Neural Computing and Applications (2020) 32:16647–16679
https://doi.org/10.1007/s00521-020-04908-5

REVIEW

Network representation learning: a systematic literature review

Bentian Li[1] · Dechang Pi[1,2]

Received: 4 September 2019 / Accepted: 6 April 2020 / Published online: 20 April 2020

Abstract
Omnipresent network/graph data generally have the characteristics of nonlinearity, sparseness, dynamicity and heterogeneity, which bring numerous challenges to network related analysis problem. Recently, influenced by the excellent ability of deep learning to learn representation from data, representation learning for network data has gradually become a new research hotspot. Network representation learning aims to learn a project from given network data in the original topological space to low-dimensional vector space, while encoding a variety of structural and semantic information. The vector representation obtained could effectively support extensive tasks such as node classification, node clustering, link prediction and graph classification. In this survey, we comprehensively present an overview of a large number of network representation learning algorithms from two clear points of view of homogeneous network and heterogeneous network. The corresponding algorithms are deeply analyzed. Extensive applications are introduced in an all-round way, and related experiments are conducted to validate the typical algorithms. Finally, we point out five future promising directions for next research in terms of theory and application.

Keywords Representation learning · Network embedding · Network data mining · Deep neural network

1 Introduction

Big data have aroused extensive attention of industry and academia [1–3]. It is worth noting that most of the current studies are based on the assumption of independence between data, which leads to the fact that mining large-scale data is far from enough. In fact, there is a general relation between these data. For example, large-scale image and text can be constructed as the network for realizing the multi-information fusion [4]. Polypharmacy side effects of the drug–drug interactions may be affected with protein–protein interactions, drug–protein target interactions [5]. That is to say, there not only exist big data with a single data type including image, text, speech and video, but also exists the ubiquitous network data, such as Google knowledge graph, protein–protein interaction network, gene network, brain network, Internet, social network, multimedia network, molecular compound network and traffic network. In particular, the recent online social network represented by Twitter, WeChat and Facebook has entered the era of billions of nodes, making it more urgent for researchers to study large-scale network data. However, different from image and text data, network data often have the characteristics of nonlinearity, sparseness, dynamicity and heterogeneity, which bring many challenges to deep and advanced data mining task.

Data representation has always been the top priority of research in the field of data mining and machine learning. The so-called data representation (or feature representation) means using a set of symbols or numbers, i.e., feature vectors, to describe an object and to be able to reflect some of its characteristics, structure and distribution. Good data representation can often greatly reduce the volume of data, while retaining the essential characteristics of the original data and capturing the posterior distribution of potential interpretable data and maintaining robustness against random noise. The success of machine learning algorithms

✉ Dechang Pi
dc.pi@nuaa.edu.cn

Bentian Li
lbt@nuaa.edu.cn

1 College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing 211106, China

2 Collaborative Innovation Center of Novel Software Technology and Industrialization, Nanjing 211106, China

generally depends on data representation. This is because different representations can entangle and hide more or less the distinctive explanatory factors of variation behind the data [6]. At the initial stage of development, data representation is mainly dependent on feature engineering by using domain knowledge or expert knowledge to create features. However, it is also the biggest constraint on machine learning applications. It is not only time-consuming and laborious, but also requires a lot of transcendental experience and complicated manual design. This has led to a study of representation learning.

Representation learning from data is an important research hotspot in the field of machine learning and pattern recognition [6]. Especially in recent years, representation learning has seen a surge of research in the well-known journals and conferences in the field of artificial intelligence and data mining, such as TPAMI, TNNLS, TKDE, KDD, NIPS, ICML, AAAI, IJCAI and ICLR. Representation learning can learn high-level abstract feature representation directly from original low-level perception data, which is more conducive to subsequent data analysis and processing [7]. That is, the original feature engineering is transformed into the process of automatic machine learning. In particular, deep learning is a typical method of representation learning, which can automatically extract the appropriate feature or obtain the multiple levels of representation from the data. Deep learning has been successfully applied in speech recognition and signal processing [8–10], image recognition [11, 12], object recognition [13–18] and other fields, especially in the field of natural language processing (NLP) [19–22].

In the upsurge of representation learning, network representations learning (a.k.a. network embedding) is in full swing. It is precisely because the traditional method of extracting structural information from network data usually depends on topological statistical information such as degree, aggregation coefficient or the limitations of well-designed manual features. This brings about many drawbacks as mentioned above. Therefore, inspired by the successful application of representation learning in the NLP fields (e.g., word2vec [19, 23]), network representations learning aims to automatically learn a variety of features from the given network-structured data to low-dimensional vector space, while encoded a variety of structural and semantic information. When the vector data representation is obtained, the problem of network data mining can be solved directly by the off-the-shelf machine learning method. As expected, network representations learning has been proven to be useful in many tasks of data mining and machine learning such as link prediction [24–30], node classification [26–28, 30–41], network reconstruction [25], recommendation [25, 29, 42], visualization [26, 33, 34, 36, 38, 43] and community detection [44, 45].

Network representation learning has become one of the indispensable and hot topics in domain of data mining and machine learning in recent years. We conduct academic search in important academic library such as Springer Link, IEEE xplore, DBLP and Elsevier Science Direct through a given keyword about network embedding and network representation learning and find that more than 400 related articles have been published in the past 5 years. Figure 1 shows the number of published papers on this topic from 2010 to March 2020.

It can be seen that from 2014 to 2019, the number of papers has increased year by year. Therefore, the research trend on this topic is growing. We believe it is the right time to have a paper to summarize this series of the representative work published in important journals and top conference papers in the field of data mining and machine learning. It is worth noting that we mainly focus on network embedding for network inference and analysis task rather than graph embedding for dimensionality reduction [46]. Therefore, the classical nonlinear dimensionality reduction methods such as ISOMAP [47], locally linear embedding (LLE) [48], Laplacian eigenmaps (LE) [49] and the latest manifold learning algorithms for dimensionality reduction such as the semi-supervised out-of-sample interpolation (SOSI) [50], the robust local manifold representation (RLMR) [51] and the projective unsupervised flexible embedding models with optimal graph (PUFE-OG) [52] will not be involved. The major difference between network embedding and graph embedding has detailed discussed by the recent work [53]. This article will not repeat. To the best of our knowledge, this is the first work to survey the network representation learning comprehensively and systematically, although there have already been a few survey works related to network representation

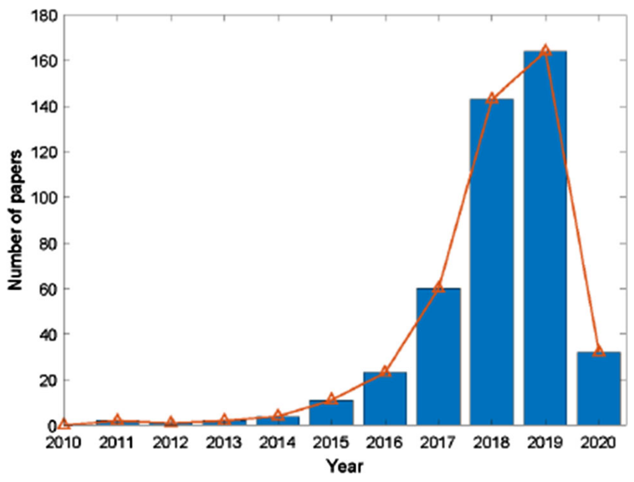

Fig. 1 Statistics of papers by publication date

learning problem. For example, Moyano [54] summarizes network representations learning from the point of view of network science. This work is valuable and multipurpose. However, as a new research direction of machine learning, it is still necessary to review from two perspectives of computer science and data science. Moreover, even though some literature has done relative work in the view of computer science, such as [53, 55–57], they all have the following defects more or less. Firstly, all the frameworks of the articles are not classified from different type of network data. It is clear that homogeneous network representations learning is greatly different from heterogeneous network representations learning. Secondly, there is no clear classification for different object-oriented embedding in homogeneous network embedding. Most of the articles are merely a general introduction to network embedding methods. Obviously, node embedding, subgraph embedding, whole network embedding and dynamic network embedding should have separate categories. Thirdly, most of the studies do not divide the embedding algorithm of heterogeneous networks comprehensively.

1.1 Contributions

Synthetically speaking, this review consists of three main contributions.

1. Our work presents an unambiguous taxonomy of network embedding from the two perspective of homogeneous network and heterogeneous network. Apparently, heterogeneous network representation learning is more challenging.
2. We propose a systematic taxonomy for homogeneous network representation learning and heterogeneous network representation learning. For homogeneous network, the embedding methods for different targets are summarized, respectively. Specifically, node embedding, subgraph embedding, whole network embedding and dynamic network embedding are listed. For heterogeneous network, the embedding approaches are clearly divided into two categories: knowledge graph embedding and heterogeneous information network embedding.
3. We summarize and list the publicly available datasets used in network embedding, which can facilitate study for other researchers. We comprehensively summarize the related applications of network embedding and empirically evaluate the surveyed methods on several publicly available real-world network datasets. Finally, we point out five future directions in the view of theory and application, which provide valuable suggestions for future researchers.

1.2 Framework and organization of the survey

First of all, we survey the number of representative papers over the past ten years in different sub-directions of network representation learning. The results are shown in Fig. 2. We can find the current important points in the era of network representation learning are homogeneous node embedding and heterogeneous knowledge graph embedding. To provide a comprehensive overview, we cover all sub-directions. Our comprehensive framework of reviewing the network embedding is shown in Fig. 3.

The structure of this paper is organized as follows. In Sect. 2, we introduce the concepts and definitions related to network embedding. Section 3 introduces the homogeneous network embedding systematically. We analyze the algorithm in detail from four aspects of node embedding, subgraph embedding, whole network embedding and dynamic network embedding. Section 4 introduces heterogeneous network embedding comprehensively in the view of knowledge graph embedding and heterogeneous information network (HIN) embedding. Section 5 reviews extensively application of network representation learning. Section 6 includes some experiments on several representative datasets for testing some typical methods. Finally, we draw our conclusions and point out the five future research directions and prospects in terms of theory and application.

2 Definitions

In this section, we introduce some concepts and definitions related to network embedding.

2.1 Definitions of network data

Definition 1 (*Network*) Given a network $G = (V, E)$, V denotes the sets of vertex (node) and E denotes the sets of

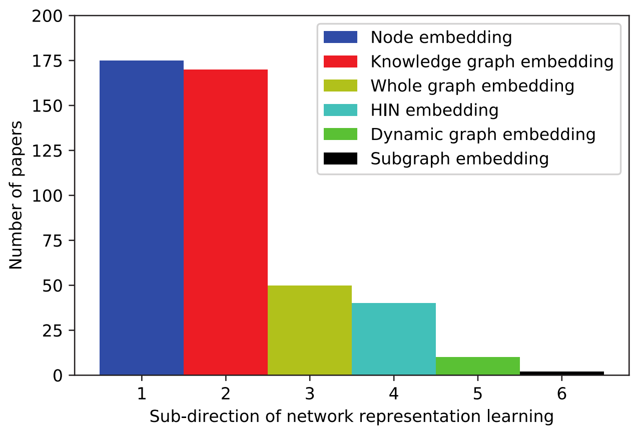

Fig. 2 Comparison of sub-direction research

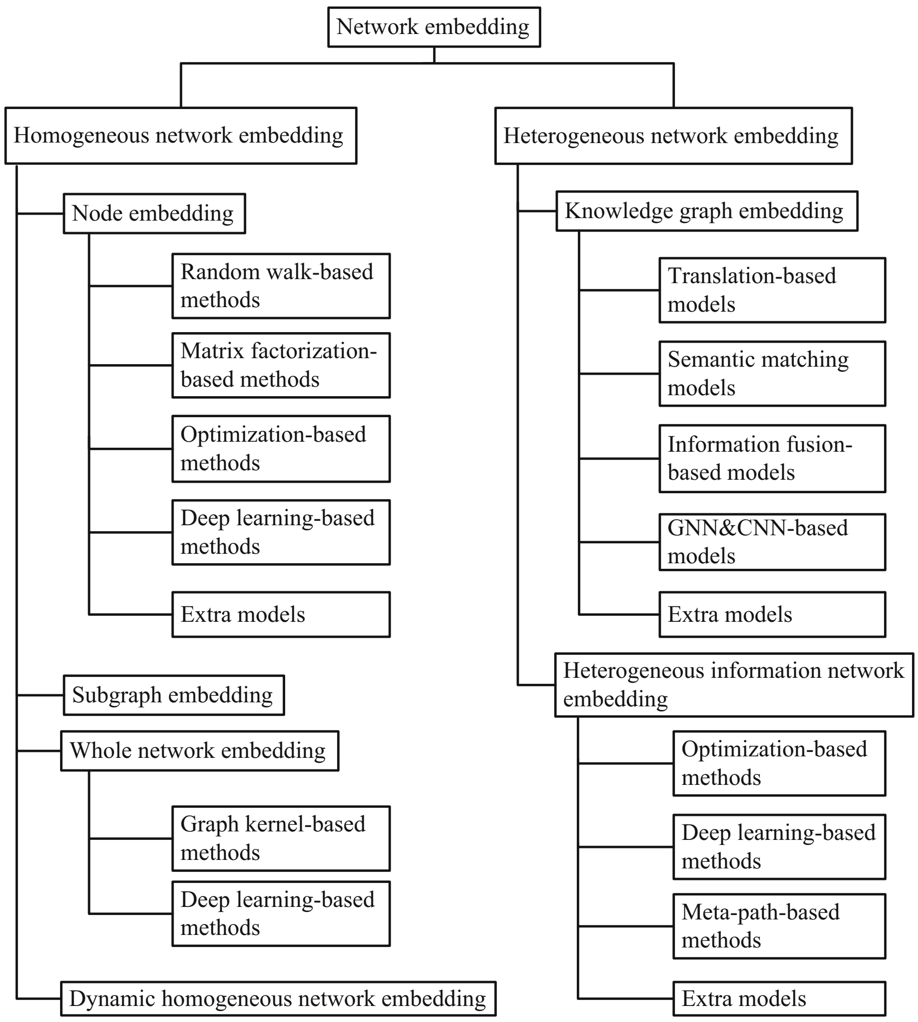
Network embedding
Homogeneous network embedding
Node embedding
Random walk-based methods
Matrix factorization-based methods
Optimization-based methods
Deep learning-based methods
Extra models
Subgraph embedding
Whole network embedding
Graph kernel-based methods
Deep learning-based methods
Dynamic homogeneous network embedding
Heterogeneous network embedding
Knowledge graph embedding
Translation-based models
Semantic matching models
Information fusion-based models
GNN&CNN-based models
Extra models
Heterogeneous information network embedding
Optimization-based methods
Deep learning-based methods
Meta-path-based methods
Extra models

Fig. 3 The comprehensive organization structure of network embedding

edges between the vertex. According to the directivity of edges, it is divided into undirected network and directed network. If the weight of edges is considered, the network can be also divided into weighted and unweighted network.

Definition 2 (*Homogeneous network, heterogeneous network*) Given $G = (V, E)$, $\emptyset : V \rightarrow A$ and $\Psi : E \rightarrow R$ are type mapping functions for nodes and edges, respectively. When $|A| > 1$ and/or $|R| > 1$, the network is called a heterogeneous network. Otherwise, it is a homogeneous network.

Definition 3 (*Knowledge graph*) A knowledge graph is defined as a directed graph, whose nodes are entities and edges are subject–property–object triple facts. Each edge of the form (head entity, relation, tail entity) (denoted as $\langle h, r, t \rangle$.) indicates a relationship of r from entity h to entity t. Note that the entities and relations in a knowledge graph are usually of different types. Therefore, knowledge graph can be viewed as an instance of the heterogeneous network.

Definition 4 (*Static network, dynamic network*) The static network is the opposite direction to dynamic network. A dynamic homogeneous or heterogeneous network at time step t is defined as $G^t = \{V^t, E^t\}$, where $V^t = \{v_1^t, v_2^t, \ldots, v_N^t\}$ denotes a set of nodes and N is the number of the nodes. E^t is the set of edges between the nodes. When the time t is a fixed value, the dynamic network becomes a static network. Without special explanation, this article generally refers to the static network.

2.2 Structure and other information in network

Definition 5 (*First-order proximity*) The first-order proximity is the weight w_{uv} between the edges of nodes u and v. It is the local pairwise proximity between two vertices. If there is no edge between nodes, then its first-order proximity is 0 [33].

Definition 6 (*Second-order and high-order proximity*) The second-order proximity is the similarity of the first-order proximity of node pairs such as nodes u and v [33]. For example, we can let $s_u = \left(w_{u1}, w_{u2}, w_{u3}, \ldots, w_{u|V|}\right)$ denote the first-order proximity of u with all the other vertices. The second-order proximity between u and v can be computed by the cosine similarity between s_u and s_v [39]. Similar to the second-order proximity definition, the high-order (X-order) proximity is the similarity by calculating the lower-order ($X-1$ order) proximity.

Definition 7 (*Meta-path of the network*) Given a heterogeneous information network $G = (V, E)$, a meta-path π is a sequence of node types $v_1, v_2, \ldots, v_n$ and/or edge types $e_1, e_2, \ldots, e_{n-1}$: $\pi = v_1 \xrightarrow{e_1} \cdots v_i \xrightarrow{e_i} \cdots \xrightarrow{e_{n-1}} v_n$.

2.3 Problem definition of network embedding

Problem 1 (*Homogeneous network embedding*) Given the input of a graph G, homogeneous network embedding is designed to map node or subgraph or whole network into a low-dimensional space $R^d (d \ll |V|)$. The embedding vectors are expected to preserve the various features of the network as much as possible.

Problem 2 (*Heterogeneous network embedding*) For network G, the problem of heterogeneous network embedding aims to map the different type of entities or relations into a low-dimensional space $R^d (d \ll |V|)$. The embedding vectors are expected to preserve the structure and semantics in G as rich as possible.

Problem 3 (*Dynamic network embedding*) Given a series of dynamic networks $\left\{G^1, G^2, \ldots, G^T\right\}$, dynamic homogeneous or heterogeneous network embedding aims to learn a mapping function $f^t : v_i \rightarrow R^d$. The objective function is to preserve the structure and semantics between v_i and v_j, with the evolution of the network at the time.

3 Homogeneous network embedding

The overall architecture of homogeneous network embedding could be presented as Fig. 4. The original raw network data are fed to network embedding algorithm, and the vectors obtained can be utilized to various graph analysis tasks by using the off-the-shelf machine learning algorithm. For homogeneous network embedding, we analyze the algorithm in detail from four aspects of node embedding, subgraph embedding, whole network embedding and dynamic network embedding. Besides, we summarize some representative homogeneous node embedding method, which is shown as Table 1. The graph type, evaluation task, advantages and disadvantages, time complexity are included. Fore time complexity, considering that some works lack of the technique details, therefore we mainly summarize the time complexity of some representative network embedding methods, where N and $|E|$ are the number of nodes and edges, respectively. I is the number of iterations. d is the representation dimension. a is the length of node attributes. R is the number of samples per vertex. L is the expected sample length.d' is the maximum number of hidden layer nodes in DNN. k is the number of layer. C, H and F are the weight matrix of the $W^0 \in R^{C \times H}$ and $W^1 \in R^{H \times F}$. F and F' are the number of input and attention head features. M is the average number of neighborhoods. Tables 2 and 3 show the open-source datasets used for node embedding and whole network embedding. The details are as follows.

3.1 Node embedding

3.1.1 Random walk-based methods

Motivation Random walk has been widely used in network data analysis to capture the topological structure information [77]. As the name suggests, random walk chooses a certain vertex in the graph for the first step and then randomly migrates through the edges. By Analogy, the walking path obtained when performing random walks can be taken as the sentence in the corpus, the vertices in the network could be regarded as vocabulary in the text corpus. Then, combining with mature natural language processing technology (e.g., word2vec [19, 23]), nodes could be mapped to low-dimensional vector space. Therefore, some methods are proposed based on the similarity of the node

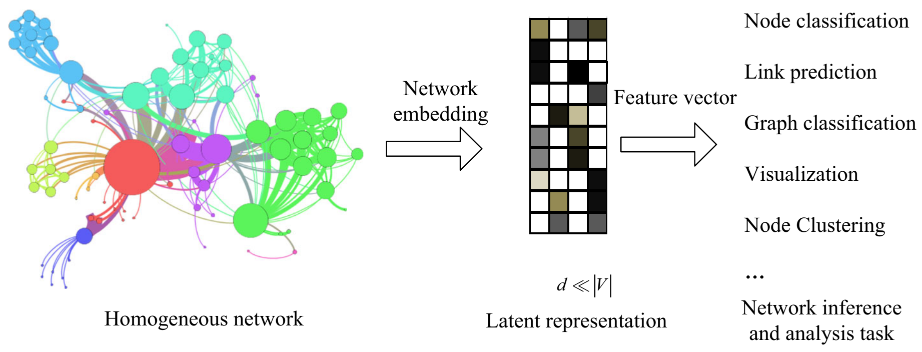

Fig. 4 Architecture of homogeneous network embedding

Table 1 Summary of representative homogeneous node embedding methods

Method type	Method	Year	Graph type	Evaluation task	Advantage	Disadvantage	O_{time}
Random walk	Deepwalk [32]	2014	Undirected, unweighted, non-attributed	Node classification	Comparatively effective	Mostly capture local structure, randomness	$O(dN\log N)$
	DDRW [37]	2016		Network classification			–
	TriDNR [38]	2016		Node classification, network visualization			$O(dN\log N)$
	Node2vec [27]	2016		Node classification, link prediction			$O(dN)$
	Struct2vec [30]	2017		Node classification			$\mathrm{O}(kN^3)$
	CARE [58]	2018		Node classification, link prediction			–
	DiaRW [59]	2019		Node classification, link prediction			–
Optimization	Line [33]	2015	(Un)directed, (un)weighted, non-attributed	Node classification, visualization	Direct modeling of specific structure	Designing the proper objective function is challenging; Optimization may fall into local optimum	$O(d\|E\|)$
	APP [29]	2017		Link prediction, node recommendation			$O(NdRLI)$
Matrix factorization	GraRep [34]	2015	Undirected, weighted, non-attributed	Node classification, clustering, visualization	Mostly capture global structure or high-order proximity	High time and space costs	$O(N\|E\| + dN^2)$
	TADW [35]	2015	Undirected, unweighted, attributed	Vertex classification			$\mathrm{O}(N\|E\| + dI\|E\| + daIN + d^2IN)$
	HOPE [25]	2016	Directed, weighted, non-attributed	Graph reconstruction, link prediction, node recommendation			$O(d^2I\|E\|)$
	MMDW [36]	2016	Undirected, unweighted, non-attributed	Vertex classification, visualization			–
	M-NMF [39]	2017		Node clustering, classification			$O(dIN^2)$

Table 1 (continued)

Method type	Method	Year	Graph type	Evaluation task	Advantage	Disadvantage	O_{time}
Deep learning	SDNE [26]	2016	Undirected, (un)weighted, non-attributed	Network reconstruction, node classification, link prediction, visualization	Effective and robust, automatic feature learning	Difficulty in tuning parameters; Poor interpretability; Time-consuming for large-scale graph data	$O(d'IN)$
	DNGR [43]	2016	Undirected, weighted, non-attributed	Clustering, visualization			$O(d'IN)$
	GCN [60]	2017	Undirected, unweighted, attributed	Node classification			$O(\|E\|CHF)$
	DGCN [61]	2018		Node classification			–
	LGCL [62]	2018		Node classification			–
	ANE [63]	2018		Network visualization, node classification			–
	GATs [64]	2018		Node classification			$O\left(NFF' + \|E\|F'\right)$
	GraphGAN [65]	2018	Undirected, unweighted, non-attributed	Link prediction, node classification, recommendation			$O(IN\log N)$
	ARGA [66]	2018	Undirected, unweighted, attributed	Link prediction, node clustering, graph visualization			–
	H-GCN [67]	2019		Node classification			$O(N \log N + MN)$
	ProGAN [68]	2019		Node classification, node clustering, visualization			–
	HesGCN [69]	2020		Node classification			–

Table 2 Summary of datasets used for homogeneous node embedding

Data source	Description	Nodes	Edges	Sites of datasets
Social network	Flickr	1,715,256	22,613,981	http://socialnetworks.mpi-sws.org/data-imc2007.html
	Youtube	1,138,499	2,990,443	
	epinions	75,879	508,837	http://konect.uni-koblenz.de/networks/soc-Epinions1
	Twitter	465,017	834,797	http://konect.uni-koblenz.de/networks/munmuntwittersocial
	Tencent Weibo	1,944,589	50,655,143	http://www.kddcup2012.org/c/kddcup2012-track1/data
DBLP	Author citation	524,061	781,109	http://arnetminer.org/citation
	Paper citation	20,580,238	4,191,677	
SNAP datasets	Facebook	4039	88,234	http://snap.stanford.edu/data
	arXiv ASTRO-PH	18,722	198,110	
	Amazon network	262,111	1,799,584	http://snap.stanford.edu/data/amazon0302.html
BioGRID interaction database	Protein–protein interactions	19,706	390,633	https://thebiogrid.org/download.php
Social media	BlogCatalog	10,312	333,983	http://socialcomputing.asu.edu/pages/datasets
		5196	171,743	https://github.com/xhuang31/AANE_MATLAB
	Political blog network	1222	16,715	http://www-personal.umich.edu/~mejn/netdata/
Co-occurrence network of words	Wikipedia	4777	184,812	http://www.mattmahoney.net/dc/textdata
		2405	17,981	http://www.cs.umd.edu/~sen/lbc-proj/LBC.html#
CiteSeerX data	CiteSeer-M10	10,310	77,218	http://citeseerx.ist.psu.edu/
Paper citation network	Cora	2277	5214	https://people.cs.umass.edu/~mccallum/data.html
		23,166	91,500	http://konect.uni-koblenz.de/networks/subej_cora
		2708	5429	http://www.cs.umd.edu/~sen/lbc-proj/LBC.html#
	Arxiv	5242	28,980	http://snap.stanford.edu/data/ca-GrQc.html
	HepTh	1038	1990	https://snap.stanford.edu/data/cit-HepTh.html

Table 3 Summary of datasets used for homogeneous whole network embedding

Data source	Dataset	Graph	Class	Avg. node	References
Source: https://ls11-www.cs.tu-dortmund.de/staff/morris/graphkerneldatasets					
Bioinformatics datasets	MUTAG	188	2	17.9	[70]
	PTC	344	2	25.2	[71]
	D&D	1178	2	284.3	[72]
	PROTEINS	1113	2	39.1	[73]
	ENZYMES	600	6	32.6	
	NCI1	4110	2	29.8	[74]
	NCI109	4127	2	29.6	
Social network datasets	COLLAB	5000	3	74.49	[75]
	IMDB-BINARY	1000	2	19.77	[76]
	IMDB-MULTI	1500	3	13	
	REDDIT-BINARY	2000	2	429.61	
	REDDIT-MULTI-5K	5000	2	508.5	
	REDDIT-MULTI-12 K	11,929	11	391.4	

sequence and the text sequence. The representative methods are as follows.

The pioneering work of this kind of method was DeepWalk [32], which was inspired by word2vec algorithm [19, 23] of modeling language. In word2vec, the short sequences of words in text corpus could be embedded in continuous vector space. Similar to word2vec, DeepWalk generated sequences of nodes from a stream of truncated random walks on the graph. Each path sampled from the graph corresponds to a sentence from the corpus, where a node corresponded to a word. Then, the skip-gram model [23] was applied on the paths to maximize the probability of observing a node's neighbor conditioned on its embedding. The hierarchical softmax [78] technology was to approximate the probability distribution. Node representation can be obtained, just like getting the representation of a word.

However, this model has different limitations. Firstly, researchers found that different random walk strategies may produce different node representations in DeepWalk. Therefore, Node2vec [27] improved DeepWalk method by employing biased strategy of breadth-first sampling (BFS) and depth-first sampling (DFS). By adjusting search bias parameters, the model can obtain a good node feature vector. Secondly, DeepWalk cannot obtain discriminative representation for specific node classification tasks. Thus, DDRW [37] was aimed at learn the latent space representations that well captured the topological structure and meanwhile were discriminative for the network classification task. The model extended DeepWalk by jointly optimizing the classification objective and the objective of embedding entities in a latent space. Thirdly, DeepWalk did not model the node label information. Therefore, TriDNR [38] learnt the network representation from three parties, namely node structure, node content and node label. Specifically, the inter-node relationship, node-word correlation and label-word correspondence were modeled by maximizing the probability of observing surrounding nodes given a node in random walks, maximizing the co-occurrence of word sequence given a node and maximizing the probability of word sequence given a class label. Finally, DeepWalk showed fail in structural equivalence tasks. Hence, Struct2vec [30] assessed structural similarity using a hierarchy to measure similarity at different scales and constructed a multilayer graph to encode the structural similarities and generate structural context for nodes via a biased random walk. Then, the skip-gram model was also adopted to learning latent representation of node, which maximized the likelihood of its context in a sequence.

Discussions and insights Although the method based on random walk has achieved certain effects, it cannot ignore the limitations brought by randomness. Therefore, designing a structure-oriented walking strategy will be very important for performance improvement, such as capturing the community structure (see [58]) and preserving the scale-free feature of real-world networks (see [59, 79]). Researcher also finds the limitation of adopting short random walks to explore the local neighborhoods of nodes and non-convex optimization which can become stuck in local minima in the model of DeepWalk and Node2vec, HARP [80] captured the global structure of the input network, by recursively coalescing the input graph into smaller but structurally similar graphs. By learning graph representation on these smaller graphs, a good initialization scheme for the input graph was derived. This multilevel paradigm can improve the graph embedding methods (e.g., DeepWalk, Node2vec) for yielding better node embedding. Besides, as we can see from Table 1, the time complexity of the model Node2vec in above method is linear with respect to the number of nodes N. Therefore, this model can be scalable for large networks.

3.1.2 Optimization-based methods

Motivation Optimization-based algorithm aims to design a proper objective function modeling various structural and semantic information of the graph, then minimize or maximize the objective function by optimization method to obtain node vector representation using a variety of optimization methods, such as Line [33] model using negative sampling [19] + asynchronous stochastic gradient algorithm (ASGD) [81] and APP [29] model using negative sampling + stochastic gradient descent (SGD) [82].

The representative Line model designed two optimization functions by first-order proximity [24] and second-order proximity [33]. The author proposed to minimize the Kullback–Leibler (KL) divergence of joint probability distribution and empirical probability distribution. To obtain the embedding of vertex, an edge-sampling strategy was proposed to minimize either one of the objective function of first-order proximity and second-order proximity. APP model captured both asymmetric proximities and high-order similarities between node pairs. To preserve the asymmetric proximity, each vertex v needed to have two different roles, the source role and the target role, represented by vector $\overrightarrow{s_v}$ and $\overrightarrow{t_v}$, respectively. The global objective function was designed by the inner product of s_u and t_v, which can represent the proximity of vertex pair (u, v).

Discussions and insights From Table 1, we can find that the Line model is linear with respect to the number of edges E, which can easily scale up to networks with millions of vertices and billions of edges. At the same time, it is not difficult to find that this type of method mainly depends on the modeling ability of the objective function. Therefore, the optimization-based method can further improve the performance of the algorithm by designing more accurate objective functions.

3.1.3 Matrix factorization-based methods

Motivation A network or graph can usually be expressed in some forms of matrix, such as adjacency matrix and Laplacian matrix. Therefore, it is an intuitively way to design node embedding method by matrix factorization. Matrix factorization (MF), such as singular value decomposition (SVD) [83] and eigenvalue decomposition, is an important method in math, which has many successful applications in many fields [84–89]. For node embedding methods based on MF, representative algorithms are as follows.

For modeling the different structural information and obtaining the node embedding in framework of MF, GraRep [34], HOPE [25] and M-NMF [39] have made some attempts. Specifically, GraRep presented a node embedding method by decomposing the global structural information matrix, which integrated various k-step relational information. HOPE [25] proposed a high-order proximity embedding which decomposed the high-order proximity matrix rather than adjacency matrix using a generalized SVD. M-NMF [39] proposed a modularized nonnegative matrix factorization model, which preserved both the microscope structure of first-order and second-order proximities and mesoscopic structure of community feature. Note that the real-world networks also contain rich information in addition to the structure. Therefore, for modeling such auxiliary information, TADW [35] introduced text features of vertices into matrix factorization framework. MMDW [36] incorporated the labeling information into vertex representations in an unified learning framework based on matrix factorization.

Discussions and insights The approach of GraRep [34] just performs linear dimension reduction by employing SVD, resulting in the loss of more nonlinear information. Typical extension work is DNGR [43], which will be analyzed in the next section. Considering the flexibility of matrix decomposition, the frameworks of TADW [35], MMDW [36] and M-NMF [39] are well scalable and can easily integrate other information. HOPE [25] could also be extended by designing new higher-order proximity matrices.

3.1.4 Deep learning-based methods

Motivation In recent years, deep learning technology has shown the strong ability to model the data in many fields (see [6]). Therefore, the network data are no exception. For example, autoencoder is first applied to network presentation learning which is influenced by the mature unsupervised learning ability (see [26, 43]). The variant of convolutional neural networks is also designed to network data (see [60]). Besides, attention models have been successfully adopted in many computer vision tasks, including object detection [90]. Inspired by it, the attention mechanism has also recently been applied to network data (see [64]). In addition, the principle of adversarial learning has been widely applied in many fields, such as image classification [91, 92], sequence generation [93], dialog generation [94] and information retrieval [95]. The typical method is generative adversarial networks (GAN) [96] of which the framework consists of two components, i.e., a generator and a discriminator. GAN can be formulated as a minimax adversarial game, where the generator aims to map data samples from some prior distribution to data space, while the discriminator tries to tell fake samples from real data. Considering this ubiquitous game problem, this thought for network representation learning has also made significant research progress (see [63, 65]). The specific branches are as follows.

3.1.4.1 Autoencoder-based methods There are two main categories of these methods: classical autoencoder-based and graph autoencoder-based algorithms. The representative methods of the first category are SDNE [26] and DNGR [43]. SDNE [26] proposed a modified conventional autoencoder-based model, which simultaneously optimizes the first-order and second-order proximity to characterize the local and global network structure. The first-order proximity was preserved by idea of Laplacian Eigenmaps [49]. The second-order proximity was preserved by the extension of traditional deep autoencoder. Finally, the intermediate layer of deep autoencoder was used as the final node representation. DNGR [43] captured the structural information of the graph via determining the positive pointwise mutual information (PPMI) matrix [97] and learnt the stacked denoising autoencoders to obtain the node embedding. Note that the DNGR model is task-agnostic. This results in the fact that the performance of specific task cannot be guaranteed. Therefore, DNNNC [41] proposed an efficient classification-oriented node embedding method for improving the performance of node classification by extending DNGR in framework of deep learning. The representative algorithms in second class are GAE [98] and ARGA [66]. The former can be regarded as the pioneering work of such methods. The model designed a graph convolutional network (GCN) [60] encoder and a simple inner product decoder to perform the unsupervised learning on graph-structured data. However, the model has at least three defects. The first defect is that the model cannot obtain robust graph representation. In order to remedy this defect, ARGA [66] proposed an adversarial training scheme to regularize the latent codes, which is obtained by GAE. The second defect is that GAE is not designed for specific task of graph analysis. Therefore, DAEGC [99] proposed clustering-oriented node embedding method via jointly optimizing to simultaneously obtain both graph embedding and graph clustering result. The third flaw is that the decoder module is too simple and loses too much structural information. Thus, recent work TGA [100] proposed a triad decoder via modeling the triadic closure property that is fundamental in real-world networks.

3.1.4.2 Graph convolutional network-based methods GCN [60] can be considered as the most representative work, which adopted an efficient variant of convolutional neural networks on graph-structured data. The main convolutional architecture was via a localized first-order approximation of spectral graph convolutions. However, the researchers found that the model had some representative limitations. Firstly, GCN cannot model the global information of the network. Thus, DGCN [61] extended the GCN model with dual graph convolutional networks, i.e., the graph adjacency matrix-based convolution ConvA and positive pointwise mutual information (PPMI) matrix-based convolution ConvP. In this way, the local-consistency-based knowledge and the global-consistency-based knowledge in the data are embedded. Secondly, GCN will need excessive memory and computational resources when the graph has a large amount of nodes. To overcome this limitation, LGCL [62] proposed a large-scale learnable GCN via the learnable graph convolutional layer and an efficient subgraph training strategy. Thirdly, GCN is too shallow to capture more information. Therefore, H-GCN [67] proposed a deep hierarchical GCN by introducing coarsening layers and symmetric refining layers to increase the receptive field. Finally, GCN has poor extrapolating ability. To improve this capability, HesGCN [69] proposed a more efficient convolution layer rule by optimizing the one-order spectral graph Hessian convolutions, which is a combination of the Hessian matrix and the spectral graph convolutions.

3.1.4.3 Graph attention-based methods Recently, graph attention networks (GATs) were proposed by Veličković et al. [64], which was an attention-based architecture to perform node classification. The main idea was to compute the hidden representations of each node, by attending over its neighbors, following a self-attention strategy.

Discussions and insights For autoencoder-based methods and GAN-based method, they are all unsupervised learning model. The representations learned can accomplish a variety of tasks, such as node clustering and link prediction. It is noteworthy that the learning ability of simple application of autoencoder to network data is limited, because of the particularity of network data as stated above. Therefore, the design of graph-oriented autoencoder is an urgent research issue. Although the work mentioned above has been covered, there is still a lot of potential development space. E.g., most of the variational graph autoencoder and its variants all use the KL divergence as the similarity measure between the distributions, which is not the true metrics. Therefore, the next generation of the graph autoencoder will be inspired by the recent work of Wasserstein autoencoders [101]. The GAN-based method also has some limitations such as poor stability, which maybe come from the characteristics of GAN itself. Like GAN and its variants which have developed very prosperously in the field of image processing, there is still a great potential for the development of GAN for graph data. For graph convolutional network-based methods and graph attention-based methods, they are specific for the task of semi-supervised node classification. For GCN method, we can find its time complexity is linear in the number of graph edges E in Table 1. We also note that the time complexity of GAT is on par with GCN. Hence, both of the them are suitable for large-scale networks. Although GCN and its variants are inspired by convolutional neural network (CNN), are they necessary for graph data? Is there a linear but very efficient method? Similar to attention models in other field such as NLP, the graph attention-based embedding methods also can be extended to related tasks such as graph classification as a useful mechanism to improve performance (see Sect. 3.3.2).

3.1.4.4 GAN-based methods The representative work of this kind of method mainly includes adversarial network embedding (ANE) [63], GraphGAN [65] and ProGAN [68]. ANE mainly consisted of two components, i.e., a structure preserving component and an adversarial learning component. Specifically, an inductive DeepWalk was proposed for structure preserving. The adversarial learning component consisted of two parts, i.e., a generator $G(\cdot)$ and a discriminator $D(\cdot)$. It was acting as a regularizer for learning stable and robust feature extractor, which was achieved by imposing a prior distribution on the embedding vectors through adversarial training. The parameterized function $G(\cdot)$ was shared by both the structure preserving component and the adversarial learning component. GraphGAN was an another innovative graph representation learning framework with GAN idea. For a given vertex, the generative model aims to fit its underlying true connectivity distribution over all other vertices and produces “fake” samples to fool the discriminative model. In addition, with the generative capabilities of GAN networks, ProGAN proposed to generate proximity between different nodes which can help to discover the complicated underlying proximity to benefit network embedding.

3.1.5 Extra models

Different from the above method, GraphWave [102] learnt a multidimensional structural embedding for each node based on the diffusion of a spectral graph wavelet centered at the node. The method made it possible to learn nodes' structural embeddings via spectral graph wavelets [103]. The key is to treat the wavelet coefficients as a probability distribution and characterize the distribution via empirical characteristic functions.

3.2 Subgraph embedding

Subgraph embedding aims to encode a set of nodes and edges into a low-dimensional vector. The representative work was Subgraph2vec [104], which encoded semantic substructure dependencies in a continuous vector space. Firstly, a rooted subgraph around every node in a given graph was extracted by using WL relabeling strategy [105]. Then, the algorithm proceeded to learn its embedding with the radial skip-gram model.

3.3 Whole network embedding

In this section, we focus on the whole graph embedding, which represented a graph as one vector. When such vectors are obtained, graph-level classification is often involved. The actual network is usually rich in a variety of structural information. In such situation, embedding a whole graph needs to capture the property of a whole graph. Therefore, it is a challenging task to design efficient algorithm. In this review, we summarize the representative state-of-the-art graph kernel methods and latest deep learning methods for whole graph embedding to perform graph classification task.

3.3.1 Graph kernel-based methods

Typical methods in this category are graph kernel [106] and deep graph kernel (DGK) [76]. Graph kernel defined a distance measure between subgraphs by the function. DGK was proposed to leverage the dependency information between substructures by learning their latent representations. DGK was one of the state-of-art methods in graph kernel family, which has been proven to outperform three popular graph kernel methods of Graphlet kernels [107], Weisfeiler–Lehman kernel [108] and shortest-path graph kernels [109].

Discussions and insights The main shortcomings of these types of methods are handcrafted feature. Besides, their similarity is directly calculated based on global graph structures and it is computationally expensive to calculate graph distance and prediction rules are hard to interpret, because graph features are numerous and implicit. And, it cannot capture the important substructures, which is useful for graph classification.

3.3.2 Deep learning-based methods

As mentioned above, deep learning has achieved great success in task of node embedding. For graph classification tasks, the corresponding deep-based method is also showing a booming trend. The typical method consists of two branches: CNN-based methods and attention-based methods. The details are as follows.

3.3.2.1 CNN-based methods This kind of method can be divided into two categories: conventional CNN-based methods and spectral convolution based approaches. For the first type of method, the representation work is PSCN [110]. Niepert et al. [110] proposed the method of PSCN, which can be taken as the first work for learning convolutional neural networks for arbitrary graph classification. The model designed a general approach to extracting locally connected regions from graphs via analogizing image-based convolutional networks that operate on locally connected regions of the input. The final proposed architecture consisted of node sequence selection, neighborhood graph construction, graph normalization and convolutional architecture. However, researchers found that the PSCN model has at least three drawbacks, such as the selection of neighborhood, losing the complex subgraph feature and special labeling approach. Thus, recent work NgramCNN [111] proposed to tackle the above shortcomings. It consisted of three core components: (1) the n-gram normalization, which sorted the nodes in the adjacency matrix of a graph and produced the n-gram normalized adjacency matrix, (2) a special convolution layer, called diagonal convolution, which extracted the common subgraph patterns found within local regions of the input graph and (3) a stacked deep convolution structure, which was built on top of diagonal convolution and repeated by a series of convolution layers and a pooling layer. The researchers also noted the high complexity of the data preprocessing (such as using graph canonization tool NAUTY [112]) in the PSCN model. To avoid this question, the recent model DGCNN [113] proposed a pure neural network architecture for graph classification in an end-to-end fashion. The brief processing steps are as follows. An input graph of arbitrary structure was first passed through multiple graph convolution layers where node information was propagated between neighbors. Then, the vertex features were sorted and pooled with a SortPooling layer, and passed to traditional CNN structures to learn a predictive model.

For spectral convolution based method, early attempts can be traced back to the first spectral graph CNN model [114] and later extended model [115]. However, because the above models usually contain highly complex operations, they cannot be well applied to large-scale networks. In order to make up for the above defects and efficiently perform the task of graph classification, the model of GCN combined with other effective mechanisms to obtain graph-level embedding becomes very attractive and representative. Some recent works such as dense differentiable pooling [116] and self-attention pooling [117] have been introduced the pooling operation to GCN for obtaining whole graph embedding from node embedding, which have achieved good performance in graph classification task.

3.3.2.2 Attention-based methods Similar to attention-based methods for node classification discussed above (see Sect. 3.1.4), attention thoughts have recently been applied to graph classification tasks. The representative work is graph attention model (GAM) [118], a novel RNN model, proposed to focus on small but informative parts of the graph, which avoided noise in the rest of the graph. The attention mechanism was mainly used to guide the walk toward more informative parts of the graph.

Discussions and insights We could find that many algorithms for graph classification are based on the conventional CNN model, such as NgramCNN [111] and DGCNN [113]. Their essence is still to use the traditional CNN to conduct feature processing and transformation. Although these models have proven effective, their ability to capture complex structures is inherently insufficient. To better identify complex structures in networks, recent work—Capsule Graph Neural Network (CapsGNN) [72] has proposed to graph classification task via leveraging the power of Capsule Neural Network (CapsNet) [119] in field of image processing. The model has achieved good classification results. Therefore, how to learn the most discriminative information from the complex structure space is the key to the problem.

3.4 Dynamic homogeneous network embedding

We can clearly observe that almost all the existing network embedding methods focus on static networks while ignoring network dynamics [25–27, 32–39, 43, 63, 65]. Therefore, network embedding for dynamic network is an open question, which is gradually gaining the attention of researchers. Consider the challenges of the network over time, designing efficient dynamic network embedding algorithm is not a trivial problem. Representative methods include DHPE [120], DynamicTriad [121] and Dynamic GCN [122]. The first two methods are non-deep learning methods. Specially, DHPE [120] proposed a dynamic network embedding method, which aimed to preserve the high-order proximity. The generalized SVD (GSVD) was firstly adopted to preserve the high-order proximity. Then, the researcher proposed a generalized eigen perturbation approach to incrementally update the results of GSVD. In this way, the dynamic problems are transformed into eigenvalue updating problems. DynamicTriad [121] designed a dynamic network embedding algorithm via modeling triadic closure process, which enables our model to capture network dynamics. The last method is deep learning-based approach named Dynamic GCN, which combines long short-term memory networks [123] and GCN to learn long short-term dependencies together with graph structure. The performance of the model has been proved to be outstanding in vertex-based semi-supervised classification and graph-based supervised classification tasks.

Discussions and insights Intuitively, how to model dynamic characteristics and update node embedding is the key to the problem of dynamic network embedding. With the development of deep learning, the dynamic network embedding method based on neural network will make greater progress.

4 Heterogeneous network embedding

Similar to homogeneous network, heterogeneous network also exists widely in the real world as mentioned earlier. Typical representatives of heterogeneous network are heterogeneous knowledge graph (KG) [124] and heterogeneous information network (HIIN) [125–129]. Taking online social network as an example, it could be composed of different types of multimodal nodes (e.g., users, image, text and videos) and complex relations (e.g., social or cross-media similarity relations) [130]. So, compared to homogeneous network embedding, heterogeneous network embedding is more challenging issue because of the heterogeneity and high complexity. The architecture of heterogeneous social network representation learning is shown as Fig. 5. Heterogeneous network embedding could map different heterogeneous objects in heterogeneous network into a unified latent space so that objects from different spaces can be directly compared. Consequently, heterogeneous network analysis tasks become feasible. For heterogeneous network embedding, we review the algorithm in detail from two aspects of knowledge graph embedding and heterogeneous information network embedding, in view of the rapid development and importance of knowledge graph embedding technology. Besides, we summarize the knowledge graph embedding method

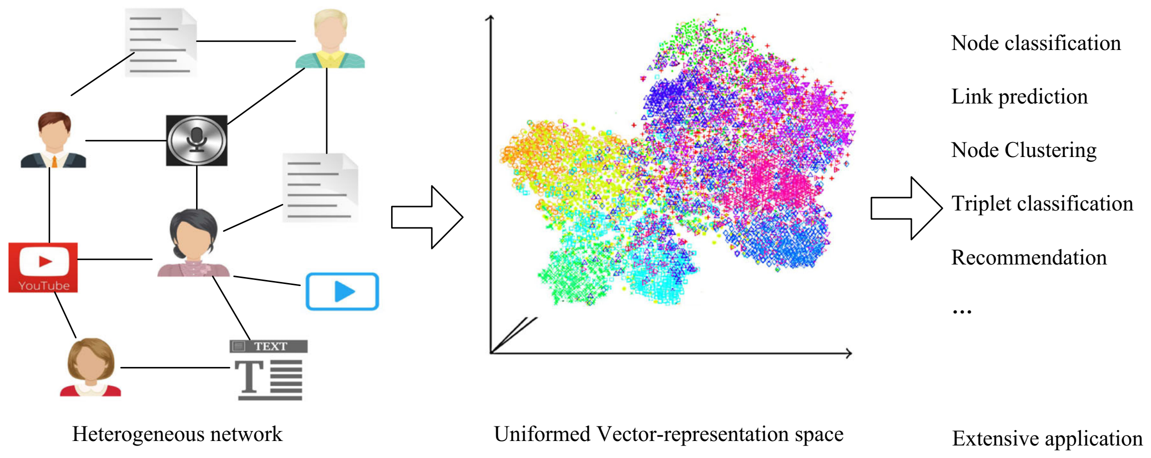

Fig. 5 Architecture of heterogeneous social network embedding

comprehensively, which is shown as Table 4. The score function, prerequisite and time complexity are included. Here, d and k are the dimensionality of entity and relation embedding space, respectively. $\sigma(\cdot)$ and ω denote the nonlinear activation function and convolutional filters. Tables 5 and 6 show the open-source datasets used in knowledge graph embedding and heterogeneous information network embedding. The specific contents are as follows.

4.1 Knowledge graph embedding

4.1.1 Translation-based models

The core of translation-based model is based on translation distance. In the early exploration, researchers used intuitive distance model, such as SE [132]. However, the model loses too much information and results in poor performance. After that, it is more extensive to adopt the translation model introduced in this section.

Table 4 Summary of representative knowledge graph embedding methods

Method	Year	Score functions	Prerequisite	O_{time}
RESCAL [131]	2011	$h^T M_r t$	$h, t \in R^d$, $M_r \in R^{d \times d}$	$\mathrm{O}(d^2)$
SE [132]	2012	$\|M_{rh} h - M_{rt} t\|_{l1}$	$h, t \in R^d$, $r \in R^k$, $M_{rh}, M_{rt} \in R^{d \times d}$	$\mathrm{O}(d^2)$
NTN [133]	2013	$u_r^T \tanh\left(h^T W_r^{[1:k]} t + V_{rh} h + V_{rt} t + b_r\right)$	$h, t \in R^d$, $u_r, b_r \in R^k$, $W_r^{[1:k]} \in R^{k \times d \times d}$, $V_{rh}, V_{rt} \in R^{k \times d}$	$O(kd^2)$
TransE [124]	2013	$\|h + r - t\|_2^2$	$h, r, t \in R^d$	$O(d)$
TransH [134]	2014	$\left\|\left(h - w_r^T h w_r\right) + r - \left(t - w_r^T t w_r\right)\right\|_2^2$	$h, r, t, w_r \in R^d$	$O(d)$
TransR [135]	2015	$\|M_r h + r - M_r t\|_2^2$	$h, r, t \in R^d$, $r \in R^k$, $M_r \in R^{k \times d}$	$O(kd)$
DistMult [136]	2015	$h^T \mathrm{diag}(r) t$	$h, r, t \in R^d$	$O(d)$
TransD [137]	2015	$\|M_{rh} h + r - M_{rt} t\|_2^2$	$h, t \in R^d$, $r \in R^k$, $M_{rh}, M_{rt} \in R^{k \times d}$	$O(kd)$
TransA [138]	2016	$\|h + r - t\|$	$h, r, t \in R^d$	$O(d)$
TranSparse [139]	2016	$\left\|M_r^h\left(\theta_r^h\right) h + r - M_r^t\left(\theta_r^t\right) t\right\|_{l1/2}^2$	$h, t \in R^d$, $r \in R^k$, $M_r^h\left(\theta_r^h\right), M_r^t\left(\theta_r^t\right) \in R^{k \times d}$	$O(kd)$
TKRL [140]	2016	$\|M_{rh} h + r - M_{rt} t\|$	$h, t \in R^d$, $r \in R^k$, $M_{rh}, M_{rt} \in R^{k \times d}$	$O(kd)$
HOLE [141]	2016	$r^T (h * t)$	$h, r, t \in R^d$	$O(d)$
DKRL [142]	2016	$\|h_d + r - t_d\| + \|h_d + r - t_s\| + \|h_s + r - t_d\|$	$h, r, t \in R^d$	$O(d)$
ManifoldE [143]	2016	$\left\|\|h + r - t\|_2^2 - D_r^2\right\|^2$	$\mathrm{h, r, t} \in R^d$	$O(d)$
ComplEx [144]	2016	$\mathrm{Re}(h^T \mathrm{diag}(r) \bar{t})$	$h, r, t \in C^d$	$O(d)$
MTransE [145]	2017	$\|h + r - t\|$	$h, r, t \in R^d$	$O(d)$
Analogy [146]	2017	$h^T M_r t$	$h, t \in R^d$, $M_r \in R^{d \times d}$	$O(d^2)$
TorusE [147]	2018	$\min_{(x,y) \in ([h]+[r]) \times [t]} \|x - y\|_i$	$[h], [r], [t] \in T^d$	$O(d)$
ConvE [148]	2018	$\sigma(\mathrm{vec}(\sigma([M_h; M_r] * \omega)) W) t$	$M_h \in R^{d_\omega \times d_h}$, $M_r \in R^{d_\omega \times d_h}$, $t \in R^d$	–

Table 5 Summary of datasets used for knowledge graph embedding

Datasets	Entities	Relations	Reference/source
FB15K	14,951	1345	[124]/https://everest.hds.utc.fr/doku.php?id=en:transe
WN18	40,493	18	
WN11	38,696	11	[133]/http://cs.stanford.edu/~danqi/data/nips13-dataset.tar.bz2
FB13	75,043	13	
YAGO37	123,189	37	[149]/https://www.mpi-inf.mpg.de/departments/databases-and-information-systems/research/yago-naga/yago/downloads/
DBpedia	–	–	[150]
Freebase	–	–	[151]
Google knowledge graph	–	–	http://googleblog.blogspot.com/2012/05/introducing-knowledge-graph-things-not.html

Table 6 Summary of datasets used for heterogeneous information network embedding

Datasets	Node type						Reference/source
DBLP	Author	Paper	Reference	Term	Venue	Year	[152]/https://aminer.org/citation
	1,003,836	1,756,680	693,406	402,687	7528	62	
IMDB	User	Movie	Actor	Director	Genre		[153]/https://grouplens.org/datasets/movielens/100k/
	943	1360	42,275	918	23		
Blogcatalog	User	Group					http://socialcomputing.asu.edu/datasets/BlogCatalog3
	10,312	39					
Yelp	User	Business	City	Category			https://www.yelp.com/dataset_challenge
	630,639	86,810	10	807			
U.S. Patents	Patent	Inventor	Assignee	Class			[154]/http://www.dev.patentsview.org/workshop/participants.html
	295,145	293,848	31,805	14			
MovieLens	User	Movie	Tag				[155]/https://movielens.org/
	2113	5908	9079				
Drug	User	Drug	Reaction				[156]/http://www.fda.gov/Drugs/
	12	1076	6398				
GPS	User	Location	Activity				[157]
	146	70	5				

The first translation-based model was TransE [124], which was the most representative translational distance model. It represented both entities and relations as vectors in the same space, as shown in Fig. 6a. It has gathered attention because of its effectiveness and simplicity. TransE was inspired by the skip-gram model [23], in which the differences between word embedding often represented their relation. TransE regarded the relation r as translation from entity h to entity t for a golden triplet $\langle h, r, t \rangle$. Hence, $(h + r)$ was close to (t). The score function used for training the vector embedding was defined as

$$f_r(h,t) = \|h + r - t\|_2^2. \tag{1}$$

Note that TransE was only suitable for 1-to-1 relations. There remain flaws for 1-to-N, N-to-1 and N-to-N relations.

To solve these problems of TransE, TransH [134] proposed an improved model named translation on a hyperplane. On hyperplanes of different relations, a given entity has different representations. As shown in Fig. 6b, TransH further projected the embedding h and t to a relation-specific hyperplane by a normal vector w_r, where $h' = h - w_r^T h w_r$ and $t' = t - w_r^T t w_r$. Then, score function used for training was defined as

$$f_r(h,t) = \left\| \left(h - w_r^T h w_r\right) + r - \left(t - w_r^T t w_r\right) \right\|_2^2. \tag{2}$$

Note that both TransE and TransH assumed that entities and relations were in the same vector space. But relations

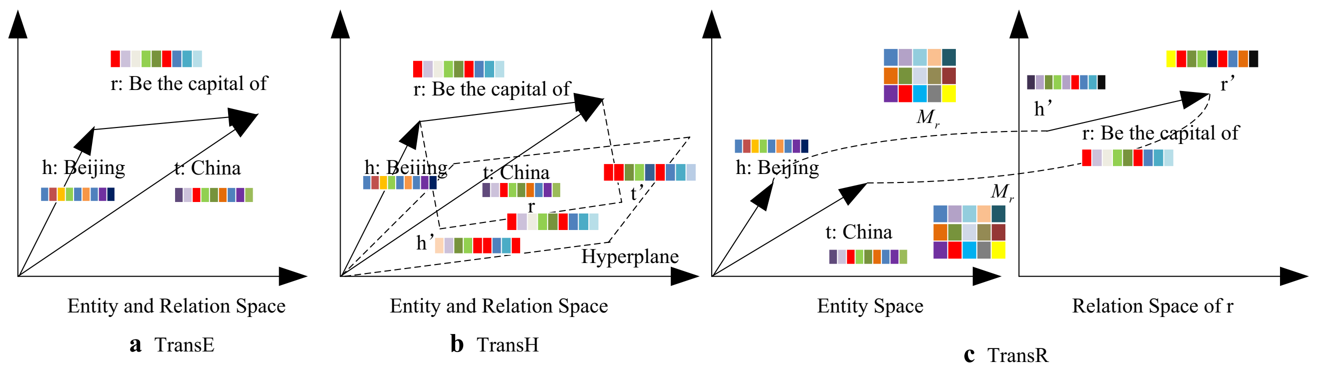

Fig. 6 Simple illustrations of TransE, TransH and TransR

and entities were different types of objects, they should not be in the same vector space.

TransR [135] was proposed based on the above idea. It extended the single vector space used in TransE and TransH to many vector spaces. TransR set a mapping matrix M_r for each relation r to map entity embedding into relation vector space, as shown in Fig. 6c, where $h' = M_r h$ and $t' = M_r t$. The score function was defined as

$$f_r(h,t) = \|M_r h + r - M_r t\|_2^2. \tag{3}$$

However, TransR also has several flaws: (1) For a typical relation r, all entities share the same mapping matrix M_r. In fact, the entities linked by a relation always contain various types and attributes. (2) The projection operation is an interactive process between an entity and a relation; it is unreasonable that the mapping matrices are determined only by relations. (3) Matrix–vector multiplication brings large amount of calculation, and when relation number is large, it also has much more parameters than TransE and TransH.

Thus, to improve the TransR, a more fine-grained model TransD [137] was proposed. Two vectors for each entity and relation were defined. The first vector represented the meaning of an entity or a relation, the other one represented the way that how to project an entity embedding into a relation vector space and it will be used to construct mapping matrices. Therefore, every entity-relation pair had a unique mapping matrix. In addition, TransD had no matrix-by-vector operations which can be replaced by vectors operations. The corresponding scoring function was defined as

$$f_r(h,t) = \|M_{rh} h + r - M_{rt} t\|_2^2, \tag{4}$$

where h_p, r_p and t_p were projection vectors and $I^{m\times n}$ was an identity matrix.

Although the above methods have strong ability to model knowledge graphs, it remains challenging for the reason that the objects (entities and relations) in a knowledge graph are heterogeneous and unbalanced. TranSparse [139] was proposed to overcome these two issues. $M(\theta)$ denoted a matrix M with sparse degree θ. The score function was defined as

$$f_r(h,t) = \|M_r^h(\theta_r^h)h + r - M_r^t(\theta_r^t)t\|_{l1/2}^2. \tag{5}$$

Besides, aiming at the limitations of optimization functions for specific graphs, TransA [138] adaptively found the optimal loss function according to the structure of knowledge graphs, and no closed set of candidates was needed in advance. It not only made the translation-based embedding more tractable in practice, but promoted the performance of embedding related applications. Its scoring function was defined as

$$f_r(h,t) = \|h + r - t\|. \tag{6}$$

Considering most of the knowledge graph embedding methods focused on completing monolingual knowledge graphs, MTransE [145], a translation-based model for multilingual knowledge graph embedding, was proposed. By encoding entities and relations of each language in a separated embedding space, MTransE provided transitions for each embedding vector to its cross-lingual counterparts in other spaces, while preserving the functionalities of monolingual embedding. The score function was the same with the TransA model.

Recently, the researchers noticed the regularization problem in TransE. TorusE [147] aimed to change the embedding space to solve the regularization problem while employing the same principle used in TransE. The required conditions for an embedding space were first considered. Then, a lie group was introduced as candidate embedding spaces. After that, the model embedded entities and relations without any regularization on a torus. Its scoring function was defined as

$$\min_{(x,y)\in([h]+[r])\times[t]} \|x - y\|_i. \tag{7}$$

Discussions and insights Most of the above improved methods are based on TransE. Considering that the model is inspired by the translation model, intuitively, designing algorithms that incorporate the latest research in machine translation will further improve the performance of the task.

4.1.2 Semantic matching models

RESCAL [131] associated each entity with a vector to capture its latent semantics. Each relation was represented by an n-by-n matrix, and the score of triple $\langle h, r, t\rangle$ was calculated by a bilinear map that corresponded to the matrix of the relation r and whose arguments are h and t. The score was defined by a bilinear function

$$f_r(h,t) = h^T M_r t, \tag{8}$$

where $h, t \in R^d$ are vector representations of the entities.$M_r \in R^{d\times d}$ is a matrix associated with the relation.

Considering the limitation of the scoring function of RESCAL, some extensions based on this work have been proposed successively. NTN [133] presented a standard linear neural network structure and a bilinear tensor structure. This can be considered as a generalization of RESCAL. DistMult [136] simplified RESCAL by restricting the matrices representing relations to diagonal matrices. However, it had the problem that the scores of (h, r, t) and (t, r, h) were the same. To solve this problem, ComplEx [144] extended DistMult by using complex numbers instead of real numbers and taking the conjugate of the embedding of the tail entity before calculating the bilinear map. HOLE [141] proposed a holographic embedding method to learn compositional vector space representations of entire knowledge graphs via combining the expressive power of the tensor product used in RESCAL and simplicity of TransE. Analogy [146] extended RESCAL to model the analogical properties, which was particularly desirable for knowledge base completion. For example, if system A (a subset of entities and relations) was analogous to system B (another subset of entities and relations), then the unobserved triples in B could be inferred by mirroring their counterparts in A. It adopted the same score function as model RESCAL and adds constraints to matrix M_r m to model Analogy.

Discussions and insights Bilinear models have more redundancy than translation-based models and so easily become overfitting.

4.1.3 Information fusion-based models

For incorporating extra information to improve the performance of knowledge graph embedding, DKRL [142] proposed description-based representations for entities constructed from entity descriptions with continuous bag-of-words (CBOW) or Convolutional neural network (CNN) was capable of modeling entities in zero-shot scenario. Besides, rich information located in hierarchical entity types was also significant for knowledge graph. TKRL [140] was the first method which explicitly encoded type information into multiple representations in with the help of hierarchical structures.

Discussions and insights Multiple information like textual information and type information, considered as supplements for the structured information embedded in triples, is significant for representation learning in knowledge graph, as already reflected in the model DKRL [142] and TKRL [140]. Intuitively, models that incorporate more information will be closer to real problems and will greatly drive relevant practical applications.

4.1.4 GNN and CNN-based models

Graph neural networks (GNNs), including GCN and GAT, have been proved to have strong ability to model graph data. Naturally, knowledge graph is also their application scenario. Representative methods mainly include R-GCNs [158] and the recent work [159] based on the thought of GAT. Specifically, R-GCNs are an extension of applying GCN to knowledge graph. The model applies a convolution operation to the neighborhood of each entity and assigns them equal weights. Nathani et al. [159] proposed a graph attention-based feature embedding that captures both entity and relation features in a multi-hop neighborhood of a given entity. In addition to GNN, researchers also tried to solve the problem of knowledge graph embedding with the ability of multilayer convolutional neural networks to learn deep expressive features. Typical work is ConvE [148]. The model uses 2D convolution over embeddings and multiple layers of nonlinear features to model knowledge graphs.

Discussions and insights With the rapid development of deep learning, especially graph neural networks and convolutional neural networks in recent years, the application of these advanced methods in the field of knowledge graphs has become increasingly popular.

4.1.5 Extra models

In addition to the above methods, ManifoldE [143] applied the manifold-based principle $M(h, r, t) = D_r^2$ for a specific triple $\langle h, t, t\rangle$. When a head entity and a relation were given, the tail entities laid in a high-dimensional manifold. Intuitively, the score function was designed by measuring the distance of the triple away from the manifold

$$f_r(h,t) = \left\| \|h + r - t\|_2^2 - D_r^2 \right\|^2, \tag{9}$$

where D_r was a relation-specific manifold parameter.

4.2 Heterogeneous information network embedding

4.2.1 Optimization-based methods

The representative methods of this kind of methods are LSHM [130], PTE [160] and Hebe [161, 162]. The first method was latent space heterogeneous model, which is one of the initial works. The model assumed that two nodes connected in the network will tend to share similar representations regardless of their types. The node latent representation could be learnt by the function which combined the classification and regularization losses. The labels of different type node were then deduced from these representations. The second method was PTE model that utilizes both labeled and unlabeled data to learn the embedding of text for heterogeneous text network. The labeled information and different levels of word co-occurrence information were first represented as a large-scale heterogeneous text network. Then, the heterogeneous text network can be composed of three bipartite networks: word–word, word-document and word-label networks. To learn the embedding of the heterogeneous text network, an intuitive approach was to collectively embed the three bipartite networks, which can be achieved by minimizing conditional probabilities between the node pair in three networks. The third method was Hebe, which captured multiple interactions in the heterogeneous information networks as a whole. The hyperedge was defined as a set of objects forming a consistent and complete semantic unit, which was taken as event. Therefore, this method preserved more semantic information in the network. Two methods were presented based on the concept of hyperedge: HEBE-PO, modeling the proximity among the participating objects themselves on the same hyperedge, and HEBE-PE modeling proximity between the hyperedge and the participating objects.

Discussions and insights The above model has limited modeling of heterogeneous networks, resulting in the loss of a lot of information, such as node type information. Due to the complexity of heterogeneous networks, designing more accurate objective functions is the key to this type of methods.

4.2.2 Deep learning-based methods

Analogous to homogeneous network embedding, deep learning models are also applied to heterogeneous network representation learning, which could capture the complex interactions between heterogeneous components. These methods are mainly divided into two branches: conventional neural network-based methods and GNN-based methods.

4.2.2.1 Conventional neural network-based methods Conventional neural networks include CNN, autoencoder, etc., which have achieved great success in image processing, computer vision and other fields. However, considering that heterogeneous networks are often rich in multiple types of information and complex structures, it is usually not a trivial problem for traditional neural networks to work well on such networks. In this type of approach, HNE [4] and DHNE [156] are the two representative methods.

The first method is HNE, which jointly modeled contents and topological structures in heterogeneous networks via deep architectures. Briefly, the model decomposed the feature learning process into multiple nonlinear layers of deep structure. Three modules of image–image, image–text and text–text were included in the model. The model used CNN network to learn the features of the image and adopted the fully connected layer to extract the discriminative text representation. These were connected to the prediction layer. Finally, it transferred different objects in heterogeneous networks to unified vector representations. Although the above method has achieved good results, it ignores the ubiquitous problem of indecomposable hyperedge [163, 164] in heterogeneous networks. Therefore, DHNE proposed to embed hypernetworks with indecomposable hyperedges by deep model. The model theoretically proved that any linear similarity metric in embedding space commonly used in existing methods cannot maintain the indecomposibility property in hypernetworks. A new deep model with autoencoder was designed to realize a nonlinear tuplewise similarity function while preserving both local and global proximities in the formed embedding space.

4.2.2.2 Graph neural network-based methods Influenced by the successful implementation of graph neural network in processing homogeneous network, recently heterogeneous graph neural network has been developed gradually. The representative methods are HetGNN [165] and HAN [166].

HetGNN was a recent heterogeneous graph neural network model that jointly considered heterogeneous structural (graph) information and heterogeneous contents information of each node effectively. Firstly, the model introduced a random walk with restart strategy to sample a fixed size of strongly correlated heterogeneous neighbors for each node and group them based upon node types. Secondly, the model designed a neural network architecture with two modules to aggregate feature information of those sampled neighboring nodes. Finally, the model leveraged a graph context loss and a mini-batch gradient descent procedure to train the model in an end-to-end manner. Note that the above model does not consider the importance information in the heterogeneous network. Thus, HAN proposed a heterogeneous graph neural network based on the hierarchical attention, including node-level and semantic-level attentions. Specifically, the node-level attention aimed to learn the importance between a node and its meta-path based neighbors, while the semantic-level attention could learn the importance of different meta-paths.

Discussions and insights It can be seen that there are fewer heterogeneous network embedding models based on deep learning model comparing with homogeneous network embedding of the same type of methods, because it is a challenging task. At the same time, as mentioned above, deep learning technology has been booming in the field of homogeneous network embedding. Therefore, designing heterogeneous network embedding algorithms based on deep learning has greater development potential. In addition to deep learning, we also notice that broad learning [167] has been introduced in the field of network embedding, such as Zhang et al. [168]. Considering that broad learning is an emerging method, thus, network embedding based on broad learning technology will have great potential for development.

4.2.3 Meta-path-based methods

Meta-path has been widely used in heterogeneous graph analysis [169, 170]. Inspired by this, two representative works: Metapath2vec and metapath2vec++ [171] and HIN2Vec [154] are proposed. The former preserved both structural and semantic correlations of heterogeneous graphs. The model designed a random walk based on meta-path. It was used to generate heterogeneous neighborhoods with network semantics for various types of node. The heterogeneous skip-gram model to perform node embedding was constructed. The latter also explored meta-paths in heterogeneous networks for representation learning. Different from the former, HIN2Vec also learnt representations of meta-paths. Specifically, the model firstly adopted random walk generation and negative sampling to prepare training data. Then, the model trains a logistic binary classifier that predicts whether two input nodes has a specific relationship in order to efficiently learn model parameters, i.e., node vectors and meta-path vectors.

Discussions and insights Although meta-path based methods have achieved good performance, it is not enough to sample rich semantic information contained in heterogeneous information networks only by meta-path. Therefore, designing a better semantic information sampling strategy is greatly important to improve the performance of heterogeneous information network embedding algorithms. The latest model such as MetaGraph2vec [172] and HERec [173] has just explored how to learn better node embedding vectors by fusing semantic information from different meta-paths, and successfully applied it to node classification and recommendation tasks.

4.2.4 Extra models

There are some extra models, which have made a relatively new exploration, such as ASPEM [153] and Tree2Vector [174]. Specifically, ASPEM encapsulated information regarding each aspect individually, instead of preserving information of the network in one semantic space. The aspect selection method was proposed, which demonstrated that a set of representative aspects can be selected from any heterogeneous information networks using statistics of the network without additional supervision. To design the embedding algorithm for one aspect, the skip-gram model [23] was extended. The final embedding for node u was thereby given by the concatenation of the learned embedding vectors from all aspects involving u. For tree-structured data, we can take it as a specific form of heterogeneous network. For example, a tree structure for representing the content of an author can be constructed by author biography node, written books nodes and book comments nodes [174, 175]. In such cases, researchers have designed effective methods to transform tree-structured data into vectorial representations. The representative work is Tree2Vector [174], which is mainly implemented through the node allocation process via employing the unsupervised clustering technique, and the locality-sensitive reconstruction method to model the reconstruction process. However, this is an unsupervised learning model, which cannot be applied to supervised learning directly. Therefore, designing an end-to-end supervised learning model for semantic tree-structured data is another promising issue.

Summary In order to show the development process of so many representation algorithms more clearly, we illustrated the development timeline that network representation learning experienced in Fig. 7.

5 Application

Because of the universality of the embedding vectors, network embedding technology can be applied to many fields and tasks by using the off-the-shelf machine learning method. Especially, with the rapid development of GNN,

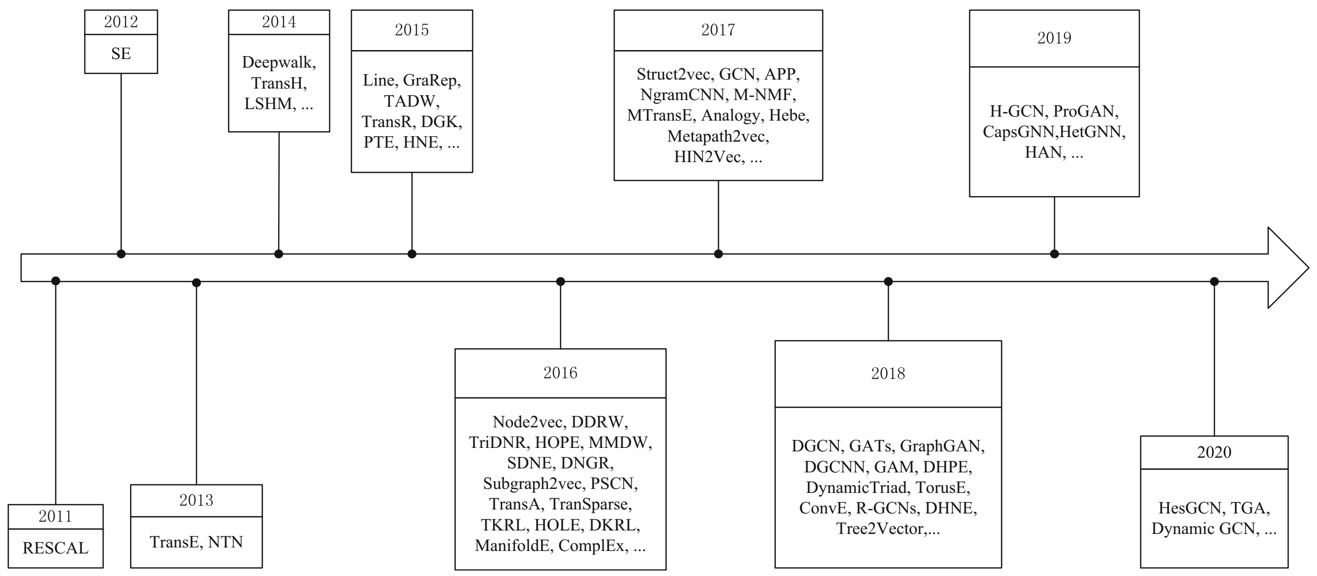
2012
SE
2014
Deepwalk, TransH, LSHM, ...
2015
Line, GraRep, TADW, TransR, DGK, PTE, HNE, ...
2017
Struct2vec, GCN, APP, NgramCNN, M-NMF, MTransE, Analogy, Hebe, Metapath2vec, HIN2Vec, ...
2019
H-GCN, ProGAN, CapsGNN,HetGNN, HAN, ...
2011
RESCAL
2013
TransE, NTN
2016
Node2vec, DDRW, TriDNR, HOPE, MMDW, SDNE, DNGR, Subgraph2vec, PSCN, TransA, TranSparse, TKRL, HOLE, DKRL, ManifoldE, ComplEx, ...
2018
DGCN, GATs, GraphGAN, DGCNN, GAM, DHPE, DynamicTriad, TorusE, ConvE, R-GCNs, DHNE, Tree2Vector,...
2020
HesGCN, TGA, Dynamic GCN, ...

Fig. 7 The timeline of network representation learning

the application of GNN in many fields is in the ascendant. In this review, we introduce the classic and the latest applications in a more comprehensive way, i.e., node classification (Sect. 5.1), node clustering (Sect. 5.2), link prediction (Sect. 5.3), visualization (Sect. 5.4), recommendation (Sect. 5.5), graph classification (Sect. 5.6), application in knowledge graph (Sect. 5.7) and extra application scenarios (Sect. 5.8).

5.1 Node classification

Node classification is a very important issue in domain of graph mining and has a great theoretical significance and application value [176, 177]. Node classification aims to predict unlabeled nodes by training the samples of labeled nodes, which is the most widely used task of node embedding. After obtaining the vector representation of nodes, many off-the-shelf machine learning methods and packages (such as SVM [35, 37–39], logistic regression [27, 32–34, 62, 65, 178] and Liblinear package [26, 63, 179]) could be leveraged to implement the task. First, the feature vectors of the labeled nodes are used to train classification model. Then, the vectors of the unlabeled nodes are utilized as the input of the classification model, and their label categories are deduced. At the same time, there are few deep learning models using the softmax layer for classification, such as GATs [64].

5.2 Node clustering

One of the most important tasks in graph mining and network analysis is node clustering, whose target is to infer the clusters in graph based on the network structure [58]. Many practical applications can be transformed into node clustering problems, such as community detection in social networks [39]. After we learn the node features through the network embedding algorithm, we can use the off-the-shelf clustering algorithm such as K-means to achieve the node clustering task. Many works such as [34, 66, 68] have followed the above steps to verify the performance of the algorithm. In addition, in recent years, deep graph clustering method such as [99] has developed rapidly, aiming to obtain clustering-oriented node embedding in framework of deep learning.

5.3 Link prediction

Link prediction refers to the task of predicting either missing interactions or links that may appear in the future or in an evolving network. e.g., in social networks, friendship links can be missing between two users who actually know each other [180]. The basic assumption of graph embedding is that a good network representation should be able to capture and preserve rich structure and semantic information of graph, which can predict unobserved links in the graph. Therefore, the graph embedding method can further infer the graph structure. Recent work has successfully applied this approach to link prediction tasks in homogenous and heterogeneous graph [5, 25–29, 65, 120, 181–183]. For example, Node2vec [27] predicts the missing edges between two users in Facebook network, protein–protein interactions network and collaboration network. For detecting the new protein–protein interactions, Cannistraci et al. [181] offered a solution for topological link prediction by network embedding. Zitnik et al. [5] modeled polypharmacy side effects in multimodal heterogeneous networks with graph convolutional networks [60].

5.4 Visualization

Data visualization is a powerful technology for data exploration [184]. Once we obtain low-dimensional vector representation of nodes, we can use the commonly data visualization techniques (such as t-distributed stochastic neighbor embedding (t-SNE) [185] and LargeVis [186]) to visualize graph data in 2D space. Good vector representation often refers to the common thing that the nodes of the same category are close to each other in 2D space with different color. Some recent work has applied to this task [26, 33, 34, 36, 38, 43, 63], which can be used to visually reflect the data representation.

5.5 Recommendation

Recommender systems have been playing an increasingly important role in various online services [187]. The conventional recommendation methods can be mainly divided into content-based recommendation [188] and collaborative filtering based recommendation [189]. Additionally, it is worth noting that network embedding has been successfully applied to the recommendation in a new light, such as [25, 29, 65, 173]. For example, for incorporating various kinds of auxiliary data in the view of HIN, Shi et al. [173] modeled and utilized these heterogeneous and complex information in recommender systems.

5.6 Graph classification

Graphs are powerful structures that can be used to model almost any kind of data, such as social network [190]. An instinctive application of whole graph embedding is graph classification. Graph classification assigns a class label to a whole graph. It has a wide range of applications in bioinformatics, chemistry, drug design and other fields, such as [61, 110, 111, 118, 191–196].

5.7 Application in knowledge graph

The applications based on knowledge graph embedding technology have been widely deployed in knowledge graph [197]. The three main applications are link prediction, triplet classification and knowledge graph completion. Link prediction is to predict the missing h or t for a golden triplet (h, r, t). There are some knowledge graphs embedding research work involved this task, such as [124, 134–139, 141, 143, 147]. Triplet classification aims to judge whether a given triplet (h, r, t) is correct or not, which is a binary classification task. It has been successfully applied in these works including [133–135, 137–140, 143]. Knowledge graph completion aims to complete a triple (h, r, t) when one of h, r, t is missing. The representative work is [140, 142].

5.8 Extra application scenarios

Network embedding technology is likewise widely used in other tasks or fields because of the ubiquity of graphs. E.g., for the question of NP-hard combinatorial optimization problems over graphs, Dai et al. [198] proposed a unique combination of reinforcement learning [199] and network embedding to address this challenge, which can automatically learn the optimization algorithm. In the field of multimodal data analysis, the heterogeneous networks can be constructed from images and text. The obtained vector via heterogeneous network embedding technology can be applied to cross-modal retrieval (see [4]). In domain of image processing and text analysis, Defferrard et al. [115] reported the application for MNIST image classification and text categorization. In the biochemistry field, there is an emerging field of graph convolutional networks and their applications in drug discovery and molecular informatics (see [200]). In the field of intelligent transportation, researchers have used GCN in traffic prediction and achieved excellent performance (see [201]). In short, given its powerful capabilities of processing omnipresent graph data, network embedding technology will have broader application prospects.

6 Experiments

For the sake of making a more comprehensive analysis of the algorithm, we conduct some experiments on some benchmark datasets. Owing to the large literature, we could not compare to every method but to test some representative network embedding algorithms with the open-source code and open platform such as PyTorch Geometric,[1] OpenNe[2] and OpenKe.[3] We try our best to cover the newer methods and show the trend of the problem. Our experiment is tested on the Ubuntu 16.04 LTS 64bit, RAM = 125 GB and CPU = Intel® Xeon(R) CPU E5-2650 v3 @ 2.30 GHz $\times$ 40 Cores. For homogeneous network embedding, we select the representative node embedding algorithms for performing the task of node classification and visualization which is the most direct and common task, and the representative whole network embedding algorithms for conducting the graph classification task. For

[1] https://github.com/rusty1s/pytorch_geometric.

[2] https://github.com/thunlp/OpenNE.

[3] https://github.com/thunlp/OpenKE.

the heterogeneous network embedding, we choose a widely developed knowledge graph embedding technology for testing the link prediction task and triple classification task, and some meta-path based methods with continuity in heterogeneous information network embedding for comparing the node classification performance. The specific experiments are as follows.

6.1 Homogeneous node embedding

For homogeneous node embedding, we employ three benchmark datasets from the open-source datasets in Table 2 and OpenNe, whose statistics are summarized in Table 7.

Blogcatalog: It is a network of social relationships of the bloggers listed on the BlogCatalog website. The labels of vertices represent blogger interests inferred through the metadata provided by the bloggers. This graph has 5196 vertices, 171,743 edges and 6 different labels.

Wiki: It contains 2405 documents from 17 classes and 17,981 links between them.

Cora: This is a citation network. The links are citation relationships between the documents. This graph contains 2708 machine learning papers from 7 classes and 5429 links between them. Each document is described by a binary vector of 1433 dimensions indicating the presence of the corresponding word.

6.1.1 Setting and metric

The latent dimension to learn for each node is 128 with default. For the model used for supervised node classification, the ratio of training data for node classification is 0.5. The training epoch of LINE is set to 25. The parameters p and q for DeepWalk and Node2vec are set to 1, 1 and 1, 0.25. The logistic regression classifier is adopted. For the model used for semi-supervised node classification, we train all models for a maximum of 200 epochs (training iterations) using Adam [202]. We set the training rate for ChebConv [115], GCN [60] to 0.01 and GAT [64] to 0.005. For comparing the performance of different algorithms, the Micro-F1, Macro-F1 and accuracy are adopted which are widely used metrics for node classification [27, 32, 35].

6.1.2 Result analysis

Table 8 compares the performance of different algorithms on node classification. For each dataset, the best performing method across all baselines is bold-faced. For supervised node classification on BlogCatalog datasets, GraRep model shows excellent results. This may be because our fixed parameters of this method are more suitable for this datasets. For Wiki datasets, the method of Node2vec achieves the best classification performance of Micro-F1 and Macro-F1 overall. This can be attributed to the advantages of BFS and DFS strategies.

For semi-supervised node classification on Cora datasets, we perform the task by reporting average accuracies of 10 runs for the fixed train/valid/test split from GCN model setting [60]. As shown in Table 8, the GAT model shows the best performance. This is due to the fact that the model fully considers the importance of different neighbor nodes through the attention mechanism.

In order to make the comparison of the results more obvious, we use the following representative four methods for the visualization task, i.e., Gf [203], Line [33], GraRep [34] and Node2vec [27]. We visualize Wiki network using t-SNE (the dimension of embedding is 128) (see Figs. 8, 9, 10, 11) to illustrate the properties preserved by embedding methods. Each point corresponds to a node in the network. Color denotes the node label.

As shown in the results, the node layout using Gf in Fig. 8 is not very informative, since vertices of different colors are almost mixed with each other. For Line and Node2vec, results look much better, as most vertices of the same color appear to form groups. However, vertices still

Table 7 Statistics of the datasets

Datasets	Blogcatalog	Wiki	Cora
Nodes	5196	2405	2708
Edges	171,743	17,981	5429
Feature	–	–	1433
Labels	6	17	7

Table 8 Results of the node classification

Metric	Method	Blogcatalog	Wiki	Cora
Supervised node classification				
Micro-F1	DeepWalk [32]	0.6586	0.6597	–
	Line (lst + 2nd) [33]	0.6967	0.6417	–
	GraRep [34]	**0.7290**	0.6517	–
	Node2vec [27]	0.6610	**0.6737**	–
Macro-F1	DeepWalk [32]	0.6520	0.5184	–
	Line (lst + 2nd) [33]	0.6921	0.5455	–
	GraRep [34]	**0.7248**	0.5077	–
	Node2vec [27]	0.6567	**0.5486**	–
Semi-supervised node classification				
Accuracy	ChebConv [115]	–	–	0.7935
	GCN [60]	–	–	0.8112
	GAT [64]	–	–	**0.8219**

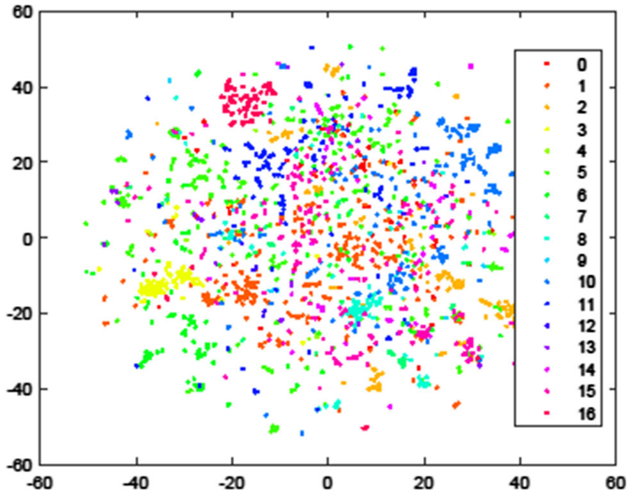

Fig. 8 Gf method

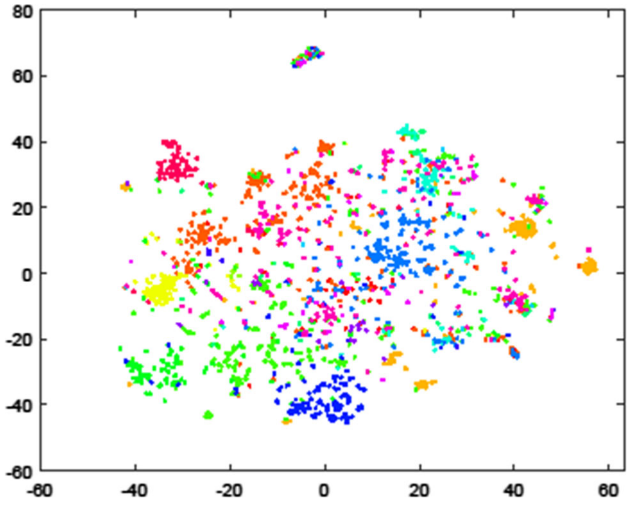

Fig. 10 Node2vec method

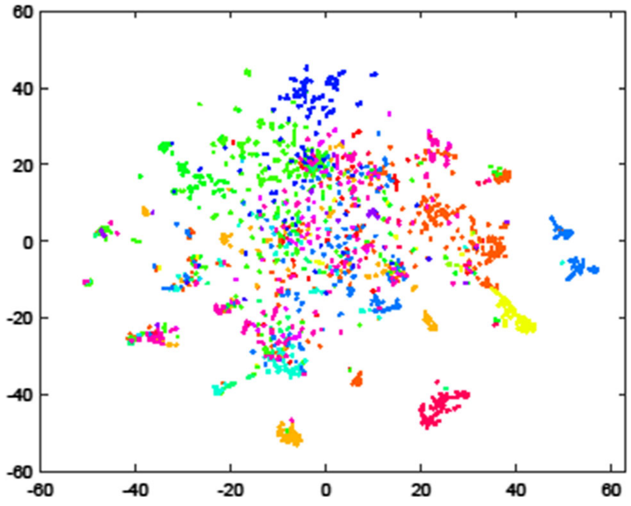

Fig. 9 Line method

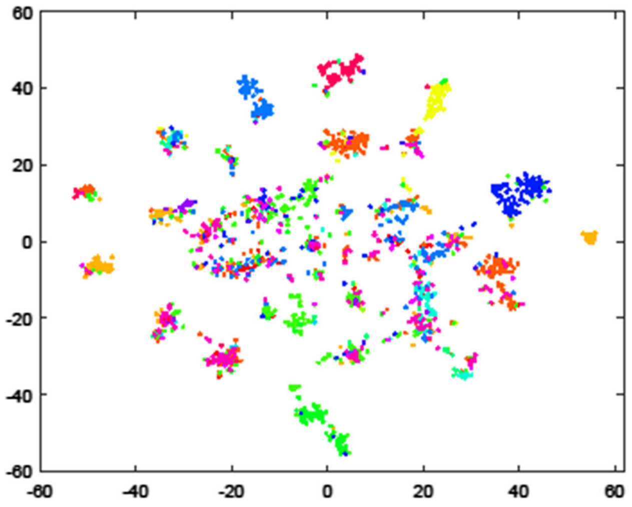

Fig. 11 GraRep method

do not appear in clearly separable regions with clear boundaries. For GraRep, the boundaries of each group become much clearer, with vertices of different colors appearing in clearly distinguishable regions.

6.2 Homogeneous whole network embedding

For homogeneous whole network embedding, we employ the three benchmark datasets from the open-source datasets in Table 3, whose statistics are summarized in Table 9. MUTAG is a dataset of 188 mutagenic aromatic and heteroaromatic nitro compounds with 7 discrete labels. PROTEINS is a dataset obtained from where nodes are secondary structure elements (SSEs) and there is an edge between two nodes if they are neighbors in the amino-acid sequence or in 3D space. It has 3 discrete labels, representing helix, sheet or turn. D&D is a dataset of 1178 protein structures. Each protein is represented by a graph, in which the nodes are amino acids and two nodes are connected by an edge if they are less than 6 Angstroms apart.

6.2.1 Setting and metric

Following the same experimental settings such as DGCNN [113], we conduct tenfold cross-validation (ninefolds for training, onefold for testing) and report the average classification accuracy. For the graph kernel parameter setting, the height parameters of WL [105] and PK [204] are chosen from the set {0, 1, 2, 3, 4, 5}. The size of the

Table 9 Statistics of the datasets

Datasets	Graphs	Classes	Avg. nodes	Avg. edges
MUTAG	188	2	17.93	19.79
PROTEINS	1113	2	39.06	72.81
D&D	1178	2	284.31	715.65

graphlets for GK was set to 3. For the random walk (RW) [106] kernel, we set the decay parameter as λ, following the suggestion in [105]. For parameter setting of the deep learning-based method, we follow the same setting to DGCNN [113]. By default, we reproduce the results reported in their original work for comparison if the experiment code is public and experimental conditions permit. However, in cases where the results are not available, we use the best testing results under the same experimental setting reported in original work.

6.2.2 Result analysis

As shown in Table 10, the deep learning-based approaches for graph classification have shown considerable advantages. Specifically, compared with all the other algorithms, DGCNN achieves highly competitive results on PROTEINS and D&D datasets with the compared graph kernels and some state-of-the-art deep learning methods. This is due to the advantages of end-to-end learning. CapsGNN achieves top 2 on 2 out of 3 datasets and achieves comparable results on the other datasets. This may be because the algorithm is suitable for capturing structures at the graph level and not for retaining finer structures.

6.3 Heterogeneous knowledge graph embedding

For heterogeneous knowledge graph embedding, we use the two benchmark datasets from the open-source datasets of Table 5. The datasets statistics are summarized in Table 11. FB15K is a subset of Freebase, a curated knowledge base of general facts, whereas WN18 is a subset of Wordnet, a database featuring lexical relations between words.

6.3.1 Setting and metric

The latent dimensions of the entities and relations, and training times are set to 100 with default uniformity. For the TransE [124] model, the optimizer is set to Adam [202]. For the RESCAL [131], DistMult [136], ComplEx [144] and Analogy [146] algorithm, the optimizer is set to Adagrad [206]. Following the literature of knowledge graph embedding [146, 147, 207], we adopt the accuracy for evaluating the task of triple classification. For the task of link prediction, we use the conventional metrics of Hits@10 and mean reciprocal rank (MRR) which evaluate the proportion of correct entities ranked in top 10 and average the scores over all ranked lists for the entire test set of instances. Similarly, the result is reported according to the two setting of “raw” and “filt” (see [124]). A higher MRR and a higher Hits@10 should be achieved by a good embedding model.

6.3.2 Result analysis

Tables 12 and 13 compare the results of triple classification and link prediction of baseline methods. Analogy [146] shows the best or the 2nd best accuracy performance in two tasks. It indicates that the original performance of the paper [146] has been reproduced to a certain extent. It may be benefited from considering the analogical properties. Especially, on FB15K dataset in Table 12, Analogy model outperforms all baseline methods. The performance of this method is better than the baseline RESCAL algorithm 11.2%. On WN18 datasets, DistMult shows the performance that is close to Analogy method in the task of triple classification. In Table 13, Analogy model has the highest value of MRR and Hit@10 on FB15K datasets. On WN18 datasets, the ComplEx algorithm has the comparable performance with Analogy in the task of link prediction.

6.4 Heterogeneous information network embedding

For heterogeneous network embedding except for knowledge graph embedding, we also choose two commonly datasets from the website (https://aminer.org/citation) in Table 6. For DBLP datasets, we extract a subset of DBLP bibliographic heterogeneous information network (DBLP-Citation-network V3) which is composed of authors (A) nodes, papers (P) nodes, venues (V) nodes and their relationships. Based on papers venues, we extract papers falling into two research areas: data mining, artificial intelligence. Specifically speaking, we choose two venues, SDM and COLT. Then, the relevant papers and authors establish corresponding relationships according to the venues. For ACM datasets, unlike the DBLP datasets, we extract a subset of heterogeneous information network (ACM-Citation-network V9) which also contains authors (A) nodes, papers (P) nodes, venues (V) nodes and their relationships. Another difference is that the venues we choose are KDD and ICML. The datasets statistics are summarized in Table 14.

Table 10 Results of the graph classification

Metric	Method	MUTAG	PROTEINS	D&D
Some classical graph kernels methods for graph classification				
Accuracy	GK [107]	81.39	71.32	74.38
	RW [106]	79.17	72.75	> 3 days
	WL [105]	83.22	74.54	78.34
	PK [204]	74.11	73.41	77.79
Some state-of-the-art deep learning methods for graph classification				
Accuracy	DGK [76]	87.44	71.68	–
	PSCN [110]	88.95	75.00	76.27
	ECC [205]	–	–	72.54
	DGCNN [113]	85.83	**75.54**	**79.37**
	CapsGNN [72]	**90.44**	75.20	74.89

Table 11 The statistics of the datasets

Datasets	Entities	Relations	Train	Valid	Test
FB15K	14,951	1345	483,142	50,000	59,071
WN18	40,493	18	141,442	5000	5000

Table 12 Results of the triple classification

Metric	Method	FB15K	WN18
Accuracy	RESCAL [131]	0.774	0.932
	TransE [124]	0.820	0.891
	DistMult [136]	0.885	**0.969**
	ComplEx [144]	0.891	0.967
	Analogy [146]	**0.893**	**0.968**

6.4.1 Setting and metric

The dimension of node representations is uniformly set to 128. The other parameter setting of all the method is set as original default in its open code. We set the meta-path of Metapath2Vec [171] as APAPA. For HERec [173] model which could be taken as a variant of Metapath2Vec, in this review, the final embedding is obtained by the simple linear fusion of the two meta-paths of APAPA and APVPA. For the MetaGraph2vec [172] model, Metagraph can be considered as a union of meta-paths (APAPA and APVPA), whose setting is the same to the original paper. After obtaining the node representation, we train the binary support vector machine (SVM) classifier using the fitcsvm tools in MATLAB 2018a. We repeat each method 10 times, and report the average performance in terms of both Macro-F1 and Micro-F1.

Table 14 The statistics of the datasets

Datasets	No. of A nodes	No. of P nodes	No. of V nodes
DBLP	1925	1511	2
ACM	1942	1175	2

6.4.2 Result analysis

Based on Table 15, we can observe that the MetaGraph2vec model achieves the best performance. The results demonstrate that it is quite important to fuse the rich semantics of meta-paths in heterogeneous network analysis.

7 Conclusion and future directions

In this review, we survey the recent advance of network representation learning comprehensively and systematically. Firstly, we classify network embedding in the framework of homogeneous and heterogeneous network. Then, the corresponding algorithm is deeply analyzed, respectively. For homogeneous network embedding, the embedding methods with different features are summarized, respectively. Specifically, node embedding, subgraph embedding, whole network embedding and dynamic network embedding are included. For heterogeneous network

Table 13 Results of the link prediction

Datasets	FB15K				WN18			
Method	MRR		Hit@10		MRR		Hit@10	
	Raw	Filter	Raw	Filter	Raw	Filter	Raw	Filter
RESCAL [131]	0.137	0.229	0.276	0.401	0.338	0.399	0.518	0.570
TransE [124]	0.193	0.315	0.420	0.575	0.224	0.266	0.569	0.638
DistMult [136]	0.206	0.379	0.436	0.636	0.506	0.757	0.792	0.937
ComplEx [144]	0.219	0.469	0.467	0.723	**0.574**	**0.931**	**0.798**	**0.939**
Analogy [146]	**0.224**	**0.475**	**0.470**	**0.725**	**0.574**	0.927	0.797	**0.939**

Table 15 Results of the author classification

Methods	Metapath2Vec [171]	HERec [173]	MetaGraph2vec [172]
Author classification performance comparison on DBLP datasets			
1%			
Macro-F1	0.4662	0.8412	**0.8674**
Micro-F1	0.6457	0.8750	**0.8983**
3%			
Macro-F1	0.6340	0.9351	**0.9579**
Micro-F1	0.7231	0.9403	**0.9612**
5%			
Macro-F1	0.6761	0.9375	**0.9593**
Micro-F1	0.7454	0.9431	**0.9624**
7%			
Macro-F1	0.7284	0.9398	**0.9598**
Micro-F1	0.7745	0.9454	**0.9630**
9%			
Macro-F1	0.7422	0.9437	**0.9623**
Micro-F1	0.7777	0.9488	**0.9653**
Author classification performance comparison on ACM datasets			
1%			
Macro-F1	0.4748	0.4775	**0.5031**
Micro-F1	0.8984	0.8992	**0.9019**
3%			
Macro-F1	0.4775	0.5460	**0.5866**
Micro-F1	0.8989	0.9082	**0.9141**
5%			
Macro-F1	0.4786	0.7345	**0.7808**
Micro-F1	0.8992	0.9348	**0.9438**
7%			
Macro-F1	0.4792	0.8033	**0.8067**
Micro-F1	0.8993	0.9465	**0.9481**
9%			
Macro-F1	0.4812	0.8447	**0.8781**
Micro-F1	0.8994	0.9550	**0.9623**

embedding, considering the vigorous development of knowledge graph embedding technology, the embedding method is classified into two categories from knowledge graph embedding and heterogeneous information network embedding. We also summarize various applications related to network embedding and their respective evaluation metrics. Finally, we empirically evaluate the surveyed methods on some typical applications using several publicly available real networks and compare their performance.

Although representation learning for network data has gained great achievements as described above, it still faces great challenges in the following areas in the future work. The future direction will be introduced from two aspects: theory and application.

7.1 Algorithmic and theoretical aspect

1. Almost all network embedding algorithms currently assume that network data are embedded into the Euclidean space. However, some network characteristics (such as power-law distributions, strong clustering and hierarchical community structure,) could not emerge naturally [208, 209] in this space. Recent work shows that exploring network embedding method in non-Euclidean space could solve the above issues, e.g., hyperbolic space [208, 209]. Therefore, designing the network embedding algorithm in non-Euclidean space is still a challenging and promising research direction.
2. At present, dynamics network embedding for homogeneous network is just beginning to be concerned by the researchers (see Sect. 3.4). Intuitively, the network

in the real world is mostly dynamic and heterogeneous without doubt. Learning rich information from this type of network is also a very big challenging problem. Consequently, designing efficient embedding algorithm for dynamic, temporal and heterogeneous networks will be the next important research topic.

3. The existing deep learning-based network representation learning methods still have some problems, such as instability, complicated training and time-consuming for big graph datasets. In particular, recent studies have found that the robustness of such methods is poor [210, 211], just like the adversarial samples problem in the field of image recognition based on deep learning [212]. Consequently, there is still great potential for these challenges.
4. How to directly measure the quality of data representation is a long-standing problem that has been troubling the field of representation learning. The current measurement of the quality of data representation is measured by indirectly executing the relevant tasks. Network representation learning as a new subdomain of representation learning is naturally no exception. Hence, proposing or designing a new evaluation or metric method for representation learning is an urgent issue. On the other hand, the current graph embedding algorithm is less interpretable. This problem greatly limits its large-scale practical application. Intuitively, the interpretability of models and the metrics of data representation may promote each other's development.
5. Designing efficient graph neural networks to support combinatorial generalization and relational reasoning is a new and promising direction. The latest research [213] from *DeepMind, Google Brain* and other institutions shows that graph neural networks can be used to achieve relational reasoning, which may solve the core problem that Turing award winner *Judea Pearl* points out that current deep learning cannot complete causal reasoning task [214].

7.2 Application oriented aspect

Considering the ubiquity of network data, many interdisciplinary problems can be abstracted as graph mining and analysis problems (see Sect. 5). Although there have been some successful applications such as protein–protein interactions network [181, 183], brain network [215, 216], molecular graph generation [217], drug discovery and synthetic biology [102], it still has a very broad prospect in the application of network embedding algorithm.

Acknowledgements This research is supported by the National Natural Science Foundation of China (Grant Number: U1433116), and the Fundamental Research Funds for the Central Universities (Grant Number: NP2017208). Thanks are due to all developers and researchers providing open-source platform. Thanks are due to all the anonymous reviewers.

Compliance with ethical standards

Conflict of interest We declare that we have no conflict of interest.

References

1. Lynch C (2008) Big data: how do your data grow? Nature 455(7209):28–29. https://doi.org/10.1038/455028a
2. Lazer D, Kennedy R, King G, Vespignani A (2014) The parable of Google Flu: traps in big data analysis. Science 343(6176):1203–1205. https://doi.org/10.1126/science.1248506
3. Howe D, Costanzo M, Fey P, Gojobori T, Hannick L, Hide W, Hill DP, Kania R, Schaeffer M, Pierre SS, Twigger S, White O, Rhee SY (2008) Big data: the future of biocuration. Nature 455(7209):47–50. https://doi.org/10.1038/455047a
4. Chang S, Han W, Tang J, Qi GJ, Aggarwal CC, Huang TS (2015) Heterogeneous network embedding via deep architectures. In: ACM SIGKDD international conference on knowledge discovery and data mining, pp 119–128. https://doi.org/10.1145/2783258.2783296
5. Zitnik M, Agrawal M, Leskovec J (2018) Modeling polypharmacy side effects with graph convolutional networks. Bioinformatics 34:i457–i466. https://doi.org/10.1093/bioinformatics/bty294
6. Bengio Y, Courville A, Vincent P (2013) Representation learning: a review and new perspectives. IEEE Trans Pattern Anal Mach Intell 35(8):1798–1828. https://doi.org/10.1109/TPAMI.2013.50
7. LeCun Y, Bengio Y, Hinton G (2015) Deep learning. Nature 521(7553):436–444. https://doi.org/10.1038/nature14539
8. Dahl GE, Ranzato MA, Mohamed AR, Hinton G (2010) Phone recognition with the mean-covariance restricted Boltzmann Machine. In: Proceedings of the 23rd international conference on neural information processing systems, pp 469–477
9. Mohamed A, Dahl GE, Hinton G (2012) Acoustic modeling using deep belief networks. IEEE Trans Audio Speech Lang Proc 20(1):14–22. https://doi.org/10.1109/tasl.2011.2109382
10. Hinton G, Deng L, Yu D, Dahl GE, Mohamed A, Jaitly N, Senior A, Vanhoucke V, Nguyen P, Sainath TN (2012) Deep neural networks for acoustic modeling in speech recognition: the shared views of four research groups. IEEE Signal Process Mag 29(6):82–97. https://doi.org/10.1109/MSP.2012.2205597
11. Farabet C, Couprie C, Najman L, LeCun Y (2013) Learning hierarchical features for scene labeling. IEEE Trans Pattern Anal Mach Intell 35(8):1915–1929. https://doi.org/10.1109/tpami.2012.231
12. Tompson J, Jain A, Lecun Y, Bregler C (2014) Joint training of a convolutional network and a graphical model for human pose estimation. In: Proceedings of the 27th international conference on neural information processing systems, pp 1799–1807
13. Hinton GE, Osindero S, Teh YW (2006) A fast learning algorithm for deep belief nets. Neural Comput 18(7):1527–1554. https://doi.org/10.1162/neco.2006.18.7.1527
14. Bengio Y, Lamblin P, Popovici D, Larochelle H (2006) Greedy layer-wise training of deep networks. In: Proceedings of the 19th

international conference on neural information processing systems, pp 153–160
15. Rifai S, Dauphin YN, Vincent P, Bengio Y, Muller X (2011) The manifold tangent classifier. In: Proceedings of the 24th international conference on neural information processing systems, pp 2294–2302
16. Krizhevsky A, Sutskever I, Hinton GE (2012) ImageNet classification with deep convolutional neural networks. In: Annual conference on neural information processing systems, pp 1106–1114
17. Schmidhuber J (2012) Multi-column deep neural networks for image classification. In: Proceedings of the 2012 IEEE conference on computer vision and pattern recognition. IEEE Computer Society, 2354694, pp 3642–3649. https://doi.org/10.1109/cvpr.2012.6248110
18. Pan H, Wang B, Jiang H (2015) Deep learning for object saliency detection and image segmentation. arXiv
19. Mikolov T, Sutskever I, Chen K, Corrado G, Dean J (2013) Distributed representations of words and phrases and their compositionality. In: Annual conference on neural information processing systems, pp 3111–3119
20. Li Y, Xu L, Tian F, Jiang L, Zhong X, Chen E (2015) Word embedding revisited: a new representation learning and explicit matrix factorization perspective. In: International conference on artificial intelligence, pp 3650–3656
21. Le QV, Mikolov T (2014) Distributed representations of sentences and documents. In: Proceedings of the 31th international conference on machine learning, pp 1188–1196
22. Chien JT, Ku YC (2016) Bayesian recurrent neural network for language modeling. IEEE Trans Neural Netw Learn Syst 27(2):361–374. https://doi.org/10.1109/TNNLS.2015.2499302
23. Mikolov T, Corrado G, Chen K, Dean J, Mikolov T, Corrado G, Chen K, Dean J (2013) Efficient estimation of word representations in vector space. In: International conference on learning representations
24. Liben-Nowell D, Kleinberg J (2007) The link-prediction problem for social networks. J Assoc Inf Sci Technol 58(7):1019–1031. https://doi.org/10.1002/asi.v58:7
25. Ou M, Cui P, Pei J, Zhang Z, Zhu W (2016) Asymmetric transitivity preserving graph embedding. In: Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining, pp 1105–1114. https://doi.org/10.1145/2939672.2939751
26. Wang D, Cui P, Zhu W (2016) Structural deep network embedding. In: Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining, pp 1225–1234. https://doi.org/10.1145/2939672.2939753
27. Grover A, Leskovec J (2016) node2vec: scalable feature learning for networks. In: Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining, pp 855–864. https://doi.org/10.1145/2939672.2939754
28. Tu C, Liu H, Liu Z, Sun M (2017) CANE: context-aware network embedding for relation modeling. In: Proceedings of the 55th annual meeting of the association for computational linguistics, pp 1722–1731. https://doi.org/10.18653/v1/p17-1158
29. Zhou C, Liu Y, Liu X, Liu Z, Gao J (2017) Scalable graph embedding for asymmetric proximity. In: Proceedings of the 31st AAAI conference on artificial intelligence, pp 2942–2948
30. Ribeiro LFR, Saverese PHP, Figueiredo DR (2017) struc2vec: learning node representations from structural identity. In: Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining, pp 385–394. https://doi.org/10.1145/3097983.3098061
31. Yang Z, Cohen WW, Salakhutdinov R (2016) Revisiting semi-supervised learning with graph embeddings. In: Proceedings of the 33nd international conference on machine learning, pp 40–48
32. Perozzi B, Al-Rfou R, Skiena S (2014) DeepWalk: online learning of social representations. In: Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, pp 701–710. https://doi.org/10.1145/2623330.2623732
33. Tang J, Qu M, Wang M, Zhang M, Yan J, Mei Q (2015) LINE: large-scale information network embedding. In: Proceedings of the 24th international conference on world wide web, pp 1067–1077. https://doi.org/10.1145/2736277.2741093
34. Cao S, Lu W, Xu Q (2015) GraRep: learning graph representations with global structural information. In: Proceedings of the 24th ACM international conference on information and knowledge management, pp 891–900. https://doi.org/10.1145/2806416.2806512
35. Yang C, Liu Z, Zhao D, Sun M, Chang EY (2015) Network representation learning with rich text information. In: Proceedings of the twenty-fourth international joint conference on artificial intelligence, pp 2111–2117
36. Tu C, Zhang W, Liu Z, Sun M (2016) Max-margin deepwalk: discriminative learning of network representation. In: Proceedings of the twenty-fifth international joint conference on artificial intelligence, pp 3889–3895
37. Li J, Zhu J, Zhang B (2016) Discriminative deep random walk for network classification. In: Proceedings of the 54th annual meeting of the association for computational linguistics, pp 1004–1013
38. Pan S, Wu J, Zhu X, Zhang C, Wang Y (2016) Tri-party deep network representation. In: Proceedings of the twenty-fifth international joint conference on artificial intelligence, pp 1895–1901
39. Wang X, Cui P, Wang J, Pei J, Zhu W, Yang S (2017) Community preserving network embedding. In: Proceedings of the 31st AAAI conference on artificial intelligence, pp 203–209
40. Huang X, Li J, Hu X (2017) Label informed attributed network embedding. In: Proceedings of the tenth ACM international conference on web search and data mining, pp 731–739. https://doi.org/10.1145/3018661.3018667
41. Li B, Pi D (2019) Learning deep neural networks for node classification. Expert Syst Appl 137:324–334. https://doi.org/10.1016/j.eswa.2019.07.006
42. Chen J, Zhang Q, Huang X (2016) Incorporate group information to enhance network embedding. In: Proceedings of the 25th ACM international conference on information and knowledge management, pp 1901–1904. https://doi.org/10.1145/2983323.2983869
43. Cao S, Lu W, Xu Q (2016) Deep neural networks for learning graph representations. In: Proceedings of the 30th AAAI conference on artificial intelligence, pp 1145–1152
44. Cavallari S, Zheng VW, Cai H, Chang CC, Cambria E (2017) Learning community embedding with community detection and node embedding on graphs. In: ACM on conference on information and knowledge management, pp 377–386. https://doi.org/10.1145/3132847.3132925
45. Yang C, Liu M, Wang Z, Liu L, Han J (2017) Graph clustering with dynamic embedding. arXiv
46. Yan S, Xu D, Zhang B, Zhang HJ, Yang Q, Lin S (2007) Graph embedding and extensions: a general framework for dimensionality reduction. IEEE Trans Pattern Anal Mach Intell 29(1):40–51. https://doi.org/10.1109/TPAMI.2007.12
47. Tenenbaum JB, Silva VD, Langford JC (2000) A global geometric framework for nonlinear dimensionality reduction. Science 290(5500):2319–2323. https://doi.org/10.1126/science.290.5500.2319

48. Roweis ST, Saul LK (2000) Nonlinear dimensionality reduction by locally linear embedding. Science 290(5500):2323–2326. https://doi.org/10.1126/science.290.5500.2323
49. Belkin M, Niyogi P (2003) Laplacian Eigenmaps for dimensionality reduction and data representation. Neural Comput 15(6):1373–1396. https://doi.org/10.1162/089976603321780317
50. Vural E, Guillemot C (2016) Out-of-sample generalizations for supervised manifold learning for classification. IEEE Trans Image Process 25(3):1410–1424. https://doi.org/10.1109/TIP.2016.2520368
51. Hong D, Yokoya N, Zhu XX (2017) Learning a robust local manifold representation for hyperspectral dimensionality reduction. IEEE J Sel Top Appl Earth Obs Remote Sens 10(6):2960–2975. https://doi.org/10.1109/JSTARS.2017.2682189
52. Wang W, Yan Y, Nie F, Yan S, Sebe N (2018) Flexible manifold learning with optimal graph for image and video representation. IEEE Trans Image Process 27(6):2664–2675. https://doi.org/10.1109/TIP.2018.2810515
53. Cui P, Wang X, Pei J, Zhu W (2019) A survey on network embedding. IEEE Trans Knowl Data Eng 31(5):833–852. https://doi.org/10.1109/TKDE.2018.2849727
54. Moyano LG (2017) Learning network representations. Eur Phys J Spec Top 226(3):499–518. https://doi.org/10.1140/epjst/e2016-60266-2
55. Goyal P, Ferrara E (2018) Graph embedding techniques, applications, and performance: a survey. Knowl Based Syst 158:78–94. https://doi.org/10.1016/j.knosys.2018.03.022
56. Cai H, Zheng VW, Chang CC (2018) A comprehensive survey of graph embedding: problems, techniques and applications. IEEE Trans Knowl Data Eng 30:1616–1637. https://doi.org/10.1109/TKDE.2018.2807452
57. Hamilton WL, Ying R, Leskovec J (2017) Representation learning on graphs: methods and applications. IEEE Eng Bull 40:52–74
58. Keikha MM, Rahgozar M, Asadpour M (2018) Community aware random walk for network embedding. Knowl Based Syst 148:47–54. https://doi.org/10.1016/j.knosys.2018.02.028
59. Zhang Y, Shi Z, Feng D, Zhan X-X (2019) Degree-biased random walk for large-scale network embedding. Future Gener Comput Syst 100:198–209. https://doi.org/10.1016/j.future.2019.05.033
60. Kipf TN, Welling M (2017) Semi-supervised classification with graph convolutional networks. In: International conference on learning representations
61. Zhuang C, Ma Q (2018) Dual graph convolutional networks for graph-based semi-supervised classification. In: International world wide web conferences, pp 499–508. https://doi.org/10.1145/3178876.3186116
62. Gao H, Wang Z, Ji S (2018) Large-scale learnable graph convolutional networks. In: Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery and data mining, pp 1416–1424. https://doi.org/10.1145/3219819.3219947
63. Dai Q, Li Q, Tang J, Wang D (2018) Adversarial network embedding. In: Thirty-second AAAI conference on artificial intelligence
64. Veličković P, Cucurull G, Casanova A, Romero A, Liò P, Bengio Y (2018) Graph attention networks. In: International conference on learning representations
65. Wang H, Wang J, Wang J, Zhao M, Zhang W, Zhang F, Xie X, Guo M (2018) GraphGAN: graph representation learning with generative adversarial nets. In: Thirty-second AAAI conference on artificial intelligence
66. Pan S, Hu R, Long G, Jiang J, Yao L, Zhang C (2018) Adversarially regularized graph autoencoder for graph embedding. In: Proceedings of the twenty-seventh international joint conference on artificial intelligence, pp 2609–2615. https://doi.org/10.24963/ijcai.2018/362
67. Hu F, Zhu Y, Wu S, Wang L, Tan T (2019) Hierarchical graph convolutional networks for semi-supervised node classification. In: Proceedings of the twenty-eighth international joint conference on artificial intelligence. https://doi.org/10.24963/ijcai.2019/630
68. Gao H, Pei J, Huang H (2019) ProGAN: network embedding via proximity generative adversarial network. In: Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery and data mining, pp 1308–1316. https://doi.org/10.1145/3292500.3330866
69. Fu S, Liu W, Tao D, Zhou Y, Nie L (2020) HesGCN: Hessian graph convolutional networks for semi-supervised classification. Inf Sci 514:484–498. https://doi.org/10.1016/j.ins.2019.11.019
70. Debnath AK, Compadre RLLD, Debnath G, Shusterman AJ, Hansch C (1991) Structure–activity relationship of mutagenic aromatic and heteroaromatic nitro compounds. Correlation with molecular orbital energies and hydrophobicity. J Med Chem 34(2):786–797. https://doi.org/10.1021/jm00106a046
71. Toivonen H, Srinivasan A, King RD, Kramer S, Helma C (2003) Statistical evaluation of the predictive toxicology challenge 2000–2001. Bioinformatics 19(10):1183–1193. https://doi.org/10.1093/bioinformatics/btg130
72. Xinyi Z, Chen L (2019) Capsule graph neural network. In: International conference on learning representations
73. Borgwardt KM, Ong CS, Schönauer S, Vishwanathan SVN, Smola AJ, Kriegel H-P (2005) Protein function prediction via graph kernels. Bioinformatics 21(1):47–56. https://doi.org/10.1093/bioinformatics/bti1007
74. Wale N, Karypis G (2008) Comparison of descriptor spaces for chemical compound retrieval and classification. Knowl Inf Syst 14(3):347–375. https://doi.org/10.1109/ICDM.2006.39
75. Leskovec J, Kleinberg JM, Faloutsos C (2005) Graphs over time: densification laws, shrinking diameters and possible explanations. In: Proceedings of the eleventh ACM SIGKDD international conference on knowledge discovery and data mining, pp 177–187. https://doi.org/10.1145/1081870.1081893
76. Yanardag P, Vishwanathan SVN (2015) Deep graph kernels. In: ACM SIGKDD international conference on knowledge discovery and data mining, pp 1365–1374. https://doi.org/10.1145/2783258.2783417
77. Fouss F, Pirotte A, Renders JM, Saerens M (2007) Random-walk computation of similarities between nodes of a graph with application to collaborative recommendation. IEEE Trans Knowl Data Eng 19(3):355–369. https://doi.org/10.1109/TKDE.2007.46
78. Mnih A, Hinton G (2009) A scalable hierarchical distributed language model. In: Proceedings of the 21st international conference on neural information processing systems, pp 1081–1088
79. Feng R, Yang Y, Hu W, Wu F, Zhuang Y (2018) Representation learning for scale-free networks. In: Thirty-second AAAI conference on artificial intelligence
80. Chen H, Perozzi B, Hu Y, Skiena S (2018) HARP: hierarchical representation learning for networks. In: Thirty-second AAAI conference on artificial intelligence
81. Feng N, Recht B, Re C, Wright SJ (2011) Hogwild: a lock-free approach to parallelizing stochastic gradient descent. In: Proceedings of the 24th international conference on neural information processing systems, pp 693–701
82. Bottou L (1998) Online algorithms and stochastic approximations. Cambridge University Press, Cambridge

83. Hochstenbach ME (2009) A Jacobi–Davidson type method for the generalized singular value problem. Linear Algebra Appl 431(3–4):471–487. https://doi.org/10.1016/j.laa.2009.03.003
84. Weimer M, Karatzoglou A, Smola A (2008) Improving maximum margin matrix factorization. Mach Learn 72(3):263–276. https://doi.org/10.1007/978-3-540-87479-9_12
85. Lin CJ (2007) Projected gradient methods for nonnegative matrix factorization. Neural Comput 19(10):2756–2779. https://doi.org/10.1162/neco.2007.19.10.2756
86. Guillamet D, Vitrià J, Schiele B (2003) Introducing a weighted non-negative matrix factorization for image classification. Pattern Recognit Lett 24(14):2447–2454. https://doi.org/10.1016/S0167-8655(03)00089-8
87. Liu W, Zheng N (2004) Non-negative matrix factorization based methods for object recognition. Pattern Recognit Lett 25(8):893–897. https://doi.org/10.1016/j.patrec.2004.02.002
88. Lee DD, Seung HS (2000) Algorithms for non-negative matrix factorization. In: Proceedings of the 13th international conference on neural information processing systems, pp 535–541
89. Klema VC, Laub AJ (1980) The singular value decomposition: its computation and some applications. IEEE Trans Autom Control 25(2):164–176. https://doi.org/10.1109/TAC.1980.1102314
90. Mnih V, Heess N, Graves A, Kavukcuoglu K (2014) Recurrent models of visual attention. In: Proceedings of the 27th international conference on neural information processing systems, pp 2204–2212
91. Donahue J, Krähenbühl P, Darrell T (2017) Adversarial feature learning. In: International conference on learning representations
92. Radford A, Metz L, Chintala S (2016) Unsupervised representation learning with deep convolutional generative adversarial networks. In: International conference on learning representations
93. Yu L, Zhang W, Wang J, Yu Y (2017) SeqGAN: sequence generative adversarial nets with policy gradient. In: Thirty-first AAAI conference on artificial intelligence
94. Li J, Monroe W, Shi T, Jean S, Ritter A, Jurafsky D (2017) Adversarial learning for neural dialogue generation. In: Proceedings of the 2017 conference on empirical methods in natural language processing, pp 2157–2169
95. Wang J, Yu L, Zhang W, Gong Y, Xu Y, Wang B, Zhang P, Zhang D (2017) IRGAN: a minimax game for unifying generative and discriminative information retrieval models. In: International ACM SIGIR conference on research and development in information retrieval, pp 515–524. https://doi.org/10.1145/3077136.3080786
96. Goodfellow IJ, Pouget-Abadie J, Mirza M, Xu B, Warde-Farley D, Ozair S, Courville A, Bengio Y (2014) Generative adversarial nets. In: International conference on neural information processing systems, pp 2672–2680
97. Bullinaria JA, Levy JP (2007) Extracting semantic representations from word co-occurrence statistics: a computational study. Behav Res Methods 39(3):510–526. https://doi.org/10.3758/BF03193020
98. Kipf TN, Welling M (2016) Variational graph auto-encoders. In: NIPS workshop on Bayesian deep learning
99. Wang C, Pan S, Hu R, Long G, Jiang J, Zhang C (2019) Attributed graph clustering: a deep attentional embedding approach. In: Proceedings of the twenty-eighth international joint conference on artificial intelligence, pp 3670–3676. https://doi.org/10.24963/ijcai.2019/509
100. Shi H, Fan H, Kwok JT (2020) Effective decoding in graph auto-encoder using triadic closure. In: Proceedings of the 34th AAAI conference on artificial intelligence
101. Tolstikhin I, Bousquet O, Gelly S, Schoelkopf B (2018) Wasserstein auto-encoders. In: International conference on learning representations
102. Donnat C, Zitnik M, Hallac D, Leskovec J (2018) Learning structural node embeddings via diffusion wavelets. In: Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery and data mining, pp 1320–1329. https://doi.org/10.1145/3219819.3220025
103. Shuman DI, Narang SK, Frossard P, Ortega A, Vandergheynst P (2013) The emerging field of signal processing on graphs: extending high-dimensional data analysis to networks and other irregular domains. IEEE Signal Process Mag 30(3):83–98. https://doi.org/10.1109/MSP.2012.2235192
104. Narayanan A, Chandramohan M, Chen L, Liu Y, Saminathan S (2016) subgraph2vec: learning distributed representations of rooted sub-graphs from large graphs. arXiv
105. Shervashidze N, Schweitzer P, Leeuwen EJ, Mehlhorn K, Borgwardt KM (2011) Weisfeiler–Lehman graph kernels. J Mach Learn Res 12(3):2539–2561
106. Vishwanathan SVN, Schraudolph NN, Kondor R, Borgwardt KM (2010) Graph kernels. J Mach Learn Res 11:1201–1242
107. Shervashidze N, Vishwanathan SVN, Petri TH, Mehlhorn K, Borgwardt KM (2009) Efficient graphlet kernels for large graph comparison. In: Proceedings of the twelfth international conference on artificial intelligence and statistics, pp 488–495
108. Shervashidze N, Borgwardt KM (2009) Fast subtree kernels on graphs. In: International conference on neural information processing systems, pp 1660–1668
109. Borgwardt KM, Kriegel HP (2005) Shortest-path kernels on graphs. In: IEEE International conference on data mining, pp 74–81. https://doi.org/10.1109/icdm.2005.132
110. Niepert M, Ahmed M, Kutzkov K (2016) Learning convolutional neural networks for graphs. In: Proceedings of the 33nd international conference on machine learning, pp 2014–2023
111. Luo Z, Liu L, Yin J, Li Y, Wu Z (2017) Deep learning of graphs with Ngram convolutional neural networks. IEEE Trans Knowl Data Eng 29(10):2125–2139. https://doi.org/10.1109/TKDE.2017.2720734
112. Mckay BD, Piperno A (2014) Practical graph isomorphism, II. J Symb Comput 60:94–112
113. Zhang M, Cui Z, Neumann M, Yixin C (2018) An end-to-end deep learning architecture for graph classification. In: Proceedings of the thirty-second AAAI conference on artificial intelligence, pp 4438–4445
114. Bruna J, Zaremba W, Szlam A, Lecun Y (2014) Spectral networks and locally connected networks on graphs. In: International conference on learning representations
115. Defferrard M, Bresson X, Vandergheynst P (2016) Convolutional neural networks on graphs with fast localized spectral filtering. In: The 30th conference on neural information processing systems
116. Ying R, You J, Morris C, Ren X, Hamilton W, Leskovec J (2018) Hierarchical graph representation learning with differentiable pooling. In: Proceedings of the 32nd international conference on neural information processing systems, pp 4805–4815
117. Chen F, Pan S, Jiang J, Huo H, Long G (2019) DAGCN: dual attention graph convolutional networks. In: International joint conference on neural networks, pp 1–8. https://doi.org/10.1109/ijcnn.2019.8851698
118. Lee JB, Rossi RA, Kong X (2018) Graph classification using structural attention. In: Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery and data mining, pp 1666–1674. https://doi.org/10.1145/3219819.3219980

119. Sabour S, Frosst N, Hinton GE (2017) Dynamic routing between capsules. In: Proceedings of the 31st international conference on neural information processing systems, pp 3859–3869
120. Zhu D, Cui P, Zhang Z, Pei J, Zhu W (2018) High-order proximity preserved embedding for dynamic networks. IEEE Trans Knowl Data Eng 30:2134–2144. https://doi.org/10.1109/TKDE.2018.2822283
121. Zhou L, Yang Y, Ren X, Wu F, Zhuang Y (2018) Dynamic network embedding by modeling triadic closure process. In: Thirty-second AAAI conference on artificial intelligence, vol 5, pp 393–406
122. Manessi F, Rozza A, Manzo M (2020) Dynamic graph convolutional networks. Pattern Recognit 97:107000. https://doi.org/10.1016/j.patcog.2019.107000
123. Hochreiter S, Schmidhuber J (1997) Long short-term memory. Neural Comput 9(8):1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735
124. Bordes A, Usunier N, Garcia-Duran A, Weston J, Yakhnenko O (2013) Translating embeddings for modeling multi-relational data. In: International conference on neural information processing systems, pp 2787–2795
125. Bian J, Chang Y, Fu Y, Chen WY (2012) Learning to blend vitality rankings from heterogeneous social networks. Neurocomputing 97(1):390–397. https://doi.org/10.1016/j.neucom.2012.06.024
126. Shen Z, Ma KL, Eliassi-Rad T (2006) Visual analysis of large heterogeneous social networks by semantic and structural abstraction. IEEE Trans Vis Comput Graph 12(6):1427–1439. https://doi.org/10.1109/TVCG.2006.107
127. Sun Y, Norick B, Han J, Yan X, Yu PS, Yu X (2012) Integrating meta-path selection with user-guided object clustering in heterogeneous information networks. In: The 18th ACM SIGKDD international conference on knowledge discovery and data mining, pp 1348–1356. https://doi.org/10.1145/2339530.2339738
128. Shi C, Li Y, Zhang J, Sun Y, Yu PS (2017) A survey of heterogeneous information network analysis. IEEE Trans Knowl Data Eng 29(1):17–37. https://doi.org/10.1109/TKDE.2016.2598561
129. Sun Y, Han J (2012) Mining heterogeneous information networks: a structural analysis approach. ACM SIGKDD Explor Newsl 14(2):20–28. https://doi.org/10.1145/2481244.2481248
130. Jacob Y, Denoyer L, Gallinari P (2014) Learning latent representations of nodes for classifying in heterogeneous social networks. In: Proceedings of the 7th ACM international conference on web search and data mining, New York, NY, USA. ACM, 2556225, pp 373–382. https://doi.org/10.1145/2556195.2556225
131. Nickel M, Tresp V, Kriegel HP (2011) A three-way model for collective learning on multi-relational data. In: International conference on machine learning, pp 809-816
132. Bordes A, Weston J, Collobert R, Bengio Y (2012) Learning structured embeddings of knowledge bases. In: AAAI Conference on artificial intelligence
133. Socher R, Chen D, Manning CD, Ng AY (2013) Reasoning with neural tensor networks for knowledge base completion. In: International conference on neural information processing systems, pp 926–934
134. Wang Z, Zhang J, Feng J, Chen Z (2014) Knowledge graph embedding by translating on hyperplanes. In: Twenty-eighth AAAI conference on artificial intelligence, pp 1112–1119
135. Lin Y, Liu Z, Sun M, Liu Y, Zhu X (2015) Learning entity and relation embeddings for knowledge graph completion. In: Twenty-ninth AAAI conference on artificial intelligence, pp 2181–2187
136. Yang B, Yih W, He X, Gao J, Deng L (2015) Embedding entities and relations for learning and inference in knowledge bases. In: International conference on learning representations
137. Ji G, He S, Xu L, Liu K, Zhao J (2015) Knowledge graph embedding via dynamic mapping matrix. In: Meeting of the association for computational linguistics and the international joint conference on natural language processing, pp 687–696
138. Jia Y, Wang Y, Lin H, Jin X, Cheng X (2016) Locally adaptive translation for knowledge graph embedding. In: Proceedings of the thirtieth conference on artificial intelligence, pp 992–998
139. Ji G, Liu K, He S, Zhao J (2016) Knowledge graph completion with adaptive sparse transfer matrix. In: Thirtieth AAAI conference on artificial intelligence, pp 985–991
140. Xie R, Liu Z, Sun M (2016) Representation learning of knowledge graphs with hierarchical types. In: International joint conference on artificial intelligence, pp 2965–2971
141. Nickel M, Rosasco L, Poggio T (2016) Holographic embeddings of knowledge graphs. In: Thirtieth AAAI conference on artificial intelligence, pp 1955–1961
142. Ruobing X, Zhiyuan L, Jia J, Huanbo L, Maosong S (2016) Representation learning of knowledge graphs with entity descriptions. In: Proceedings of the thirtieth AAAI conference on artificial intelligence
143. Xiao H, Huang M, Zhu X (2016) From one point to a manifold: knowledge graph embedding for precise link prediction. In: International joint conference on artificial intelligence
144. Welbl J, Riedel S, Bouchard G (2016) Complex embeddings for simple link prediction. In: International conference on machine learning, pp 2071–2080
145. Chen M, Tian Y, Yang M, Zaniolo C (2017) Multilingual knowledge graph embeddings for cross-lingual knowledge alignment. In: International joint conference on artificial intelligence
146. Liu H, Wu Y, Yang Y (2017) Analogical inference for multi-relational embeddings. In: International conference on machine learning, pp 2168–2178
147. Ebisu T, Ichise R (2018) TorusE: knowledge graph embedding on a Lie group. In: Thirty-second AAAI conference on artificial intelligence (AAAI-18)
148. Dettmers T, Minervini P, Stenetorp P, Riedel S (2018) Convolutional 2D knowledge graph embeddings. In: Thirty-second AAAI conference on artificial intelligence, pp 1811–1818
149. Guo S, Wang Q, Wang L, Wang B, Guo L (2018) Knowledge graph embedding with iterative guidance from soft rules. In: Thirty-second AAAI conference on artificial intelligence (AAAI-18)
150. Auer S, Bizer C, Kobilarov G, Lehmann J, Cyganiak R, Ives ZG (2007) DBpedia: a nucleus for a web of open data. In: International semantic web conference, pp 11–15
151. Bollacker K, Evans C, Paritosh P, Sturge T, Taylor J (2008) Freebase: a collaboratively created graph database for structuring human knowledge. In: SIGMOD conference, pp 1247–1250. https://doi.org/10.1145/1376616.1376746
152. Chen T, Sun Y (2017) Task-guided and path-augmented heterogeneous network embedding for author identification. In: Web search and data mining, pp 295–304. https://doi.org/10.1145/3018661.3018735
153. Shi Y, Gui H, Zhu Q, Kaplan L, Han J (2018) AspEm: embedding learning by aspects in heterogeneous information networks. In: Proceedings of the 2018 SIAM international conference on data mining. https://doi.org/10.1137/1.9781611975321.16
154. Fu TY, Lee WC, Lei Z (2017) HIN2Vec: explore meta-paths in heterogeneous information networks for representation learning. In: ACM International conference on information and

knowledge management, pp 1797–1806. https://doi.org/10.1145/3132847.3132953

155. Harper FM, Konstan JA (2016) The MovieLens datasets: history and context. ACM Trans Interact Intell Syst 5(4):19. https://doi.org/10.1145/2827872
156. Tu K, Cui P, Wang X, Wang F, Zhu W (2018) Structural deep embedding for hyper-networks. In: Thirty-second AAAI conference on artificial intelligence (AAAI-2018)
157. Zheng VW, Cao B, Zheng Y, Xie X, Yang Q (2010) Collaborative filtering meets mobile recommendation: a user-centered approach. In: Twenty-fourth AAAI conference on artificial intelligence
158. Schlichtkrull M, Kipf TN, Bloem P, Van Den Berg R, Titov I, Welling M (2018) Modeling relational data with graph convolutional networks. In: European semantic web conference. Springer, pp 593–607. https://doi.org/10.1007/978-3-319-93417-4_38
159. Nathani D, Chauhan J, Sharma C, Kaul M (2019) Learning attention-based embeddings for relation prediction in knowledge graphs. In: Proceedings of the 57th conference of the association for computational linguistics, pp 4710–4723. https://doi.org/10.18653/v1/p19-1466
160. Tang J, Qu M, Mei Q (2015) PTE: predictive text embedding through large-scale heterogeneous text networks. In: Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining, pp 1165–1174. https://doi.org/10.1145/2783258.2783307
161. Gui H, Liu J, Tao F, Jiang M, Norick B, Han J (2016) Large-scale embedding learning in heterogeneous event data. In: IEEE International conference on data mining, pp 907–912. https://doi.org/10.1109/icdm.2016.0111
162. Gui H, Liu J, Tao F, Jiang M, Norick B, Kaplan L, Han J (2017) Embedding learning with events in heterogeneous information networks. IEEE Trans Knowl Data Eng 29(11):2428–2441. https://doi.org/10.1109/TKDE.2017.2733530
163. Berge C (1989) Hypergraphs: combinatorics of finite sets. Elsevier, Amsterdam
164. Silva JG, Willett R (2009) Hypergraph-based anomaly detection of high-dimensional co-occurrences. IEEE Trans Pattern Anal Mach Intell 31(3):563–569. https://doi.org/10.1109/TPAMI.2008.232
165. Zhang C, Song D, Huang C, Swami A, Chawla NV (2019) Heterogeneous graph neural network. In: Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery and data mining, pp 793–803. https://doi.org/10.1145/3292500.3330961
166. Wang X, Ji H, Shi C, Wang B, Ye Y, Cui P, Yu PS (2019) Heterogeneous graph attention network. In: The world wide web conference, pp 2022–2032. https://doi.org/10.1145/3308558.3313562
167. Zhu J, Zhang J, He L, Wu Q, Zhou B, Zhang C, Yu PS (2017) Broad learning based multi-source collaborative recommendation. In: ACM International conference on information and knowledge management, pp 1409–1418. https://doi.org/10.1145/3132847.3132976
168. Zhang J, Xia C, Zhang C, Cui L, Fu Y, Yu PS (2017) BL-MNE: emerging heterogeneous social network embedding through broad learning with aligned autoencoder. In: International conference on data mining, pp 605–614. https://doi.org/10.1109/icdm.2017.70
169. Sun Y, Han J (2012) Mining heterogeneous information networks: principles and methodologies. Morgan & Claypool Publishers, San Rafael
170. Sun Y, Han J, Yan X, Yu PS, Wu T (2011) PathSim: meta path-based top-k similarity search in heterogeneous information networks. In: Proceedings of the VLDB endowment, vol 11, pp 992–1003
171. Dong Y, Chawla NV, 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, Halifax, NS, Canada. ACM, 3098036, pp 135–144. https://doi.org/10.1145/3097983.3098036
172. Zhang D, Yin J, Zhu X, Zhang C (2018) MetaGraph2Vec: complex semantic path augmented heterogeneous network embedding. In: Pacific-Asia conference on knowledge discovery and data mining. Springer, pp 196–208. https://doi.org/10.1007/978-3-319-93037-4_16
173. Shi C, Hu B, Zhao WX, Philip SY (2019) Heterogeneous information network embedding for recommendation. IEEE Trans Knowl Data Eng 31(2):357–370. https://doi.org/10.1109/TKDE.2018.2833443
174. Zhang H, Wang S, Xu X, Chow TW, Wu QJ (2018) Tree2Vector: learning a vectorial representation for tree-structured data. IEEE Trans Neural Netw Learn Syst 99:1–15. https://doi.org/10.1109/TNNLS.2018.2797060
175. Lu L, Zhang H (2015) A tree-structured representation for book author and its recommendation using multilayer SOM. In: 2015 International joint conference on neural networks (IJCNN). IEEE, pp 1–8. https://doi.org/10.1109/ijcnn.2015.7280530
176. Kazienko P, Kajdanowicz T (2012) Label-dependent node classification in the network. Neurocomputing 75(1):199–209. https://doi.org/10.1016/j.neucom.2011.04.047
177. Sun Y, Yuan Y, Wang G (2015) An on-line sequential learning method in social networks for node classification. Neurocomputing 149:207–214. https://doi.org/10.1016/j.neucom.2014.04.074
178. Bojchevski A, Günnemann S (2018) Deep Gaussian embedding of graphs: unsupervised inductive learning via ranking. In: International conference on learning representations
179. Fan R-E, Chang K-W, Hsieh C-J, Wang X-R, Lin C-J (2008) LIBLINEAR: a library for large linear classification. J Mach Learn Res 9:1871–1874
180. Peng W, Baowen X, Yurong W, Xiaoyu Z (2015) Link prediction in social networks: the state-of-the-art. Sci China Inf Sci 58(1):1–38. https://doi.org/10.1007/s11432-014-5237-y
181. Cannistraci CV, Alanis-Lobato G, Ravasi T (2013) Minimum curvilinearity to enhance topological prediction of protein interactions by network embedding. Bioinformatics 29(13):199–209. https://doi.org/10.1093/bioinformatics/btt208
182. Ma J, Cui P, Zhu W (2018) DepthLGP: learning embeddings of out-of-sample nodes in dynamic networks. In: Thirty-second AAAI conference on artificial intelligence
183. Fout A, Byrd J, Shariat B, Benhur A (2017) Protein interface prediction using graph convolutional networks. In: Proceedings of the 31st international conference on neural information processing systems, pp 6533–6542
184. Godfrey P, Gryz J, Lasek P (2016) Interactive visualization of large datasets. IEEE Trans Knowl Data Eng 28(8):2142–2157. https://doi.org/10.1109/TKDE.2016.2557324
185. Maaten LVD, Hinton G (2008) Visualizing data using t-SNE. J Mach Learn Res 9(2605):2579–2605
186. Tang J, Liu J, Zhang M, Mei Q (2016) Visualizing large-scale and high-dimensional Data. In: International conference on world wide web, vol 7, pp 287–297. https://doi.org/10.1145/2872427.2883041
187. Bobadilla J, Ortega F, Hernando A (2013) Recommender systems survey. Knowl Based Syst 46(1):109–132. https://doi.org/10.1016/j.knosys.2013.03.012
188. Chumki B, Haym H, Cohen W (1998) Recommendation as classification: using social and content-based information in

recommendation. In: Proceedings of the fifteenth national conference on artificial intelligence and tenth innovative applications of artificial intelligence conference, pp 714–720
189. Sarwar B, Karypis G, Konstan J, Riedl J (2001) Item-based collaborative filtering recommendation algorithms. In: International conference on world wide web, pp 285–295
190. Kitsak M, Gallos LK, Havlin S, Liljeros F, Muchnik L, Stanley HE, Makse HA (2010) Identification of influential spreaders in complex networks. Nat Phys 6(11):888–893. https://doi.org/10.1038/nphys1746
191. Schietgat L, Ramon J, Bruynooghe M (2013) A polynomial-time maximum common subgraph algorithm for outerplanar graphs and its application to chemoinformatics. Ann Math Artif Intell 69(4):343–376. https://doi.org/10.1007/s10472-013-9335-0
192. Deshpande M, Kuramochi M, Karypis G (2005) Frequent substructure-based approaches for classifying chemical compounds. In: IEEE International conference on data mining, pp 35–42. https://doi.org/10.1109/tkde.2005.127
193. Cheng H, Lo D, Zhou Y, Wang X, Yan X (2009) Identifying bug signatures using discriminative graph mining. In: Proceedings of the eighteenth international symposium on software testing and analysis, pp 141–152. https://doi.org/10.1145/1572272.1572290
194. Zhu X, Zhang C, Pan S, Yu PS (2013) Graph stream classification using labeled and unlabeled graphs. In: IEEE International conference on data engineering, pp 398–409. https://doi.org/10.1109/icde.2013.6544842
195. Dai H, Dai B, Song L (2016) Discriminative embeddings of latent variable models for structured data. In: International conference on machine learning, pp 2702–2711
196. You J, Liu B, Ying R, Pande V, Leskovec J (2018) Graph convolutional policy network for goal-directed molecular graph generation. In: Thirty-second conference on neural information processing systems
197. Wang Q, Mao Z, Wang B, Guo L (2017) Knowledge graph embedding: a survey of approaches and applications. IEEE Trans Knowl Data Eng 29(12):2724–2743. https://doi.org/10.1109/TKDE.2017.2754499
198. Dai H, Khalil EB, Zhang Y, Dilkina B, Song L (2017) Learning combinatorial optimization algorithms over graphs. In: Proceedings of the 31st international conference on neural information processing systems, pp 6351–6361
199. Sutton RS, Barto AG (1998) Reinforcement learning: an introduction. MIT Press, Cambridge
200. Sun M, Zhao S, Gilvary C, Elemento O, Zhou J, Wang F (2019) Graph convolutional networks for computational drug development and discovery. Brief Bioinform. https://doi.org/10.1093/bib/bbz042
201. Guo K, Hu Y, Qian Z, Liu H, Zhang K, Sun Y, Gao J, Yin B (2020) Optimized graph convolution recurrent neural network for traffic prediction. IEEE Trans Intell Transp Syst. https://doi.org/10.1109/TITS.2019.2963722
202. Kingma DP, Ba J (2014) Adam: a method for stochastic optimization. In: International conference on learning representations
203. Ahmed A, Shervashidze N, Narayanamurthy S, Josifovski V, Smola AJ (2013) Distributed large-scale natural graph factorization. In: International conference on world wide web, pp 37–48. https://doi.org/10.1145/2488388.2488393
204. Neumann M, Garnett R, Bauckhage C, Kersting K (2016) Propagation kernels: efficient graph kernels from propagated information. Mach Learn 102(2):209–245. https://doi.org/10.1007/s10994-015-5517-9
205. Simonovsky M, Komodakis N (2017) Dynamic edge-conditioned filters in convolutional neural networks on graphs. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 3693–3702. https://doi.org/10.1109/cvpr.2017.11
206. Duchi J, Hazan E, Singer Y (2011) Adaptive subgradient methods for online learning and stochastic optimization. J Mach Learn Res 12(7):257–269
207. Bordes A, Usunier N, Garcia-Duran A, Weston J, Yakhnenko O (2013) Translating embeddings for modeling multi-relational data. In: Proceedings of the 26th international conference on neural information processing systems, pp 2787–2795
208. Thomas JM, Muscoloni A, Ciucci S, Bianconi G, Cannistraci CV (2017) Machine learning meets complex networks via coalescent embedding in the hyperbolic space. Nat Commun 8(1):1615. https://doi.org/10.1038/s41467-017-01825-5
209. Chamberlain BP, Clough J, Deisenroth MP (2017) Neural embeddings of graphs in hyperbolic space. arXiv
210. Dai H, Li H, Tian T, Huang X, Wang L, Zhu J, Song L (2018) Adversarial attack on graph structured data. In: International conference on machine learning, pp 1123–1132. https://doi.org/10.1145/3219819.3220078
211. Zügner D, Akbarnejad A, Günnemann S (2018) Adversarial attacks on neural networks for graph data. In: Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery and data mining, London, UK. ACM, pp 2847–2856
212. Goodfellow IJ, Shlens J, Szegedy C (2015) Explaining and harnessing adversarial examples. In: International conference on learning representations
213. Battaglia PW, Hamrick JB, Bapst V, Sanchez-Gonzalez A, Zambaldi V, Malinowski M, Tacchetti A, Raposo D, Santoro A, Faulkner R, Gulcehre C, Song F, Ballard A, Gilmer J, Dahl G, Vaswani A, Allen K, Nash C, Langston V, Dyer C, Heess N, Wierstra D, Kohli P, Botvinick M, Vinyals O, Li Y, Pascanu R (2018) Relational inductive biases, deep learning, and graph networks. arXiv
214. Bareinboim E, Pearl J (2016) Causal inference and the data-fusion problem. Proc Natl Acad Sci 113(27):7345–7352. https://doi.org/10.1073/pnas.1510507113
215. Cao B, He L, Xiaokai, Xing M, Yu PS, Klumpp H, Leow AD (2017) t-BNE: tensor-based brain network embedding. In: SIAM International conference on data mining, pp 189–197. https://doi.org/10.1137/1.9781611974973.22
216. Ma G, Lu CT, He L, Yu PS, Ragin AB (2017) Multi-view graph embedding with hub detection for brain network analysis. In: IEEE International conference on data mining, pp 967–972. https://doi.org/10.1109/icdm.2017.123
217. Camacho DM, Collins KM, Powers RK, Costello JC, Collins JJ (2018) Next-generation machine learning for biological networks. Cell 173(7):1581–1592. https://doi.org/10.1016/j.cell.2018.05.015