Heterogeneous Graph Neural Network via Attribute Completion

Di Jin
College of Intelligence and Computing, Tianjin University
Tianjin, China
jindi@tju.edu.cn

Cuiying Huo
College of Intelligence and Computing, Tianjin University
Tianjin, China
huocuiying@tju.edu.cn

Chundong Liang
College of Intelligence and Computing, Tianjin University
Tianjin, China
liangchundong@tju.edu.cn

Liang Yang*
School of Artificial Intelligence, Hebei University of Technology, Tianjin, China
SKLOIS, IIE, CAS, Beijing, China
yangliang@vip.qq.com

ABSTRACT

Heterogeneous information networks (HINs), also called heterogeneous graphs, are composed of multiple types of nodes and edges, and contain comprehensive information and rich semantics. Graph neural networks (GNNs), as powerful tools for graph data, have shown superior performance on network analysis. Recently, many excellent models have been proposed to process hetero-graph data using GNNs and have achieved great success. These GNN-based heterogeneous models can be interpreted as smooth node attributes guided by graph structure, which requires all nodes to have attributes. However, this is not easy to satisfy, as some types of nodes often have no attributes in heterogeneous graphs. Previous studies take some handcrafted methods to solve this problem, which separate the attribute completion from the graph learning process and, in turn, result in poor performance. In this paper, we hold that missing attributes can be acquired by a learnable manner, and propose a general framework for Heterogeneous Graph Neural Network via Attribute Completion (HGNN-AC), including pre-learning of topological embedding and attribute completion with attention mechanism. HGNN-AC first uses existing HIN-Embedding methods to obtain node topological embedding. Then it uses the topological relationship between nodes as guidance to complete attributes for no-attribute nodes by weighted aggregation of the attributes from these attributed nodes. Our complement mechanism can be easily combined with an arbitrary GNN-based heterogeneous model making the whole system end-to-end. We conduct extensive experiments on three real-world heterogeneous graphs. The results demonstrate the superiority of the proposed framework over state-of-the-art baselines.

CCS CONCEPTS

• **Computing methodologies** → **Neural networks**; • **Information systems** → **Social networks**.

*Corresponding author.

WWW '21, April 19–23, 2021, Ljubljana, Slovenia

ACM ISBN 978-1-4503-8312-7/21/04.
https://doi.org/10.1145/3442381.3449914

KEYWORDS

Heterogeneous information networks, Graph neural networks, Missing data, Attribute completion

ACM Reference Format:
Di Jin, Cuiying Huo, Chundong Liang, and Liang Yang. 2021. Heterogeneous Graph Neural Network via Attribute Completion. In *Proceedings of the Web Conference 2021 (WWW '21), April 19–23, 2021, Ljubljana, Slovenia.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3442381.3449914

1 INTRODUCTION

In the real world, the complex systems are always represented in graph data structures, such as social networks, citation networks and so on. There is a surge of interest in learning on graph data, especially the heterogeneous graphs [31]. Heterogeneous graphs consist of multi-typed nodes and edges, corresponding to depicting various entities and their interactions in the real-world system. Taking DBLP as an example, we can model it as a heterogeneous graph which contains four node types (author, paper, term, venue) and three edge types (paper-author, paper-term, paper-venue), as shown in Figure 1 (b). Heterogeneous graphs can better model the real-world systems than traditional homogeneous graphs since they contain more comprehensive information and rich semantics.

Graph neural networks (GNNs) [42, 48], first proposed on homogeneous graphs, have shown state-of-the-art performance and caught a great attention of researchers. GNNs have been widely adopted in various tasks over graphs, such as graph classification [9, 20, 43], link prediction [18, 29, 47] and node classification [19, 38]. Recently, many well-performed models have been proposed to process hetero-graph [15, 32] data using GNNs, such as HAN [39] and MAGNN [11]. These GNN-based heterogeneous models can be interpreted as smooth node attributes in neighbors guided by graph structure. In order to learn node representation, all nodes' attributes are required. However, this is not always satisfied. Some nodes have no attributes because the cost is prohibitively expensive or even impossible (like sensitive personal information). Especially in heterogeneous graphs, we usually cannot get the attributes of all types of nodes, which will affect the performance of GNN-based models. We divide the attribute missing in the heterogeneous graph into two categories. One is that the nodes that need to be analyzed have no attributes, as shown the author nodes in DBLP in Figure 1 (b). The other is that the nodes that do not need to be analyzed have no attributes, as shown the actor nodes in IMDB

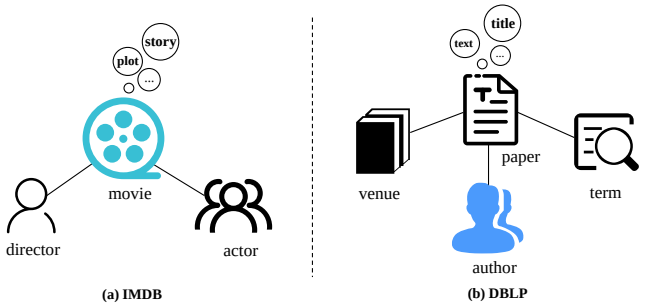


Figure 1: The network schemas of IMDB and DBLP with incomplete attributes. The colored items represent node types to be analysed (classification, clustering). Only the movie nodes in IMDB and paper nodes in DBLP have raw attributes, while other types of nodes have no attributes.

in Figure 1 (a). IMDB contains three types of nodes: movie, actor and director. However, only movie nodes have attributes which are bag-of-words representation of their plots. Similarly, in DBLP, only paper nodes have attributes directly derived from their keywords. Specifically, in DBLP, we usually perform analysis tasks on author nodes, but the attributes of author nodes are hard to get. In IMDB, we usually perform analysis tasks on movie nodes. Although movie nodes have attributes, research shows that the attribute information of actors and directors will also have a great impact on node analysis tasks [11]. The missing attributes will significantly affect the performance.

Although some types of nodes have no attributes, in most cases these nodes without attributes will be directly connected to nodes with attributes, so previous studies have adopted some handcrafted ways to deal with this problem of missing attributes in heterogeneous graphs. Taking MAGNN and HAN as an example, in the DBLP dataset, they use bag-of-words representation of paper keywords as the attributes of papers, which seems to be reasonable. But for authors, because no more relevant information is provided in the dataset, they use bag-of-words representation of keywords extracted from their published papers. This is the same as that an author's attribute vector comes from the mean of its directly connected papers' attribute vectors (which is actually done in IMDB). What's more, they use no computer-science-specialized pre-trained word vectors [25] and one-hot representation as the attributes of terms and venues, which may provide less effective information.

In this paper, we propose a general framework for Heterogeneous Graph Neural Network via Attribute Completion (HGNN-AC). The goal of the proposed framework is to solve the problem of missing some types of node attributes in heterogeneous graphs via learning. We use topological relationship between nodes as guidance to complete attributes for no-attribute nodes by weighted aggregation of the attributes of these attributed nodes. Specifically, HGNN-AC first uses HIN-Embedding methods [8, 10, 30, 40] to obtain node embeddings, and then distinguishes different contributions of different nodes by computing the attention value of node embeddings when conducting weighted aggregation. This complement mechanism can be easily combined with an arbitrary HINs model making the whole system end-to-end. Note that there is a weak supervision loss in the process of node attribute completion. The weak supervision loss, combining with the model's prediction loss, are used to optimize the learning process of attribute completion.

Our attribute completion framework can give a performance improvement to GNN-based heterogeneous models. Specifically, when the target types of nodes being analyzed have no attributes (e.g., author nodes in DBLP), using the proposed framework to complete attributes for this type of nodes can greatly improve model performance. Even the target types of nodes already have attributes (e.g., movie nodes in IMDB), using the proposed framework to complete attributes for other types of nodes also can improve model performance. This is because the nature of the GNN-based model, i.e., completed attributes of other types nodes will be propagated and aggregated to the target types of nodes to help the model predict better.

The contributions of our work are summarized as follows:

(1) We propose the problem of missing whole attributes of some types of nodes which is the unique property of attributes missing in heterogeneous graphs. Previous heterogeneous methods usually do not consider the problem of missing attributes, or use handcrafted ways (e.g., summing or averaging) to replace missing attributes. To the best of our knowledge, this is the first attempt to complete node attributes for heterogeneous graphs.
(2) We propose a general framework for Heterogeneous Graph Neural Network via Attribute Completion (HGNN-AC), which solves the problem of missing attributes of some types of nodes in HINs. This framework addresses the deficiencies of the previous handcrafted approach by a learnable manner, and is easy to be combined with an arbitrary HINs model.
(3) We conduct extensive experiments on the DBLP, ACM and IMDB datasets to evaluate the performance of the proposed framework. The results show that the performance of existing models can be significantly improved after combining the proposed framework. We conduct a case study on the ACM dataset to further demonstrate the superiority of the proposed framework.

2 RELATED WORK

2.1 Graph Embedding

Graph embedding [3, 6, 12] aims to project nodes in a graph into a low-dimensional vector space, in which the representation of nodes can reflect the relationship between nodes, so as to retain the semantic information of nodes. This challenging topic has first been addressed in homogeneous graphs, such as DeepWalk [26], LINE [35], node2vec [13] and struc2vec [27]. Recently there are some embedding methods for heterogeneous graphs. In heterogeneous graphs, the most important thing is how to distinguish the heterogeneity of nodes and edges, and how to capture the rich semantic information brought by network heterogeneity. For example, metapath2vec [8] generates random node sequences guided by meta-paths [30, 33], and then feeds the sequences to skip-gram [21] model to generate node embeddings. HHNE [40] has further improved metapath2vec by embedding nodes into the hyperbolic space. Unlike Metapath2vec and HHNE, ESim [30] does not use random walk to obtain node sequences, instead it generates node

Table 1: Notations and Explanations.

Notations	Explanations
$\mathcal{V}$	The set of nodes
$\mathcal{E}$	The set of edges
$\mathcal{G}$	A heterogeneous graph
$\mathcal{V}^+$	The set of nodes with attributes in $\mathcal{V}$
$\mathcal{V}^-$	The set of nodes without attributes in $\mathcal{V}$
v	A node $v \in \mathcal{V}$
$\mathcal{N}_v^+$	The set of neighbors of node $v \in \mathcal{V}^+$
A	Topological structure
X	Node attributes
X^C	Node attributes after attribute completion
H	Node embedding based on topology A
Z	The final node embedding

embeddings by learning from sampled positive and negative meta-path instances.

All these above embedding methods can learn good representations of nodes, which can be used directly for downstream tasks. However, all of the graph embedding methods introduced above have the limitations of ignoring node attribute information.

2.2 Graph Neural Networks

Graph neural networks (GNNs) are introduced in [28] with the purpose of extending deep neural networks to deal with arbitrary graph-structured data. They are divided into two types: spectral domain [2, 7, 16, 19] and spatial domain [5, 14, 38, 41, 46]. The methods based on spectral domain adopt the spectral representation form of graphs. The methods based on spatial domain define convolutions directly on the graph, and aggregate feature information from spatial neighbors for each node, such as GraphSAGE [14] and GAT [38]. However, the above graph neural networks are used to deal with homogeneous graphs. Very recently, some studies have attempted to extend GNNs to heterogeneous graphs. For example, HAN [39] model based on the hierarchical attention, including node-level and semantic-level attentions, learns the importance between meta-path [30, 33] based nodes and the importance of different meta-paths respectively. Then HAN aggregates attributes from meta-path [30, 33] based neighbors in a hierarchical manner. MAGNN [11] model which contains the node content transformation to encapsulate input node attributes, the intra-meta-path aggregation to incorporate intermediate semantic nodes, and the inter-meta-path aggregation to combine messages from multiple meta-paths. GTN [44], which generates new graph structures by identifying useful connections between unconnected nodes on the original graph, can learn effective node embeddings on the new graphs in an end-to-end fashion.

Without exception, these methods mentioned above do not give an appropriate solution for the problem of missing whole attributes of some types of nodes in heterogeneous graphs. They only complete attributes by averaging or summing, or directly use one-hot vectors, which are not satisfactory. In this paper, we propose the attribute completion for heterogeneous graphs to fill this gap.

3 PRELIMINARY

We first give formal definitions of some important terminologies related to heterogeneous graphs. We then give some notations and explanations used throughout this paper.

Definition 1. Heterogeneous Graph. A heterogeneous Graph, denoted as $\mathcal{G}(\mathcal{V}, \mathcal{E}, F, R, \varphi, \phi)$, where $\mathcal{V}$ represents the set of nodes, $\mathcal{E}$ the set of edges, F the set of node types and R the set of edge types, where $|F| + |R| > 2$. Each node $i \in \mathcal{V}$ is associated with a node type mapping function $\varphi : \mathcal{V} \to F$, and each edge $e \in \mathcal{E}$ is associated with an edge type mapping function $\phi : \mathcal{E} \to R$.

As shown in Figure 1 (a), we construct a heterogeneous graph to model IMDB. It consists of three types of nodes (movie, actor and director) and two types of edges (movie-actor and movie-director).

Definition 2. Incomplete Attributes in Heterogeneous Graph. Given a heterogeneous graph $\mathcal{G}(\mathcal{V}, \mathcal{E}, F, R, \varphi, \phi)$, X is defined as node attributes. Incomplete Attributes means that $\exists F' \subset F$ and $F' \neq \emptyset$, in which each node $i \in \mathcal{V}$ associated with a node type mapping function $\varphi : \mathcal{V} \to F'$ has no attributes.

As shown in Figure 1 (a), in IMDB, only movie nodes have attributes, while directors and actors have no attributes.

Definition 3. Heterogeneous Graph Embedding. Given a heterogeneous graph $\mathcal{G}$, the task is to learn a d-dimensional node representation $h_v \in \mathbb{R}^d$ for all $v \in \mathcal{V}$ with $d \ll |\mathcal{V}|$ that are able to capture rich structural and semantic information involved in $\mathcal{G}$.

The notations are summarized in Table 1. Next we will introduce the framework of attribute completion for heterogeneous graphs.

4 HGNN-AC FRAMEWORK

In this section, we propose a node Attribute Completion framework for Heterogeneous Graph Neural Network (HGNN-AC). The framework follows the principle that the generated attributes of a no-attribute node (i.e., $v \in \mathcal{V}^-$) should come from the attributed nodes (i.e., $v \in \mathcal{V}^+$). The main idea is that we use topological information as guidance to calculate the contribution to the node $v \in \mathcal{V}^-$ among the directly connected neighbors $v' \in \mathcal{N}_v^+$, which can be a reference when we execute attribute completion.

4.1 Overview

The goal of the proposed framework is to solve the problem of some types of nodes having no attributes in heterogeneous graphs. It uses topological relationship between nodes as guidance to complete attributes for the no-attribute nodes (i.e., $v \in \mathcal{V}^-$) through attributed nodes (i.e., $v \in \mathcal{V}^+$). Figure 2 shows the proposed framework for node attribute completion. Given a heterogeneous graph $\mathcal{G}$ with only some types of nodes having attributes, HGNN-AC first computes nodes' embeddings H via network topological structure A, and then uses H to evaluate node topological relationship by exploiting the attention mechanism [1, 4, 20, 23, 37] to learn a sortable score to determine which directly connected attributed nodes are best suited to contribute attributes to no-attribute nodes. After knowing the best nodes, HGNN-AC completes attributes for nodes in set $\mathcal{V}^-$ by weighted aggregating the attributes of nodes in set $\mathcal{V}^+$ according to the score. In order to prevent overfitting,

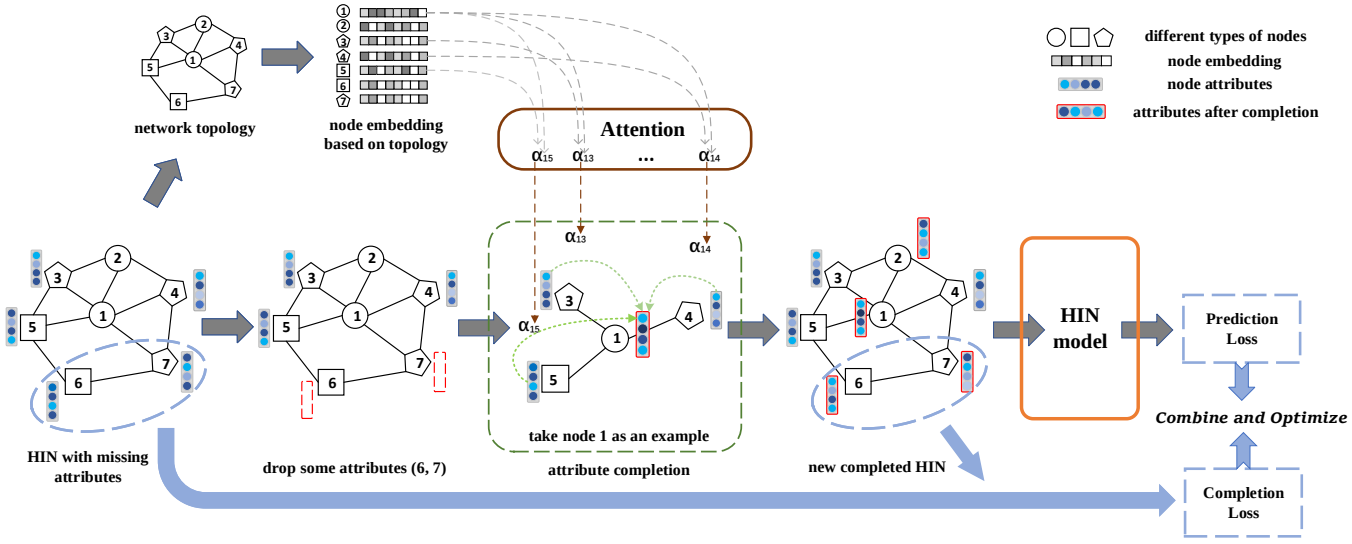


Figure 2: The overview of the proposed framework. Given a orginal heterogeneous graph with some types of nodes having no attributes as input, we first calculate node embeddings via network topology A. Then we randomly drop some attributes and perform attribute completion. The attribute completion process is essentially a weighted aggregation of directly connected neighbors' attributes, where weights are decided by the attention derived from node' embeddings. After attribute completion, we get a new heterogeneous graph with all node having attributes and send it to HINs model. Note that we have a completion loss between dropped attributes and reconstructed attributes. The combination of completion loss and model's prediction loss makes the whole model end-to-end.

and also to ensure that the attribute completion process can be guided, HGNN-AC first randomly drops some attributes of nodes in $\mathcal{V}^+$, and then reconstructs these attributes at the same time of the attribute completion process. In this way a completion loss can be calculated between dropped attributes and reconstructed attributes, making the attribute completion process guided. Finally, nodes with completed attributes, together with the network topology A, as a new graph, are fed to HINs models. The overall model can be optimized in an end-to-end manner by combining the loss of prediction and the loss of attribute completion as the final loss.

4.2 Pre-learning of Topological Embedding

In heterogeneous graphs, nodes have topology information (all nodes have) and attribute information (some types of nodes don't have, e.g., author and subject nodes have no attributes in the ACM dataset). Homophily is the principle that a contact between similar entities occurs at a higher rate than among dissimilar entities [22]. Due to the existence of network homophily [22, 24, 49], the topology and attribute information always express the similar or same semantics. For example, a scholar has a closer connection with the papers and venues in the field of his/her study, so scholar, paper and venue nodes share similar topologies. These nodes also have similar attributes because they are in the same field. With that in mind, we make an assumption that the relations of nodes' topology information can reflect well the relations of nodes' attribute information. In this paper, HGNN-AC uses existing heterogeneous graph embedding methods such as metapath2vec [8] or HHNE [40] to get node embeddings based on network topology. However, these skip-gram based methods always utilize single meta-path and may ignore some useful information. In order to get better embeddings, HGNN-AC first obtains more comprehensive node sequences by random walk according to the frequently used multiple meta-paths, and then feeds these sequences to the skip-gram model to learn node embeddings H .

4.3 Attribute Completion with Attention Mechanism

For the case that some types of nodes have no attributes, some previous studies [11, 39, 44] solve this problem by averaging aggregate attributes of the directly connected neighbors. But we noticed that the directly connected neighbors of each node play different roles and have different importance in attribute aggregation. This may be because these nodes are of different types, or because their local topologies are different, that is, the more neighbors a node has, the less important it is to each neighbor. After getting the embeddings of the nodes, HGNN-AC uses attention mechanism to automatically learn the importance of different direct neighbors, then aggregates attribute information for nodes in set $\mathcal{V}^-$ from their first-order neighbors in set $\mathcal{V}^+$.

Given a node pair (v, u) which are directly connected, the attention layer can learn the importance e_{vu} which means the contribution of node u to node v. The contribution of node u can be formulated as:

$$e_{vu} = att(h_v, h_u),$$

where h_v and h_u are the topological embeddings of nodes v and u, and $u \in \mathcal{V}^+$. $att(\cdot)$ denotes the function which can perform

attention mechanism. Note that $att(\cdot)$ is shared for all node pairs because our assumption is universal for all node pairs.

Furthermore, HGNN-AC adopts a masked attention mechanism which means we only calculate e_{vu} for nodes $u \in \mathcal{N}_v^+$, where $\mathcal{N}_v^+$ denotes the first-order neighbors of node v in set $\mathcal{V}^+$. It is obvious that first-order neighbors are likely to have more contributions, so this strategy can filter a large number of non-contributing nodes and reduce computation by adopting masked attention:

$$e_{vu} = \sigma(h_v^T W h_u),$$

where W is the parametric matrix, and σ an activation function.

After obtaining all direct neighbors' scores, a softmax function is applied to get normalized weighted coefficient a_{vu}:

$$a_{vu} = softmax(e_{vu}) = \frac{exp(e_{vu})}{\sum_{s \in \mathcal{N}_v^+} exp(e_{vs})}.$$

Then, HGNN-AC can perform weighted aggregation of attributes for node v according to weighted coefficient a_{vu}:

$$x_v^C = \sum_{u \in \mathcal{N}_v^+} a_{vu} x_u.$$

For node v, if none of its neighbor nodes has attributes (i.e. $\mathcal{N}_v^+ = \varnothing$), we set the attribute vector of node v to a zero vector. But in fact, this situation almost never occurs, so it has little impact on model performance.

Specially, the attention process is extended to a multi-head attention to stabilize the learning process and reduce the high variance (brought by the heterogeneity of networks), as done in many existing methods [38, 39]. In this way, the weighted aggregation of attributes for node v can be rewritten as:

$$x_v^C = mean(\sum_k^K \sum_{u \in \mathcal{N}_v^+} a_{vu} x_u),$$

where K means that we perform K independent attention process and $mean(\cdot)$ means we average the K results.

4.4 Dropping some Attributes

The proposed framework is for node attribute completion, and we expect the completed attributes can give a performance improvement to the HINs models. However, there is a question that how do we evaluate these new attributes generated by the proposed framework? In other words, how can we ensure the attribute completion process is learnable and the generated attribute is indeed correct?

In order to solve this problem, the proposed framework adopts a strategy of dropping some attributes. HGNN-AC first randomly drops some attributes of nodes in $\mathcal{V}^+$, and then reconstructs these attributes at the same time of the attribute completion process. In this way, a completion loss can be calculated between dropped attributes and reconstructed attributes, making the attribute completion process guided and learnable.

To be specific, for nodes in $\mathcal{V}^+$, HGNN-AC randomly divides them into two parts $\mathcal{V}_{drop}^+$ and $\mathcal{V}_{keep}^+$ according to a small ratio α, i.e. $|\mathcal{V}_{drop}^+| = \alpha|\mathcal{V}^+|$. HGNN-AC first drops attributes of nodes in $\mathcal{V}_{drop}^+$ and then reconstructs these attributes via attributes of nodes in $\mathcal{V}_{keep}^+$ by conducting attribute completion. The reconstructed attribute of node v can be formulated as:

$$x_v^C = mean(\sum_k^K \sum_{u \in \mathcal{V}_{keep}^+ \cap \mathcal{V}_i^+} a_{vu} x_u),$$

where $v \in \mathcal{V}_{drop}^+$, $u \in \mathcal{V}_{keep}^+$ and $\mathcal{V}_{keep}^+ \cap \mathcal{V}_i^+$ means that masked attention is also adopted here. The K and function $mean(\cdot)$ have the same meanings as above.

Our goal is that the reconstructed attributes are as close to the raw as possible. Here we introduce a weakly supervised loss to optimize the parameters of attribute completion. We use euclidean distance as the metric to design the loss function as:

$$\mathcal{L}_{completion} = \frac{1}{|\mathcal{V}_{drop}^+|} \sum_{i \in \mathcal{V}_{drop}^+} \sqrt{(X_i^C - X_i)^2}.$$

4.5 Combination with HIN Model

Now, we have completed attributes nodes in $\mathcal{V}^-$, and the raw attributes nodes in $\mathcal{V}^+$, then the new attributes of all nodes are defined as:

$$X^{new} = \{X_i^C, X_j | \forall i \in \mathcal{V}^-, \forall j \in \mathcal{V}^+\}.$$

The proposed framework keeps topological structure unchanged, so the new attributes X^{new}, together with network topology A, as a new graph, are sent to the HIN model:

$$\widetilde{Y} = \Phi(A, X^{new}),$$

$$\mathcal{L}_{prediction} = f(\widetilde{Y}, Y),$$

where Φ denotes an arbitrary HINs model, $\widetilde{Y}$ and Y are model's prediction and label respectively, and f is the loss function, depending on the specific task of the model.

Finally, we apply the proposed framework to a HIN model. The loss of label prediction and the loss of attribute completion are combined as the final loss of this new model. After that, the overall model can be optimized via back propagation in an end-to-end manner:

$$\mathcal{L} = \lambda \mathcal{L}_{completion} + \mathcal{L}_{prediction},$$

where λ is a weighted coefficient to balance these two parts.

5 EXPERIMENTS

5.1 Datasets

To evaluate the effectiveness of attribute completion by the proposed framework, we use three common HINs datasets. The statistics of the three datasets are summarized in Table 2.

(1) DBLP[1]. We extract a subset of DBLP with 14328 papers (P), 4057 authors (A), 20 venue (V) and 8789 terms (T). Authors are divided into four research areas according to the conferences they submitted. In this dataset, papers' attributes are bag-of-words representation of their keywords, authors' attributes are bag-of-words representations of keywords extracted from their published papers, terms' attributes are no computer-science-specialized pre-trained word vectors [25] and venues' attributes are one-hot vectors. Only papers' attributes are directly derived from the dataset.

[1] https://dblp.uni-trier.de/

(2) ACM[2]. We extract a subset of ACM using the same principle as HAN [39]. The papers are divided into three classes according to the conference they published. Then we construct a heterogeneous graph that comprises 4019 papers (P), 7167 authors (A) and 60 subjects (S). In this dataset, papers' attributes are bag-of-words representations of their keywords, authors' and subjects' attribute vectors come from the mean of their directly connected papers' attribute vectors. Only papers' attributes are directly derived from the dataset.
(3) IMDB[3]. We extract a subset of IMDB with 4780 movies (M), 5841 actors (A) and 2269 directors (D). Movies are divided into three classes according to their genres. In this dataset, movies' attributes are bag-of-words representations of their plots; actors' and directors' attribute vectors come from the mean of their directly connected movies' attribute vectors. Only movies' attributes are directly derived from the dataset.

Note that, while all types of nodes in the above datasets seems to have attributes, it is in fact the result of some handcrafted designs from previous works. That is, only papers in DBLP and ACM, and movies in IMDB have their own attributes that can express their real semantic information. In the following experiments, if the comparison experiment requires node attributes, we use these handcrafted attributes as input. But when conducting experiment of our framework, we get rid of these handcrafted attributes as our framework can complete missing attributes automatically.

For semi-supervised learning on the above three datasets, the labeled nodes are divided into training, validation, and testing sets by 10%, 10%, 80%, respectively.

5.2 Baselines

We combined the proposed framework with two state-of-the-art models, i.e., MAGNN and GTN, denoted as MAGNN-AC and GTN-AC respectively. We compare the performance of MAGNN-AC and GTN-AC with some existing methods, including MAGNN and GTN.

(1) Metapath2vec [8]: a skip-gram based heterogeneous embedding method which performs meta-path based random walk and utilizes skip-gram to generate embedding. We test on all meta-paths separately and report the best results.
(2) GCN [19]: a homogeneous GNN. This model uses an efficient layer-wise propagation rule that is based on a first-order approximation of spectral convolutions on graphs. We test GCN on serveral meta-path based homogeneous graphs and report the best results.
(3) GAT [38]: a homogeneous GNN. This model operates graph-structured data by leveraging masked self-attentional layers. We test GAT on serveral meta-path based homogeneous graphs and report the best results.
(4) HetGNN [45]: a heterogeneous GNN. This model jointly considers node heterogeneous contents encoding, type-based neighbors aggregation, and heterogeneous types combination.

[2]http://dl.acm.org/
[3]https://www.imdb.com/

Table 2: Statistics of datasets.

Datasets	Nodes	Edges	Attributes
DBLP	# author(A):4057 # paper(P):14328 # term(T):7723 # venue(V):20	# A-P:19645 # P-T:85810 # P-V:14328	# A:handcrafted # P:raw # T:handcrafted # V:handcrafted
ACM	# paper(P):4019 # author(A):7167 # subject(S):60	# P-P:9615 # P-A:13407 # P-S:4019	# P:raw # A:handcrafted # S:handcrafted
IMDB	# movie(M):4278 # director(D):2081 # actor(A):5257	# M-D:4278 # M-A:12828	# M:raw # D:handcrafted # A:handcrafted

(5) HAN [39]: a heterogeneous GNN. This model is based on the hierarchical attention mechanism. It generates node embedding by aggregating attributes from meta-path based neighbors in a hierarchical manner.
(6) MAGNN [11]: a heterogeneous GNN. This model realizes the prediction task through three steps (Node Content Transformation, Intra-meta-path Aggregation, Inter-meta-path Aggregation), and uses encoder functions to further improve performance.
(7) GTN [44]: a heterogeneous GNN. This model is able to learn a new graph structure which involves identifying useful meta-paths and multi-hop connections automatically. Due to the large memory and computing power this model required, we sampled multiple sub-networks and trained in batches according to the sampling method provided by [17].

5.3 Implementation Details

For the original settings/parameters of the model (MAGNN or GTN), we keep them unchanged. For the proposed framework, we use the following settings: We set the dropout ratio to 0.5 when conducting the masked attention and we extend masked attention to a multi-head attention with the number of attention head $K = 8$. We set divided ratio α of N^+ to 0.3 , and loss weighted coefficient λ to 0.5. The learning rate of the proposed framework's parameters is 0.005. For a fair comparison, we set the embedding dimension to 64 for all methods compared. The code and data of this work are available at https://github.com/liangchundong/HGNN-AC.

5.4 Node Classification

Node classification is a traditional task to evaluate the quality of the learned node embeddings. We perform node classification tasks under two types of datasets to compare the performance of different models. One type of dataset is that the nodes that need to be classified have no raw attributes. Here we choose the DBLP dataset for experimentation. Another type of dataset is that the nodes that need to be classified have raw attributes, while other types of nodes have no attributes. Here we choose the ACM and IMDB datasets for experimentation. We verify the effectiveness by combining the proposed framework with MAGNN and GTN respectively. First, we generate embeddings of the labeled nodes (i.e., authors in DBLP, authors in ACM and movies in IMDB). Then we feed the embeddings

Table 3: Results (%) on DBLP dataset on the node classification task.

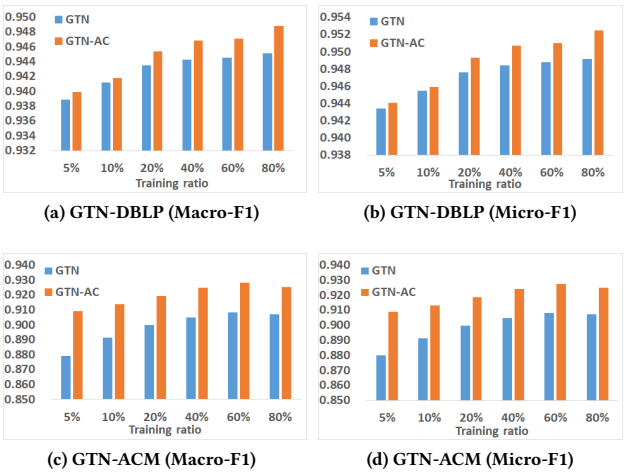
Datasets	Metrics	Training	Metapath2vec	GCN	GAT	HetGNN	HAN	MAGNN	MAGNN-AC
DBLP	Macro-F1	1%	88.76	86.99	32.68	86.61	89.37	92.45	**92.99**
		5%	90.49	89.03	57.20	90.60	90.83	92.44	**93.64**
		10%	91.09	89.53	64.57	91.09	91.24	92.44	**93.80**
		20%	91.50	90.06	66.92	91.72	91.69	92.53	**93.92**
		40%	92.55	90.37	73.23	92.03	91.84	92.97	**94.06**
		60%	93.25	90.57	77.17	92.26	92.01	93.30	**94.04**
		80%	93.48	90.74	78.20	92.39	92.15	93.77	**94.22**
	Micro-F1	1%	89.91	87.55	48.74	87.73	90.12	93.11	**93.51**
		5%	91.19	89.58	70.79	91.17	91.49	93.02	**94.09**
		10%	91.74	90.02	75.90	91.64	91.88	93.02	**94.22**
		20%	92.14	90.53	76.98	92.23	92.30	93.08	**94.34**
		40%	93.09	90.83	79.61	92.55	92.46	93.50	**94.46**
		60%	93.76	91.01	81.62	92.79	92.65	93.83	**94.46**
		80%	93.94	91.15	82.22	92.92	92.78	94.27	**94.61**

Table 4: Results (%) on ACM and IMDB datasets on the node classification task.

Datasets	Metrics	Training	Metapath2vec	GCN	GAT	HetGNN	HAN	MAGNN	MAGNN-AC
IMDB	Macro-F1	1%	35.23	39.72	49.52	37.32	52.49	50.78	**53.50**
		5%	42.37	42.95	53.08	42.70	56.16	54.28	**57.94**
		10%	44.29	43.70	53.61	45.68	57.02	56.39	**58.69**
		20%	46.42	44.75	54.81	48.92	57.61	58.11	**59.67**
		40%	47.70	45.26	55.09	51.61	57.75	59.39	**60.18**
		60%	48.25	46.72	55.71	53.00	57.66	59.97	**60.60**
		80%	48.73	47.13	55.40	53.24	57.23	60.02	**60.75**
	Micro-F1	1%	39.55	44.01	51.32	38.62	54.38	51.62	**54.74**
		5%	44.33	46.41	53.73	43.52	56.74	54.46	**58.26**
		10%	46.15	47.02	54.14	46.56	57.35	56.53	**58.97**
		20%	48.08	47.44	55.02	49.70	57.82	58.16	**59.84**
		40%	49.55	47.62	55.29	52.47	57.98	59.46	**60.38**
		60%	50.06	48.49	55.91	53.91	57.87	60.05	**60.79**
		80%	50.68	48.73	55.67	54.25	57.46	60.15	**60.98**
ACM	Macro-F1	1%	58.38	66.83	88.58	81.67	86.95	85.58	**88.95**
		5%	66.20	72.45	89.20	86.49	88.65	86.12	**90.31**
		10%	67.81	70.79	89.29	88.20	89.39	86.60	**90.29**
		20%	69.95	70.41	89.59	89.16	90.01	88.01	**91.51**
		40%	71.15	70.82	89.77	90.14	90.82	89.42	**92.75**
		60%	71.74	69.67	89.72	90.71	91.51	90.39	**93.46**
		80%	72.18	67.23	89.42	91.01	91.71	90.79	**93.79**
	Micro-F1	1%	63.18	71.74	88.60	82.20	87.37	85.95	**89.26**
		5%	68.88	74.95	89.10	86.55	88.62	86.24	**90.48**
		10%	70.29	74.10	89.19	88.18	89.32	86.67	**90.43**
		20%	72.12	74.02	89.47	89.12	89.89	88.08	**91.64**
		40%	73.17	74.57	89.65	90.11	90.73	89.48	**92.90**
		60%	73.65	74.10	89.60	90.64	91.37	90.42	**93.57**
		80%	74.14	72.69	89.29	90.93	91.56	90.80	**93.87**

to a linear support vector machine (SVM) [34] classifier with different training ratios from 1% to 80%. Note that for a fair comparison, only the nodes in the testing set are fed to the linear SVM, because in semi-supervised experiments, labels in training set and testing set have participated in the process of model training. Since the variance of graph-structured data can be very high, we repeat the process 5 times and report the averaged Macro-F1 and Micro-F1.

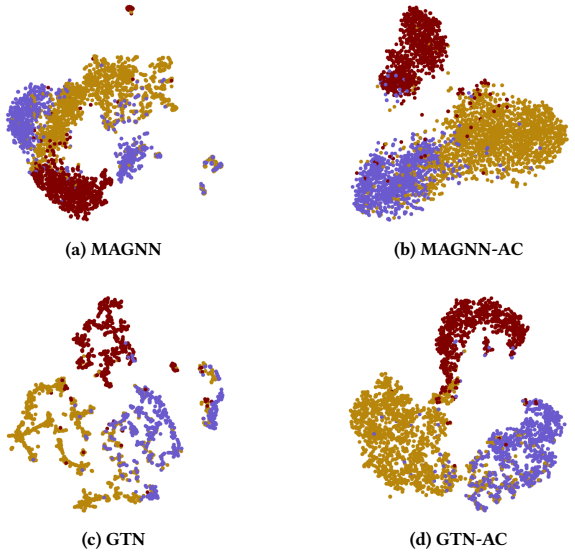
As shown in Table 3, MAGNN-AC performs best across different training ratios on the DBLP dataset after combining the proposed framework. On the DBLP dataset, researchers usually perform task analysis on author nodes with labels which have no raw attributes.



(a) GTN-DBLP (Macro-F1) (b) GTN-DBLP (Micro-F1)

(c) GTN-ACM (Macro-F1) (d) GTN-ACM (Micro-F1)

Figure 3: Comparison between GTN and GTN-AC.



(a) MAGNN (b) MAGNN-AC

(c) GTN (d) GTN-AC

Figure 4: Visualization of the paper nodes of embeddings in the ACM dataset. Different colors correspond to different research areas in ground truth.

After using the proposed framework to complete the attributes of authors, conferences and term nodes, MAGNN-AC is 0.34%-1.39% more accurate than MAGNN which exceeds that of all comparison algorithms. In another way, compared with MAGNN, MAGNN-AC reduces the error rate by 6.16%-22.86%.

In Table 4, we perform node classification tasks on the paper nodes in ACM and the movie nodes in IMDB. These two types of nodes have attributes, while other types of nodes have no raw attributes. After completing attributes by the proposed framework, MAGNN-AC also performs the best across different training ratios. On the ACM dataset, model performance has improved significantly (3.00%-4.24%) after attribute completion. That is, the error rate is reduced by 30.50%-50.08%. Most notably, MAGNN-AC outperforms HAN while MAGNN underperforms HAN. This is due to the fact that the proposed framework provides a better representation of node attributes. On the IMDB dataset, MAGNN-AC has a 0.63%-3.80% improvement, that is, it reduces the error rate of MAGNN by 1.60%-9.10%. Note that MAGNN-AC has a greater superiority when the training ratio decreases, which illustrates that the model will be more stable when combined with the proposed framework.

Here we present the comparative results between GTN and GTN-AC visually in the form of bar charts. From Figure 3, we can see that the performance of GTN-AC has been significantly improved on the ACM and DBLP datasets. It further proves the superiority of the proposed framework.

All above results show that better node attributes can be obtained through the proposed framework that uses the attention mechanism for attribute completion under the guidance of topological relations. For a node, its neighbors are often not equally important. Vanilla methods use handcrafted manner and ignore unequal importance of neighbors, making the generated attributes of no-attribute nodes inefficient and compromising model performance. Our framework can automatically learn the importance of different direct neighbors. Moreover, the proposed framework can solve the problem of more than one type of missing attributes of heterogeneous datasets.

5.5 Visualization

We conduct the task of visualization to show a more intuitively comparation. We learn the node embeddings of methods mentioned above (i.e., MAGNN, GTN, MAGNN-AC and GTN-AC) on the ACM dataset and project the embeddings into a 2-dimensional space. Then we utilize t-SNE [36] to visualize the paper embeddings in ACM and colored the nodes based on their classes.

As shown in Figure 4, MAGNN and GTN do not perform so well. Some papers belong to different classes are mixed with each other, and the boundary is blurry. After combining the proposed framework, MAGNN-AC shows a clearer boundary and denser cluster structures to distinguish different classes in visualization, and so is GTN-AC. It demonstrates that the accurate attribute information can make a significant contribution to heterogeneous graph analysis. However, MAGNN and GTN only use handcrafted methods to obtain the missing node attributes, resulting in poor model performance. Under the guidance of topological relations, the framework improves the performance of the model greatly by using the attention mechanism for attribute completion, and can effectively distinguish papers belonging to different research fields.

5.6 Case Study

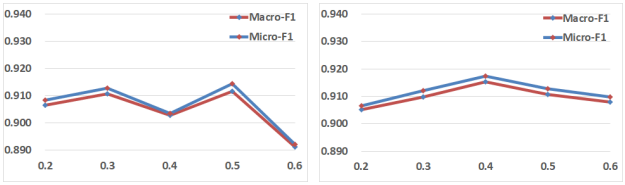
In order to prove the superiority of the proposed framework, we use MAGNN as an example to compare and analyze the ACM dataset by combining different node attributes processing methods. The ACM dataset contains three types of nodes: paper, author, and subject. Only papers' attributes are directly derived from the dataset. We carry out the following five designs according to different processing methods of attributes, and compare the performance of the models through the node classification task. We also repeat the process 5 times and report the averaged Macro-F1 and Micro-F1.

Table 5: Results (%) of node classification on ACM dataset using different attributes.

Datasets	Metrics	Training	MAGNN	MAGNN-onehot1	MAGNN-onehot2	MAGNN-AC1	MAGNN-AC2
ACM	Macro-F1	1%	85.58	83.16	86.71	88.24	**88.95**
		5%	86.12	85.00	87.24	89.60	**90.31**
		10%	86.60	85.86	87.76	89.61	**90.29**
		20%	88.01	88.14	89.66	90.86	**91.51**
		40%	89.42	89.87	91.13	92.18	**92.75**
		60%	90.39	90.84	92.01	93.03	**93.46**
		80%	90.79	91.15	92.56	93.55	**93.79**
	Micro-F1	1%	85.95	84.09	87.02	88.67	**89.26**
		5%	86.24	85.31	87.29	89.87	**90.48**
		10%	86.67	86.08	87.77	89.89	**90.43**
		20%	88.08	88.30	89.66	91.12	**91.64**
		40%	89.48	89.98	91.17	92.42	**92.90**
		60%	90.42	90.87	92.02	93.20	**93.57**
		80%	90.80	91.17	92.53	93.70	**93.87**

(1) MAGNN: The attributes of paper nodes are bag-of-words representations of their keywords. The attribute vectors of author and subject nodes come from the mean of the attribute vectors of paper nodes they are directly related to.
(2) MAGNN-onehot1: The attributes of paper nodes are bag-of-words representations of their keywords. The attributes of author nodes are one-hot vectors. The attribute vectors of subject nodes come from the mean of the attribute vectors of the paper nodes directly related to them.
(3) MAGNN-onehot2: The attributes of paper nodes are bag-of-words representations of their keywords. The attributes of author and subject nodes are one-hot vectors.
(4) MAGNN-AC1: The attributes of paper nodes are bag-of-words representations of their keywords. The attributes of the author nodes are obtained by using the proposed framework to complete. The attributes of subject nodes are one-hot vectors.
(5) MAGNN-AC2: The attributes of paper nodes are bag-of-words representations of their keywords. The attributes of author and subject nodes are both obtained by using the proposed framework to complete.

The results are shown in Tables 5. As we can see, MAGNN-AC2 achieves the best performance. MAGNN-AC2 has an obvious improvement, i.e. 3.00%-4.24% and 1.23%-3.19% more accurate than MAGNN and MAGNN-onehot2. MAGNN-AC1 is the second best, right after MAGNN-AC2. So we believe that using the mean of the attribute vectors of the directly connected paper nodes as the attributes of the author nodes and the subject nodes will result in similar node attributes and reduce the effective information provided to the model. For any author node, not all paper nodes connected to it are equally important. At the same time, it is not advisable to simply assume that there is no relationship between node attributes. Therefore, the proposed framework learns different importance of neighbors and obtains better node attributes to improve performance by using the attention mechanism for attribute completion under the guidance of topological relations, as shown in results.



(a) Divided ratio α **(b) Weighted coefficient λ**

Figure 5: Parameters Analysis.

5.7 Parameters Experiments

We investigate the sensitivity of parameters and report the scores of node classification (i.e., Macro-F1 and Micro-F1) on ACM dataset with the divided ratio of node attributes and weighted coefficient of completion loss in HGNN-AC. Each presented score is an average of the scores in different training proportions (explained in Section 5.4). We show the results in the form of line charts in Figure 5 .

(1) Divided ratio of node attributes α: We first test the effect of the divided ratio of node attributes. The result is shown in Figure 5 (a). We can see that with the growth of the divided ratio, the performance shows a trend of first rising and then slowly decreasing. This is because HGNN-AC needs a proper attribute divided ratio. Dropping too many attributes will result in insufficient completed attributes. Dropping too few attributes will result in biased loss calculations.
(2) Weighted coefficient of completion loss λ: In order to achieve the best performance of the model, we test the effect of the divided ratio of node attributes. The result is shown in Figure 5 (b). We also find that with the growth of the weighted coefficient, the performance shows a trend of first rising and then slowly decreasing. This is reasonable. Too small coefficient of completion loss will weaken the guiding role of completion loss on the process of attribute completion, while too large coefficient will weaken the role of prediction loss. Therefore, we choose an appropriate proportion to achieve the best overall performance.

6 CONCLUSION

In this paper, we propose HGNN-AC to solve the problem of missing attributes in heterogeneous graphs by a learnable manner. HGNN-AC includes two parts, pre-learning of topological embedding and attribute completion. We obtain node topological embedding by using HIN-Embedding methods. Then we use topological relationship between nodes as guidance to complete attributes for no-attribute nodes by weighted aggregation of the attributes of these attributed nodes with attention mechanism. Finally, we get an end-to-end model after combining the proposed framework with arbitrary HIN method. In experiments, we combine HGNN-AC with MAGNN and GTN model. The results on node classification demonstrate the superiority of the proposed framework over the state-of-the-arts. The task on visualization shows the effectiveness of HGNN-AC. The case study also demonstrates its good interpretability.

ACKNOWLEDGMENTS

This work is supported by National Natural Science Foundation of China (61772361, 61876128, 61972442), Key Research and Development Project of Hebei Province of China (20350802D), Natural Science Foundation of Hebei Province of China (F2020202040), and Natural Science Foundation of Tianjin of China (20JCYBJC00650).

REFERENCES

[1] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural Machine Translation by Jointly Learning to Align and Translate. In *ICLR*.
[2] Joan Bruna, Wojciech Zaremba, Arthur Szlam, and Yann LeCun. 2014. Spectral Networks and Locally Connected Networks on Graphs. In *ICLR*.
[3] Hongyun Cai, Vincent W. Zheng, and Kevin Chen-Chuan Chang. 2018. A Comprehensive Survey of Graph Embedding: Problems, Techniques, and Applications. *IEEE Trans. Knowl. Data Eng.* 30, 9 (2018), 1616–1637.
[4] Sneha Chaudhari, Varun Mithal, Gungor Polatkan, and Rohan Ramanath. 2019. An Attentive Survey of Attention Models. *arXiv preprint arXiv:1904.02874* (2019).
[5] Jie Chen, Tengfei Ma, and Cao Xiao. 2018. FastGCN: Fast Learning with Graph Convolutional Networks via Importance Sampling. In *ICLR*.
[6] Peng Cui, Xiao Wang, Jian Pei, and Wenwu Zhu. 2019. A Survey on Network Embedding. *IEEE Trans. Knowl. Data Eng.* 31, 5 (2019), 833–852.
[7] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. 2016. Convolutional Neural Networks on Graphs with Fast Localized Spectral Filtering. In *NIPS*. 3837–3845.
[8] Yuxiao Dong, Nitesh V. Chawla, and Ananthram Swami. 2017. metapath2vec: Scalable Representation Learning for Heterogeneous Networks. In *SIGKDD*. 135–144.
[9] David Duvenaud, Dougal Maclaurin, Jorge Aguilera-Iparraguirre, Rafael Gómez-Bombarelli, Timothy Hirzel, Alán Aspuru-Guzik, and Ryan P. Adams. 2015. Convolutional Networks on Graphs for Learning Molecular Fingerprints. In *NIPS*. 2224–2232.
[10] Tao-Yang Fu, Wang-Chien Lee, and Zhen Lei. 2017. HIN2Vec: Explore Meta-paths in Heterogeneous Information Networks for Representation Learning. In *CIKM*. 1797–1806.
[11] Xinyu Fu, Jiani Zhang, Ziqiao Meng, and Irwin King. 2020. MAGNN: Metapath Aggregated Graph Neural Network for Heterogeneous Graph Embedding. In *WWW*. 2331–2341.
[12] Palash Goyal and Emilio Ferrara. 2018. Graph Embedding Techniques, Applications, and Performance: A Survey. *Knowl. Based Syst.* 151 (2018), 78–94.
[13] Aditya Grover and Jure Leskovec. 2016. node2vec: Scalable Feature Learning for Networks. In *SIGKDD*. 855–864.
[14] William L. Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive Representation Learning on Large Graphs. In *NIPS*. 1024–1034.
[15] Jiawei Han. 2009. Mining Heterogeneous Information Networks by Exploring the Power of Links. In *ALT*. 3.
[16] Mikael Henaff, Joan Bruna, and Yann LeCun. 2015. Deep Convolutional Networks on Graph-Structured Data. *arXiv preprint arXiv:1506.05163* (2015).
[17] Ziniu Hu, Yuxiao Dong, Kuansan Wang, and Yizhou Sun. 2020. Heterogeneous Graph Transformer. In *WWW*. 2704–2710.
[18] Thomas N. Kipf and Max Welling. 2016. Variational Graph Auto-Encoders. *NIPS Workshop on Bayesian Deep Learning* (2016).
[19] Thomas N. Kipf and Max Welling. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *ICLR*.
[20] Junhyun Lee, Inyeop Lee, and Jaewoo Kang. 2019. Self-Attention Graph Pooling. In *ICML*. 3734–3743.
[21] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient Estimation of Word Representations in Vector Space. In *ICLR*.
[22] Mcpherson Miller and S. L. M. Cook. 2001. Birds of a Feather: Homophily in Social Networks. *Annual Review of Sociology* 27 (2001), 415–444.
[23] Volodymyr Mnih, Nicolas Heess, Alex Graves, and Koray Kavukcuoglu. 2014. Recurrent Models of Visual Attention. In *NIPS*. 2204–2212.
[24] Hongbin Pei, Bingzhe Wei, Kevin Chen-Chuan Chang, Yu Lei, and Bo Yang. 2020. Geom-GCN: Geometric Graph Convolutional Networks. In *ICLR*.
[25] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. GloVe: Global Vectors for Word Representation. In *EMNLP*. 1532–1543.
[26] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 2014. DeepWalk: Online Learning of Social Representations. In *SIGKDD*. 701–710.
[27] Leonardo Filipe Rodrigues Ribeiro, Pedro H. P. Saverese, and Daniel R. Figueiredo. 2017. *struc2vec*: Learning Node Representations from Structural Identity. In *SIGKDD*. 385–394.
[28] Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. 2009. The Graph Neural Network Model. *IEEE Trans. Neural Networks* 20, 1 (2009), 61–80.
[29] Michael Sejr Schlichtkrull, Thomas N. Kipf, Peter Bloem, Rianne van den Berg, Ivan Titov, and Max Welling. 2018. Modeling Relational Data with Graph Convolutional Networks. In *ESWC*. 593–607.
[30] Jingbo Shang, Meng Qu, Jialu Liu, Lance M. Kaplan, Jiawei Han, and Jian Peng. 2016. Meta-Path Guided Embedding for Similarity Search in Large-Scale Heterogeneous Information Networks. *arXiv preprint arXiv:1610.09769* (2016).
[31] Chuan Shi, Yitong Li, Jiawei Zhang, Yizhou Sun, and Philip S. Yu. 2017. A Survey of Heterogeneous Information Network Analysis. *IEEE Trans. Knowl. Data Eng.* 29, 1 (2017), 17–37.
[32] Yizhou Sun and Jiawei Han. 2012. Mining Heterogeneous Information Networks: A Structural Analysis Approach. *SIGKDD Explor.* 14, 2 (2012), 20–28.
[33] Yizhou Sun, Jiawei Han, Xifeng Yan, Philip S. Yu, and Tianyi Wu. 2011. PathSim: Meta Path-Based Top-K Similarity Search in Heterogeneous Information Networks. *Proc. VLDB Endow.* 4, 11 (2011), 992–1003.
[34] Johan A. K. Suykens. 2001. Support Vector Machines: A Nonlinear Modelling and Control Perspective. *Eur. J. Control* 7, 2-3 (2001), 311–327.
[35] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. 2015. LINE: Large-scale Information Network Embedding. In *WWW*. 1067–1077.
[36] Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing Data using t-SNE. *Journal of Machine Learning Research* 9, 2605 (2008), 2579–2605.
[37] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is All You Need. In *NIPS*. 5998–6008.
[38] Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph Attention Networks. In *ICLR*.
[39] Xiao Wang, Houye Ji, Chuan Shi, Bai Wang, Yanfang Ye, Peng Cui, and Philip S. Yu. 2019. Heterogeneous Graph Attention Network. In *WWW*. 2022–2032.
[40] Xiao Wang, Yiding Zhang, and Chuan Shi. 2019. Hyperbolic Heterogeneous Information Network Embedding. In *AAAI*. 5337–5344.
[41] Felix Wu, Amauri Souza, Tianyi Zhang, Christopher Fifty, Tao Yu, and Kilian Weinberger. 2019. Simplifying Graph Convolutional Networks. In *ICML*. 6861–6871.
[42] Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and Philip S. Yu. 2021. A Comprehensive Survey on Graph Neural Networks. *IEEE Trans. Neural Networks Learn. Syst.* 32, 1 (2021), 4–24.
[43] Zhitao Ying, Jiaxuan You, Christopher Morris, Xiang Ren, William L. Hamilton, and Jure Leskovec. 2018. Hierarchical Graph Representation Learning with Differentiable Pooling. In *NIPS*. 4805–4815.
[44] Seongjun Yun, Minbyul Jeong, Raehyun Kim, Jaewoo Kang, and Hyunwoo J. Kim. 2019. Graph Transformer Networks. In *NIPS*. 11960–11970.
[45] Chuxu Zhang, Dongjin Song, Chao Huang, Ananthram Swami, and Nitesh V. Chawla. 2019. Heterogeneous Graph Neural Network. In *SIGKDD*. 793–803.
[46] Jiani Zhang, Xingjian Shi, Junyuan Xie, Hao Ma, Irwin King, and Dit-Yan Yeung. 2018. GaAN: Gated Attention Networks for Learning on Large and Spatiotemporal Graphs. In *UAI*. 339–349.
[47] Muhan Zhang and Yixin Chen. 2018. Link Prediction Based on Graph Neural Networks. In *NIPS*. 5171–5181.
[48] Jie Zhou, Ganqu Cui, Zhengyan Zhang, Cheng Yang, Zhiyuan Liu, and Maosong Sun. 2018. Graph Neural Networks: A Review of Methods and Applications. *arXiv preprint arXiv:1812.08434* (2018).
[49] Jiong Zhu, Yujun Yan, Lingxiao Zhao, Mark Heimann, Leman Akoglu, and Danai Koutra. 2020. Generalizing Graph Neural Networks Beyond Homophily: Current Limitations and Effective Designs. In *NIPS*. 7793–7804.