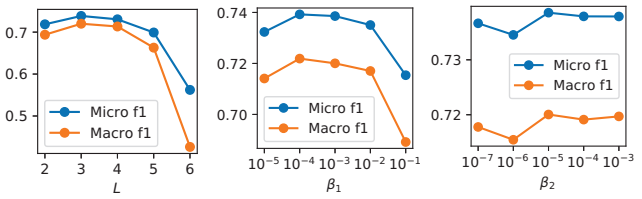
### 3.3 Parameter Sensitivity Study

Finally, we test the parameter sensitivity of HetSANN.$R.V$ on IMDB and the results are presented in Fig. 6. The left figure shows the effect of the number of type-aware attention layers $L$ when other parameters are fixed. The performance of HetSANN.$R.V$ goes down when $L > 5$. The observation is consistent with the analysis in (Li, Han, and Wu 2018), that is, the graph convolution is a form of Laplacian smoothing over the features of neighborhood, and it will lead to indistinguishable features of the node and inferior performance of node classification when many convolutional layers are included in a GNN. The remaining two figures focus on the weighting factors $\beta_1$ and $\beta_2$ of the cycle-consistency loss. Fixing $\beta_2 = 10^{-5}$, the lines in Fig. 6b increase when $\beta_1$ increases to $10^{-4}$ and lines decrease when we set a larger $\beta_1$, which may suppress the learning of main task, i.e., node classification. Fig. 6c in turn varying $\beta_2$ and fixing $\beta_1 = 10^{-3}$, lines tend to be stable when $\beta_2 > 10^{-5}$, indicating that the model have done its best to reach the solution of the inverse matrix in Eq. (11).

## 4 Related Work

### 4.1 Heterogeneous Graph Representation Learning

The existing works of HIN embedding tend to utilize the meta-path to adapt the heterogeneous graph for the application of the homogeneous graph embedding methods, such as (Perozzi, Al-Rfou, and Skiena 2014; Tang et al. 2015; Wang, Cui, and Zhu 2016). metapath2vec (Dong, Chawla, and Swami 2017) designs meta-paths to guide the random walks in a heterogeneous graph and then follows skip-gram model to learn the latent-space representations of vertices. Inspired by metapath2vec, HERec(Shi et al. 2018) proposed to fuse different representations learned in the view of different meta-path schemes. Both metapath2vec and HERec are trained by linkage-guided objective function which is independent of the downstream tasks. To obtain optimum embeddings for a specific task, (Chen and Sun 2017) joints author identification task-guided and linkage-guided objectives to learn the heterogeneous graph embeddings. HAN (Wang et al. 2019) introduces a two-levels hierarchical attention to GNN, in which a node-level attention captures the relations between neighboring nodes generated by one meta-path scheme and a semantic-level attention aggregates multiple meta-path scheme for each node in a graph. All in all, these aforementioned methods are dependent on the design of experts and brings inevitable loss of information.




(a) Sensitivity to $L$ (b) Sensitivity to $\beta_1$ (c) Sensitivity to $\beta_2$

Figure 6: Parameter analysis of HetSANN.$R.V$ on IMDB.

### 4.2 Graph Neural Networks (GNNs)

More recently, graph neural networks (GNNs) (Gori, Monfardini, and Scarselli 2005; Hamilton, Ying, and Leskovec 2017; Kipf and Welling 2017; Battaglia et al. 2018) have become increasingly studied. GNNs generate node embeddings by the spatial filter which convolutes each node over its neighborhood in graph. The convolutional operation enables GNNs to propagate structural information of graphs layer by layer, and frees graph embedding methods from linkage-guided learning. Motivated by the thriving of attention mechanism, GAT (Velickovic et al. 2018) introduces an attention strategy to GNN framework. For the relational learning of knowledge bases, R-GCN (Schlichtkrull et al. 2018) builds multiple relation spaces for all nodes in a graph, which can not capture the relative importance of various type of nodes. RSHN (Zhu et al. 2019) focuses on mining semantic interactions of edge types via coarsened line graph and incorporating it into the H-GNN model.

## 5 Conclusion

The paper proposes HetSANN to perform meta-path-free embedding based on structural information in heterogeneous graphs. We design a type-aware attention layer for HetSANN, which embeds each vertex of heterogeneous graph by jointing different types of neighboring nodes and associated linkages. A few variants of our model are developed based on three extensions, i.e., *voices-sharing product*, *cycle-consistency loss* and *multi-task learning*. Comprehensive experiments on three popular datasets show that the proposed solutions outperform state-of-the-art methods in HIN embedding and node classification. Under the framework of HetSANN, the representation learning of HIN does not need to rely on the meta-path to tackle the heterogeneous structural information, thereafter the heterogeneous attributes of vertices will be considered in the future work.

## References


Battaglia, P. W.; Hamrick, J. B.; Bapst, V.; Sanchez-Gonzalez, A.; Zambaldi, V.; Malinowski, M.; Tacchetti, A.; Raposo, D.; Santoro, A.; Faulkner, R.; et al. 2018. Relational inductive biases, deep learning, and graph networks. *arXiv preprint arXiv:1806.01261*.

Baxter, J. 1997. A bayesian/information theoretic model of learning to learn via multiple task sampling. *Machine Learning* 28(1):7–39.

Brislin, R. W. 1970. Back-translation for cross-cultural research. *Journal of cross-cultural psychology* 1(3):185–216.

Chen, T., and Sun, Y. 2017. Task-guided and path-augmented heterogeneous network embedding for author identification. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*, 295–304. ACM.

Dong, Y.; Chawla, N. V.; and Swami, A. 2017. metapath2vec: Scalable representation learning for heterogeneous networks. In *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining*, 135–144. ACM.

Gori, M.; Monfardini, G.; and Scarselli, F. 2005. A new model for learning in graph domains. In *Proceedings. 2005 IEEE International Joint Conference on Neural Networks, 2005.*, volume 2, 729–734. IEEE.

Hamilton, W.; Ying, Z.; and Leskovec, J. 2017. Inductive representation learning on large graphs. In *Advances in Neural Information Processing Systems*, 1024–1034.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, 770–778.

Ji, M.; Sun, Y.; Danilevsky, M.; Han, J.; and Gao, J. 2010. Graph regularized transductive classification on heterogeneous information networks. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, 570–586. Springer.

Kingma, D. P., and Ba, J. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Kipf, T. N., and Welling, M. 2017. Semi-supervised classification with graph convolutional networks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*.

Li, Q.; Han, Z.; and Wu, X.-M. 2018. Deeper insights into graph convolutional networks for semi-supervised learning. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Maas, A. L.; Hannun, A. Y.; and Ng, A. Y. 2013. Rectifier nonlinearities improve neural network acoustic models. In *Proc. icml*, volume 30, 3.

Perozzi, B.; Al-Rfou, R.; and Skiena, S. 2014. Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, 701–710. ACM.

Rumelhart, D., and Mcclelland, J. 1988. Learning internal representations by error propagation. *Readings in Cognitive Science* 323(6088):399–421.

Schlichtkrull, M.; Kipf, T. N.; Bloem, P.; Van Den Berg, R.; Titov, I.; and Welling, M. 2018. Modeling relational data with graph convolutional networks. In *European Semantic Web Conference*, 593–607. Springer.

Shi, C.; Li, Y.; Zhang, J.; Sun, Y.; and Philip, S. Y. 2016. A survey of heterogeneous information network analysis. *IEEE Transactions on Knowledge and Data Engineering* 29(1):17–37.

Shi, C.; Hu, B.; Zhao, W. X.; and Philip, S. Y. 2018. Heterogeneous information network embedding for recommendation. *IEEE Transactions on Knowledge and Data Engineering* 31(2):357–370.

Srivastava, N.; Hinton, G.; Krizhevsky, A.; Sutskever, I.; and Salakhutdinov, R. 2014. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research* 15(1):1929–1958.

Stallings, J.; Vance, E.; Yang, J.; Vannier, M. W.; Liang, J.; Pang, L.; Dai, L.; Ye, I.; and Wang, G. 2013. Determining scientific impact using a collaboration index. *Proceedings of the National Academy of Sciences* 110(24):9680–9685.

Tang, J.; Zhang, J.; Yao, L.; Li, J.; Zhang, L.; and Su, Z. 2008. Arnetminer: extraction and mining of academic social networks. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, 990–998. ACM.

Tang, J.; Qu, M.; Wang, M.; Zhang, M.; Yan, J.; and Mei, Q. 2015. Line: Large-scale information network embedding. In *Proceedings of the 24th international conference on world wide web*, 1067–1077. International World Wide Web Conferences Steering Committee.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, L. u.; and Polosukhin, I. 2017. Attention is all you need. In Guyon, I.; Luxburg, U. V.; Bengio, S.; Wallach, H.; Fergus, R.; Vishwanathan, S.; and Garnett, R., eds., *Advances in Neural Information Processing Systems 30*. Curran Associates, Inc. 5998–6008.

Velickovic, P.; Cucurull, G.; Casanova, A.; Romero, A.; Liò, P.; and Bengio, Y. 2018. Graph attention networks. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*.

Wang, X.; Ji, H.; Shi, C.; Wang, B.; Ye, Y.; Cui, P.; and Yu, P. S. 2019. Heterogeneous graph attention network. In *The World Wide Web Conference*, 2022–2032. ACM.

Wang, D.; Cui, P.; and Zhu, W. 2016. Structural deep network embedding. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*, 1225–1234. ACM.

Zhu, S.; Zhou, C.; Pan, S.; Zhu, X.; and Wang, B. 2019. Relation structure-aware heterogeneous graph neural network. In *Proceedings of the 19th IEEE International Conference on Data Mining*, 1534–1539.