IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received March 12, 2021, accepted March 31, 2021, date of publication April 5, 2021, date of current version April 27, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3071274

Graph Neural Network: A Comprehensive Review on Non-Euclidean Space

NURUL A. ASIF[1], YEAHIA SARKER[1], (Student Member, IEEE), RIPON K. CHAKRABORTTY[2], (Member, IEEE), MICHAEL J. RYAN[2], (Senior Member, IEEE), MD. HAFIZ AHAMED[1], DIP K. SAHA[1], FAISAL R. BADAL[1], SAJAL K. DAS[1], (Member, IEEE), MD. FIROZ ALI[1], SUMAYA I. MOYEEN[1], MD. ROBIUL ISLAM[1], (Member, IEEE), AND ZINAT TASNEEM[1], (Member, IEEE)

[1]Department of Mechatronics Engineering, Rajshahi University of Engineering & Technology, Rajshahi 6204, Bangladesh
[2]School of Engineering and Information Technology, University of New South Wales (UNSW) at Canberra, Canberra, ACT 2610, Australia

Corresponding author: Nurul A. Asif (asifamin.ruet@gmail.com)

ABSTRACT This review provides a comprehensive overview of the state-of-the-art methods of graph-based networks from a deep learning perspective. Graph networks provide a generalized form to exploit non-euclidean space data. A graph can be visualized as an aggregation of nodes and edges without having any order. Data-driven architecture tends to follow a fixed neural network trying to find the pattern in feature space. These strategies have successfully been applied to many applications for euclidean space data. Since graph data in a non-euclidean space does not follow any kind of order, these solutions can be applied to exploit the node relationships. Graph Neural Networks (GNNs) solve this problem by exploiting the relationships among graph data. Recent developments in computational hardware and optimization allow graph networks possible to learn the complex graph relationships. Graph networks are therefore being actively used to solve many problems including protein interface, classification, and learning representations of fingerprints. To encapsulate the importance of graph models, in this paper, we formulate a systematic categorization of GNN models according to their applications from theory to real-life problems and provide a direction of the future scope for the applications of graph models as well as highlight the limitations of existing graph networks.

INDEX TERMS Graph neural network, geometric deep learning, graph-structured network, non-euclidean space.

I. INTRODUCTION

The graph is a variant of data structure that learns the relationship between nodes and explores the relationship among these nodes. Graph nets have gained researchers' attention due to the recent progress in computational devices. A wide range of applications using Graph Neural Networks (GNNs) demonstrates the potential of graph reasoning nets in order to explore modern problems [1]–[7]. Several biological structures can be represented as graphs including the brain, vascular system, and nervous system. Additionally, the inter-molecular relationships among chemical systems can also be visualized as graphs. Link prediction is one of the most significant topics in graph networks which allows data to be visualized as graphs in which there are edges and nodes. Due to having a non-euclidean structure, graph reasoning models are generally applied to node classification and link prediction [8]. Simply stated, GNN follows deep learning strategies to exploit relational information via the graph. Additionally, GNN also picks up the important edges and nodes to perform a given task using filtering algorithms.

Graph nets also utilize shared weighted local connections in order to exploit relations among graph elements. Fig. 1 shows a general architecture of a graph neural network where the input graph is fed into the hidden nodes to learn the representations of graph-structured data, and the output graph is generated from the learned graph-structured representations. From the observation on convolution operation and graphs, it can be concluded that Convolution models have a large impact on designing graph reasoning models as the shared

The associate editor coordinating the review of this manuscript and approving it for publication was Hiram Ponce.

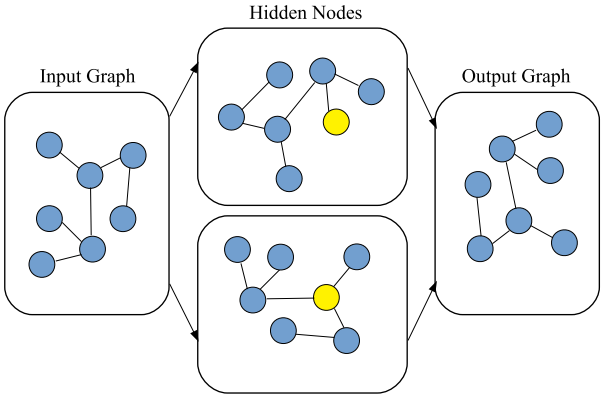


FIGURE 1. **Illustration of graph neural network.**

weights feature dramatically reduces the computational cost compared with traditional spectral graph theory [9]. Interconnected layers develop a bridge in order to extract variant patterns of various sizes. As CNN can extract useful features from 2D grid images or 1D text sequences, this strategy is also used in graph exploitation as instances. To exploit the necessity of sub-graphs, graph embedding is utilized to express the relations among nodes, edges, and sub-graphs. Graph embeddings can define nodes and edges in low-dimensional vectors for further classification or regression.

A directed graph can be represented as the graph direction from one node to another node. Directed graph and focuses on a particular node direction containing useful information for graph representation learning. On the other hand, an undirected graph can be defined as two directed edges presenting the relationship between two nodes. It is to be noted that a graph will be undirected only if the adjacency matrix is symmetric. It is worth noting that the directed graph contains more valuable information than the undirected graph. This phenomenon can be observable in knowledge graph feature learning [10]. On the other hand, the heterogeneous graph resembles a collection of various nodes. This graph variation can be processed by simply converting them into one-hot feature vector added to the original vector representation. The heterogeneous graph information can work better with meta path information propagation. With the meta path, the neighbour nodes can be categorized as the distances among nodes and types. For example, the Graph Inception model uses this meta path information propagation system for a better understanding of graph structure data [11]. This network handles the heterogeneous graph converting it into sub-graph for the information propagation and finally adds the results from different sub-graphs for node feature learning process.

To exploit graph-structured data, graph filtering is usually applied to harvest features or remove irrelevant node relations. Graph filtering is a filtering process that operates on graph data as input and gives an output graph signal. The filtering process can be done on the spatial domain or spectral domain. The graph filtering process can be divided into sub-filtering methods – frequency filtering and vertex filtering. Generally, the graph filtering operation can be expressed as a convolution filter on graph-structured data in either the spatial or time domain. However as graph data does not have any fixed pattern, graph convolution is quite different to traditional signal processing. It is should be noted that the time-domain dependent convolution filter is the inverse process of Fourier multiplication between spectral featured data. The graph filtering process can be expressed as linear combinations of signal components in the vertex field. Mathematically, it can be defined as:

$$x_{\text{out}}(i) = w_{i,i}x(i) + \sum_{j \in R(i,K)} w_{i,j}x(j) \tag{1}$$

The neighbourhood nodes in equation 1 can be expressed as $R(i, K)$ and the combination of the weights can be represented as $\{w_{i,j}\}$.Using of K-polynomial filter, the frequency filters can be presented as vertex filters.

The graph Fourier transform (GFT) is an important element of graph nets often used for frequency analysis. GFT can be presented as 1D signal g measured by $\hat{g}(\xi) = \langle g, e^{2\pi i \xi t} \rangle$. The ξ can be denoted as the frequency of $\hat{f}$. It can also be defined as the complex exponential of an eigen function of the Laplace operator. The eigenvector of M containing a fixed value is the complex exponential of a fixed frequency. The random-walk transition matrix can be presented as a graph Laplace operator. The eigen decomposition of $\tilde{M} = U \Lambda U^T$ contains the corresponding eigenvalue. Now, the Fourier transform on graph y can be presented as

$$\hat{y}(\lambda_l) = \langle y, u_l \rangle = \sum_{j=1}^{n} y(j) u_l^*(j) \tag{2}$$

Which defines the spectral graph. The vertex graph can be expressed as

$$y(j) = \sum_{l=1}^{n} \hat{y}(\lambda_l) u_l(j) \tag{3}$$

These filtering processes and preprocessing techniques are commonly used in the data-driven graph exploitation methods. The success of DL methods to learn representations of euclidean data motivated researchers to use the strength of data-driven methods for non-euclidean data. The application of GNN includes e-commerce where a graph-based technique can be applied in order to explore the interactions among users rating, number of items bought, and products [12]. Aside from this application, GNNs can also be applicable to social science problems. Fig. 2 shows the exploitation of molecular structure in the form of a graph net. GNNs have also been used in text classification providing solutions for one of the most complicated problems in NLP. Peng *et al.* [13] proposed a graph-CNN methods transforming text into graph embeddings for the further classification process. Traditional convolutional and recurrent networks exploit the features in a predefined order. On the other hand, graph models do not contain a natural order for nodes and edges. This makes those traditional models fail to explore the possible representation of graphs. To express the relationship among all possible

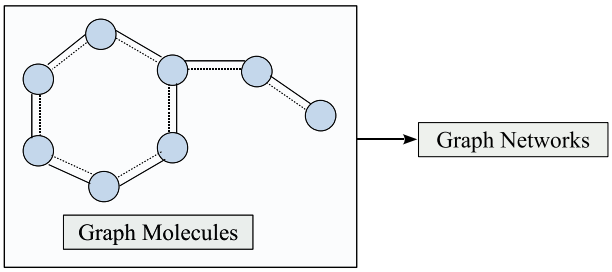


FIGURE 2. Graph molecule embedding structure for graph network.

orders of the graph, the input sequence of these models cannot handle properly. To solve this issue, GNN utilizes the propagation method on all nodes overlooking the order of the nodes. GNNs update the weights of aggregated neighborhood nodes, so the information propagation system of GNN is supported by a graphical structure instead of features of nodes.

In GNNs, relation extraction is one of the primary tasks that learns the relationship between entities. In some literature, this setting can also be defined as entity recognition. In [14], a tree-structured LSTM-RNNs have been developed for complete node feature learning. To explore the relationships among entities, a GCN has been introduced with a pruning technique for the input items. The cross-sentence N-array entity recognition technique utilizes graph algorithms to learn the relationships among multiple sentences. For this setting, LSTM networks are generally used to establish the best relationship among various sentences. The application of GNNs can also be noticed at event extraction that recognizes event types that exist in data. Liu *et al.* [15] proposed a joint event extraction system that increases information flow using attention-guided GCNs.

To explore these applications of graph-based networks multiple reviews have been observed. Lee *et al.* [16] provided a detailed review of the graph-based attention networks. Three different categorizations have been introduced and some methods related to the categorizations are explained. The author focuses on the superiority of the attention module and describes the taxonomy according to the attention mechanism. Despite having an excellent categorization, the review only focuses on the attention units. This makes the literature limited to the special domain of graph-based models.

In [17], A GCN-based categorization has been proposed in the literature where the GCN-oriented methods have been divided into the spatial and spectral category and from the application perspective, GCN has been divided into three sections : 1) Computer Vision, 2) Natural Language Processing, and 3) Science. It is to be noted that this survey does not focus on the analysis of spatial and spectral-based methods rather than rely on the categorization. Moreover, from the application point of view, only three types of applications have been introduced.

Bronstein *et al.* [18] introduced a review on GCN focusing non-euclidean space. This review divides the GCN methods into two sections — spatial and spectral-based methods. The classification of spectral and spatial based methods depends on the types of the convolution operation. As it only focuses on convolution filters, other types of network structures for graph models are overlooked. This makes the review limited to only GCN models.

To overcome the limitation of the existing graph-based reviews, we introduce a new taxonomy accommodating both the theoretical and application perspectives. For a solution to the scarcity of graph-based networks to individual problems, we provide abundant resources of graph-based models along with their applications and core-algorithms.

The main contribution of this survey is four fold:

- In contrast to the spatial and spectral-based graph networks, an application-oriented graph-based taxonomy has been proposed focusing on both theoretical and real-life processes.
- We demonstrate a detailed algorithmic overview of the SOTA methods focusing on both theoretical and application perspective including graph classification and generation.
- Useful resources of graph-based networks have been provided along with their core algorithm from both theoretical and application perspectives.
- We provide a future direction for application-oriented graph-based networks as well as discuss the limitations of existing solutions.

The residue of the manuscript is organized as follows. Section II provides the preliminary definitions and section II presents the background studies. In section IV, we provide an algorithmic overview of SOTA methods according to categorization. Section V shows a clear application-oriented taxonomy. In section VI, we provide some directions focusing on the application of graph-based models. Section V presents the concluding remark.

II. BACKGROUND STUDIES

This section explains the commonly studied neural architecture used for graph exploitation. We discuss both discriminative and generative models along with the attention module and sequential model for better perception of different graph-based architectures.

A. CONVOLUTIONAL NEURAL NETWORKS

CNN has become the backbone of data-driven models for computer vision tasks [19]. Many applications of standard CNN are being used in many sectors [20]. Modern machine vision architectures utilize CNN or convolution operation for learning complicated features from image patches [21]. The usage of CNN is also noticeable in exploiting graph networks. Fully convolutional architecture uses convolution operation for representation learning and standard CNN utilizes fully connected layers by flattening into a single dimension. For CNN, every pixel is given as input for the input layer which is why the input layer size is showed as $n_1 \times 1$ where n_1 is the number of input channels. The $n_1 \times 1$ input vector through t kernels with the size of $k_1 \times 1$ is filtered by the hidden

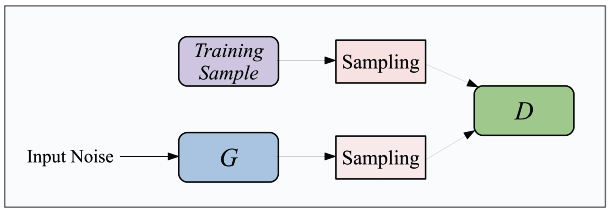


FIGURE 3. Illustration of GAN. *G* defines the generator that samples fake data to the *D*. *D* denotes the discriminator that computers the probability whether the sample is real or fake.

convolution layer. The Convolution layer nodes number can be represented as $t \times n_2 \times 1$, and $n_2 = n_1 - k_1$. The convolutional layer activation map can be obtained as:

$$y^i(p) = max\left(0, b^{j(p)} + \sum_j k^{ij(p)} * x^{i(p)}\right) \tag{4}$$

where, x^{ip} and y^{jp} are defined as the ith input and the jth output activation map respectively. $b^{j(r)}$ is the bias of the jth output map and * represents convolution. $k^{ij(p)}$ is represented as the convolution kernel between the ith input map and the jth output map.

B. GENERATIVE MODELS

In the domain of machine learning, two approaches are highly appreciated — discriminating learning and generative learning. Many network architectures have been proposed including auto-regressive networks, Markov models, variational autoencoder, generative adversarial network. These methods are highly appreciated in many real-life applications [22]–[26]. Recently, generative models are being exploited to understand the graph structure and achieved excelled performance on graph data [27]. Among generative models, GAN has gained popularity due to its adversarial training process. Fig. 3 depicts the vanilla generative adversarial network architecture. At first, Goodfellow *et al.* [28] proposed F_C layers for both generator G and discriminator P_D. This approach was utilized on several datasets including MNIST [29], CIFAR-10 [30] and Toronto face dataset. The generator develops a mapping from noise distribution p_z to a data point $G(z)$. The generator tries to fool the discriminator by generating fake samples and maps a distribution p_g over real data X. G generates synthetic data through an adversarial training process appearing as realistic as real data distribution. So, the objective function of G is:

$$\min_G E_{Z\sim P_z}[log(1 - P_d\ (G(z))\ )] \tag{5}$$

$P_d(*)$ defines the probability that the possible data distribution is from real data rather than generated data. Equation (5) is minimized if $\mathbf{P_d}$ is wrong and is maximized if P_d is right. The goal of $\mathbf{P_d}$ is to improve its classification accuracy to distinguish between real data and fake data. Therefore, the objective function of $\mathbf{P_d}$ becomes:

$$\max_D E_{X\sim P_{data}}\left[\log P_d\ (x)\right] + E_{Z\sim P_z}\left[\log\left(1 - P_d\ (G(z))\right)\right] \tag{6}$$

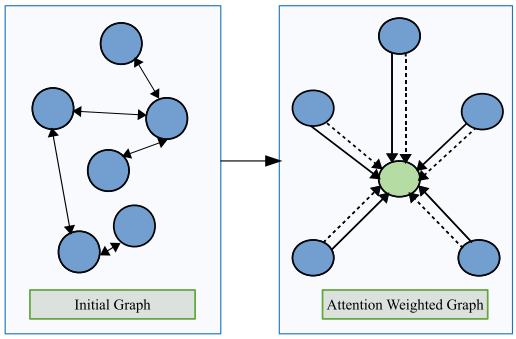


FIGURE 4. Multi-head attention representation of graphs.

The total objection function of the generative model follows a min-max game that can be expressed as:

$$\min_G \max_{P_d} E_{X\sim P_{data}}\left[\log P_d\ (x)\right] + E_{Z\sim P_z}\left[\log\left(1 - P_d\ (G(z))\right)\right] \tag{7}$$

C. ATTENTION MECHANISM

Attention mechanism was first introduced as self-attention mechanism in [31] in order to perform computer vision tasks inspired by the way that retinas fixate on necessary parts of the optic array. However, for NLP this idea has been utilized to do machine translation tasks. Then the term 'visual attention' has become popular boosting the classification performance of spatial images. Aside from boosting the results, the attention module can effectively interpret aspects of neural architecture that are quite difficult to understand. This mechanism can also adaptively focuses on important parts of the image or text and can simply ignore the irrelevant parts. A robust attention mechanism can fabricate a relationship between the data-driven neural structure and intuitive representation learning. Computed weights by the attention module can accurately harvest effective structural image patches and provide an excellent explanation of the neural system. Attention mechanism has been widely used in many tasks [32]–[35]. With significant performance in NLP, attention mechanism has also gained popularity due to its robust feature learning process in computer vision. Several types of application of attention mechanism are noticeable in different sectors of computer vision including [36]–[39]. To learn long-term contextual details, self-attention module has been adopted in these methods. This module enables parallelizability leveraging modern computational devices developing long-distance interactions. The Self-attention unit can also be used as a standalone-unit for neural computation [40]. To learn spatial dependencies, spatial-attention is also utilized in those methods. It can be mathematically expressed as:

$$y_{ij} = \sum_{a,b\in N_k(i,j)} \text{softmax}_{ab}(q_{ij}^T k_{ab} + q_{ij}^T r_{a-i,b-j}) v_{ab} \tag{8}$$

In equation (8), $r_{a-i,b-j}$ is the concatenation representations of row and column vectors. The measured probability

between the query and elements are parameterized with the element position $N_k(i, j)$ and distance.

D. LONG SHORT-TERM MEMORY MODELS

Hochreiter and Schmidhuber *et al.* [41] proposed the long short-term memory (LSTM) model that reduces the gradient vanishing problem of RNN. The LSTM model consists of recurrent networks where every node of hidden layers is exchanged by memory cells. Each memory cell of LSTM is composed of self-connected recurrent edge that has a fixed weight. This allows the gradient to flow throughout the network without vanishing. The memory cells in LSTM can be denoted as c. LSTM is developed on the idea of holding long term information for a certain time period. The LSTM model leverages memory storage in the form of short-term activations.

The input node receives the input activation from the input sequence $x^{(t)}$ at the present time step and the current time step is processed from the previous time step $h^{(t-1)}$ which can be defined as hidden states. Generally, the weighted sum is calculated using the *tanh* activation function, but the main paper [41] utilizes the sigmoid activation function. Individual memory cell is connected via linear activation that can be denoted as $s^{(t)}$. As already mentioned, there exist self-connected recurrent cells that hold the internal state. This edge flows according to time steps with fixed weighted value without causing gradient exploding. The vectorized expression of internal state is as $\boldsymbol{s}^{(t)} = \boldsymbol{g}^{(t)} \cdot \boldsymbol{i}^{(t)} + \boldsymbol{s}^{(t-1)}$, where $\cdot$ is element-wise multiplication.

In [42], the forget gate was introduced demonstrating a technique to delete the contents of the internal state. It helps the gradient to flow throughout the network smoothly. Now, the internal state with forget gate can be described as:

$$\boldsymbol{s}^{(t)} = \boldsymbol{g}^{(t)} \odot \boldsymbol{i}^{(t)} + \boldsymbol{f}^{(t)} \odot \boldsymbol{s}^{(t-1)} \tag{9}$$

The output value v_c processed by the internal state is multiplied by output gate value o_c. Usually, in the output gate, for a non-linear function, the rectified linear unit is utilized that has a greater range than other activation functions.

In the forward gradient propagation, the internal state controls the gradient through activation. When the internal cell and output cell are closed, the activation is trapped inside the memory cell without any change to the intermediate time steps. In backward gradient propagation, the constant error carousel makes the model backpropagate following the time steps. Multiple memory cells improve the LSTM network to learn more dependencies from the input sequence. This model successfully removes the existing problem of recurrent networks and proves itself as a robust sequential model. LSTM model as a sequential model is being used in many sectors including hyperspectral imagery [43] and medical imaging [44]. In modern deep learning architecture, LSTM is usually used to capture long-term dependencies or spatio-temporal information. Thus, LSTM has provided a better solution leveraging long-term information for post-processing.

III. APPLICATION OF VARIOUS GRAPH MODELS

To shed light on the applications of various graph-based methods, we provide a novel categorization that explores the graph-based models from both theoretical and application perspectives. Table 1 presents an useful resource of these methods covering the real-world applications, which covers the applications of both theoretical and application-oriented methods. It is to be noted that the table 1 not only provides a collection of GNN-based methods but also show the core algorithms for proper understanding of these state-of-the-art methods. In Fig. 5, we have shown our featured taxonomy for graph-based strategies. From Fig. 5, it can be noticed that the proposed taxonomy divides all graph-based methods into three branches – 1) Graph theory as the data-driven graph strategies, 2) Methods focusing on real-world problems, and 3) General application-oriented methods. A brief description of these three sections are described as below:

- **Data-driven graph exploitation methods.** This branch provides a theoretical perspective of different solutions for graph exploitation such as pooling-based, attention-based strategies. The graph theory-based branch focuses on exploiting different algorithms for variant graph-oriented tasks including classification, clustering and generation. Pooling-based methods provide different pooling operations for graph exploitation.
- **Real-world application-oriented methods.** In this part, we focus on the methods that prioritize real-world applications including knowledge-based solutions and computer vision strategies. As graph-based networks have been highly appreciated among computer vision researchers, they are also well known for the variety of their applications solving many real-world problems.
- **General application-oriented methods.** Despite having various applications on real-world problems, graph-based models have also been used to exploit different branches of theoretical problems. This branch explains the adaptation of graph networks in other sectors including reinforcement learning, graph generation and graph clustering, etc.

IV. OVERVIEW OF THE THEORETICAL AND APPLICATION-ORIENTED METHODS

This section provides a number of methods related to the three branches from graph network categorization. This section divides these branches into two sub-section for better understanding.

A. DATA-DRIVEN METHODS FOR GRAPH EXPLOITATION

In this sub-section, we provide a number of graph-data exploitation methods for non-euclidean space data exploration. These data-driven methods include different types of graph-based methods as well as some of their limitations.

a: GRAPH NEURAL NETWORK

Scarselli *et al.* [45] first introduced the graph network leveraging neural network for data exploration in the graph

TABLE 1. Various graph structured networks and their applications.

Types	References	Algorithm	Applications
Science (Chemistry & Biology)	Duvenaud *et al.* (2015) [55]	GCN	Learning Molecular Fingerprints
	Zitnik *et al.* (2018) [56]	GCN	Modeling Polypharmacy Side Effects
	Knyazev *et al.* (2018) [57]	GCN	Exploiting Relationships in Molecules
	Jin *et al.* (2018) [58]	GAN	Molecular Optimization
	Xu *et al.* (2019) [59]	GCN	Structured Entity Interactions Prediction
	Dai *et al.* (2019) [60]	GNN	Retrosynthesis Prediction
Traffic Network	Yu *et al.* (2017) [61]	GCN	Traffic Forecasting
	Guo *et al.* (2019) [62]	GCN	Traffic Flow Forecasting
	Wang *et al.* (2019) [63]	GCN	Passenger Demand Modeling
	Hu *et al.* (2019) [64]	GCN	High Resolution Routing
	Bai *et al.* (2019) [65]	GCN	Multi-step Passenger Demand Forecasting
	Zhang *et al.* (2020) [66]	GNN	City-Wide Parking Availability Prediction
Social Network	Li *et al.* (2019) [67]	GNN	Political Perspective Detection in News Media
	Peng *et al.* (2019) [68]	GCN	Fine-grained Event Categorization
	Wu *et al.* (2020) [69]	GCN	Social Spammer Detection
	Bian *et al.* (2020) [70]	GCN	Rumor Detection
Computer Network	Rusek *et al.* (2019) [71]	GNN	Network modeling and Optimization in SDN
Graph Generation	Ma *et al.* (2018) [72]	VAE	Semantically Valid Graphs
	Li *et al.* (2018) [73]	GNN	Graph Generation
	De *et al.* (2018) [74]	GAN	Small molecular graphs
	Bojchevski *et al.* (2018) [75]	GAN	Graph Generation
	Grover *et al.* (2019) [76]	VAE	Graph Generation
	Grover *et al.* (2019) [77]	GNN	Code Generation
	Liao *et al.* (2019) [78]	RNN	Graph Generation
Combinatorial Optimization	Nowak *et al.* (2017) [79]	GNN	Quadratic Assignment
	Li *et al.* (2018) [80]	GCN	Combinatorial Optimization
	Kool *et al.* (2018) [81]	Attention Network	Attention Solves
	Prates *et al.* (2019) [82]	GNN	NP-Complete Problems
	Sato *et al.* (2019) [83]	GNN	Combinatorial Problems
	Luo *et al.* (2020) [84]	GNN	Topological Denoising
Graph Classification	Sun *et al.* (2019) [85]	GCN	Molecular Property Prediction & Graph Classification
	Zheng *et al.* (2020) [86]	GAN	Node Classification
	Luo *et al.* (2020) [87]	GNN	Arbitrary Machine Leaning Tasks
	Wang *et al.* (2020) [88]	VAE	Graph Clustering & Classification
	Xu *et al.* (2019) [89]	GRU+Attention Module	Node Classification
	Yu *et al.* (2019) [90]	GAN+Attention Module	Link Prediction Learning
	Xu *et al.* (2019) [91]	GRU+Attention Module	Node Classification
	Bai *et al.* (2019) [92]	Attention Network	Graph Classification
	Al *et al.* (2019) [93]	GNN + Attention Module	Graph Classifcation
	Peng *et al.* (2020) [94]	GCN + Attention Module	Graph Classification
	Wei *et al.* (2020) [95]	GNN	Probabilistic Type Inference
Reinforcement Learning	Wang *et al.* (2019) [96]	MDP	Node Classification
	Zambaldi *et al.* (2018) [97]	DRL	Reasoning Inter-Object Relations
	Ammanabrolu *et al.* (2018) [98]	DRL	Playing Text-Adventure Games
	Liu *et al.* (2020) [99]	GNN + Attention Module	Multi-Agent Game Abstraction
	Chen *et al.* (2019) [100]	DRL	Natural Question Generation
	Paliwal *et al.* (2019) [101]	Genetic Algorithm + RL	Optimizing Computation Graphs
Program Representation	Allamanis *et al.* (2017) [102]	Gated GNN	Represent Programs
	Cvitkovic *et al.* (2019) [103]	GNN	Graph-Structured Cache
Graph Matching	Fey *et al.* (2020) [104]	GNN (SpliceCNN)	Deep Graph Matching Consensus
Adversarial Attack	Zugner *et al.* (2018) [105]	GNN	Graph Structured Data
	Dai *et al.* (2018) [106]	GNN	Graph Structured Data
	Wu *et al.* (2019) [107]	GCN	Attack and Defense
	Xu *et al.* (2019) [108]	GNN	Topology Attack and Defense
	Zhu *et al.* (2019) [109]	GCN	Adversarial Attack
	Bojchevski *et al.* (2019) [110]	GNN	Node Embeddings
	Bojchevski *et al.* (2019) [111]	GNN	Graph Perturbations

TABLE 1. ***(Continued.)*** **Various graph structured networks and their applications..**

Types	References	Algorithm	Applications
Natural Language Processing	Johnson *et al.* (2016) [112]	Graphical State Transitions	Discovering the rules for governing cellular automation
	Peng *et al.* (2017) [113]	Graph LSTM	Cross-Sentence Relation Extraction
	Bastings *et al.* (2017) [114]	Graph Convolutional Encoders	Machine Translation
	Marcheggiani *et al.* (2017) [115]	GCN	Semantic Role Labeling
	Nguyen *et al.* (2018) [116]	GCN	Event Detection
	Song *et al.* (2018) [117]	GNN	Multi-hop Reading Comprehension
	Zhang *et al.* (2018) [118]	GCN	Relation Extraction
	Song *et al.* (2018) [119]	Graph-State LSTM	Relation Extraction
	Song *et al.* (2018) [120]	LSTM	AMR-to-Text Generation
	Beck *et al.* (2018) [121]	Gated GNN	Text Generation
	Zhang *et al.* (2018) [122]	Sentence-State LSTM	Text Representation Learning
	Rahimi *et al.* (2018) [123]	GCN	User Geolocation
	Sorokin *et al.* (2018) [124]	Gated GNN	Question-Answering
	Zayats *et al.* (2018) [125]	Graph-Structured LSTM	Conversation Modeling
	Palm *et al.* (2018) [126]	Recurrent Relational Networks	Puzzle Solving, Question-Answering
Knogledge-Based	Schlichtkrull *et al.* (2018) [127]	GCN	Modeling Relational Data
	Wang *et al.* (2018) [128]	GCN	Knowledge Graph Alignment
	Kim *et al.* (2018) [129]	Dynamic Graph Generation Network	Relational Knowledge Generation
	Park *et al.* (2019) [130]	GNN	Node Importance Estimation
	Nathani *et al.* (2019) [131]	Graph Attention Network	Relation Prediction
	Xu *et al.* (2019) [132]	Graph Matching Neural Network	Knowledge Graph Alignment
	Xu *et al.* (2019) [133]	GNN	Knowledge Graph Reasoning
	Shang *et al.* (2019) [134]	Structure-Aware Convolutional Networks	Knowledge-Base Completion
	Wang *et al.* (2019) [135]	Logic Attention Based Neighborhood Aggregation	Knowledge Graph Embedding
	Zhang *et al.* (2020) [136]	GNN	Logic Reasoning
Computer Vision	Teney *et al.* (2017) [137]	Gated Recurrent Unit	Visual Question-Answering
	Qi *et al.* (2017) [138]	3D GNN	Semantic Segmentation
	Li *et al.* (2017) [139]	GNN	Situation Recognition
	Qi *et al.* (2018) [140]	Graph Parsing Neural Networks	Human-Object Interactions
	Norcliffe *et al.* (2018) [141]	GCN + Graph Learner Module	Interpretable Visual Question Answering
	Narasimhan *et al.* (2018) [142]	GCN	Factual Visual Question Answering
	Wang *et al.* (2018) [143]	Graph Reasoning Model	Social Relationship Understanding
	Guo *et al.* (2018) [144]	GCN	Action Recognition
Few-shot and Zero-shot Learning	Lee *et al.* (2018) [145]	GRU	Structured Knowledge Graphs
	Gidaris *et al.* (2019) [146]	GNN	Generating Classification Weights
	Liu *et al.* (2019) [49]	Attention Network	Graph Meta-Learning
	Liu *et al.* (2020) [147]	K-Nearest Neighbor	Attribute Propagation Network
	Yao *et al.* (2020) [148]	GNN	Knowledge Transfer
	Chauhan *et al.* (2020) [149]	GNN	Graph Spectral Measurement
	Baek *et al.* (2020) [150]	GNN	Extrapolate Knowledge
Recommendation Systems	Berg *et al.* (2017) [151]	GCN	Matrix Completion
	Ying *et al.)* (2018) [152]	GCN	Web-Scale Recommender Systems
	Zhang *et al.* (2019) [153]	Star-GCN	Recommender Systems.
	Wang *et al.* (2019) [154]	Distilling GCN	Collaborative Filtering
	Xu *et al.* (2019) [155]	Self-Attention Network	Session-based Recommendation
	Gong *et al.* (2019) [156]	Maximal Clique Optimization	Exact-K Recommendation
	Wang *et al.* (2019) [157]	Attention Network	Exact-K Recommendation
	Wang *et al.* (2019) [158]	GCN	Recommender Systems
	Zhang *et al.* (2019) [159]	GNN	Inductive Matrix Completion
	Chen *et al.* (2020) [160]	Linear Residual GCN	Collaborative Filtering
Science (Physics)	Raposo *et al.* (2017) [161]	GNN	Predicting Scene Images
	Raposo *et al.* (2017) [161]	VAE	Objects Discovery and Relations
	Santoro *et al.* (2017) [162]	Relation Networks	Relational Reasoning
	Watters *et al.* (2017) [163]	Interaction Networks	Physics Simulator from Video
	Xu *et al.* (2019) [59]	GNN	Interacting Systems
	Wang *et al.* (2020) [164]	GNN	Climate Observation
Security Analysis	Zhou *et al.* (2019) [165]	GNN	Effective Vulnerability Identification
	Tehranipoor *et al.* (2019) [166]	RNN	Breaking Logic Encryption
	Chen *et al.* (2020) [167]	GCN	Obfuscation Policy Development
	Wang *et al.* (2020) [168]	GCN	Data Management & Anomaly Detection
	Pei *et al.* (2020) [169]	GCN	Malware Detection

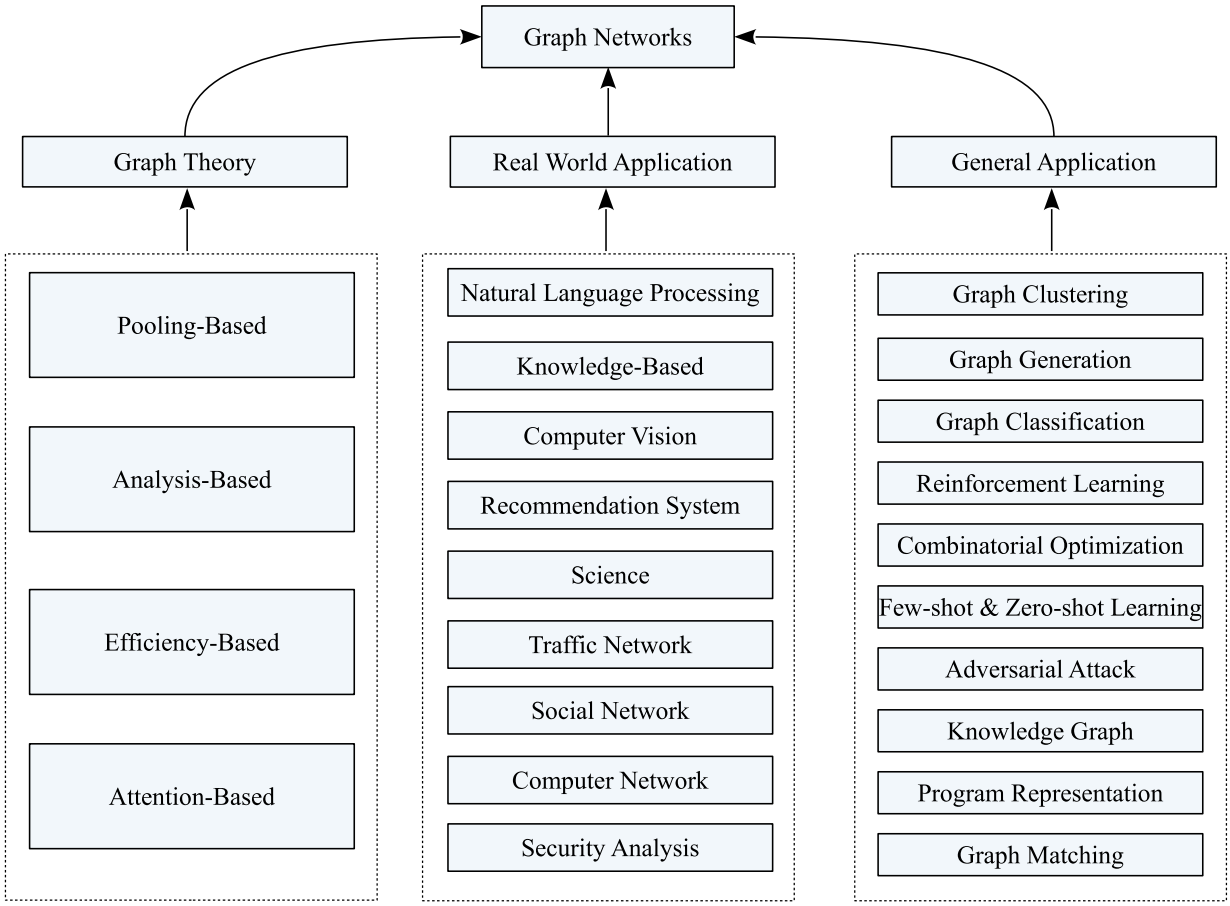


FIGURE 5. **Overview of our proposed categorization of graph networks.**

domain. GNN model tends to learn a featured embedding that is composed of nodes' neighbourhood information. For a node classification problem, the individual node v is defined by its feature and a ground truth label G. The GNN targets to predict the labels of unlabeled data. Mathematically, it can be represented as:

$$h_v = f(y_v, y_{co[v]}, h_{ne[v]}, y_{ne[v]}) \tag{10}$$

$y_{co[v]}$ defines the representations of the edges attached with v and $h_{ne[v]}$ and $y_{ne[v]}$ define the embedding and features of the neighbourhood nodes of v respectively. The f function defines the transition function that transforms the input vector into a d dimensional space. In order to get a particular solution for h_v, the Banach fixed point theorem can be applied [46], then equation (10) goes through an iterative update process that can be expressed as:

$$H^{t+1} = F(H^t, Y) \tag{11}$$

This process is known as message passing or neighborhood aggregation. Now, the output function of GNN becomes:

$$o_v = G(h_v, y_v) \tag{12}$$

$F()$ and $G()$ functions are feed-forward fully connected network. $l1$ loss function is minimized via gradient descent that can be defined as:

$$loss = \sum_{j=1}^{p} (t_j - o_j) \tag{13}$$

GNN has shown significant performance utilizing the data-driven strategy for graph-structured data. Despite the outstanding performance, this method does not learn long-term information and focuses on all nodes equally. Thus, many irrelevant node are also considered for the desired task. Moreover, GNN has an arbitrary inductive bias, that can cause poor generalization while applying GNN on other kinds of data e.g. images, videos.

b: GRAPH CONVOLUTIONAL NETWORK

GCN makes an effort by combining graph nets and CNN together in order to exploit the power of CNN and applying to graph reasoning models [47]. The Graph Convolution Network is based on the layer-wise propagation rule. Spectral convolutions on graphs defined as the multiplication of a signal in the Fourier domain:

$$h_\theta * x = Qh\theta Q^x \tag{14}$$

In equation 14, Q is the eigenvector matrix of the normalized graph laplacian $L = I_N - D^{-1/2}AD^{-12} = QQ$,

with a diagonal matrix of its eigenvalues being the graph Fourier transform. GCN as the first approach utilizes convolution operation for exploiting graph-structured data. As this approach relies only on a convolution operation, it focuses on fixed receptive regions learning from fixed feature space. As GCN focuses on the arbitrary inductive bias, more local descriptors along with GCN will improve the overall classification performance.

c: GRAPH ATTENTION NETWORK

In [48], an attention network has been proposed to exploit graph-structured data with an attention mechanism. Graph Attention Network (GAN) works on both inductive and transductive problems leveraging the power of the attention unit to learn from different regions of feature space. In GAN, the input node features can be defined as $i = \vec{i_1}, \vec{i_2}, \ldots \vec{i_N}, \vec{i_j} \in R^F$. Here, F and N are denoted as the number and features of nodes respectively. To perform the self-attention mechanism, the initial input features need to have a weighted shared transformation. After the transformation, we get a new set of features defined as $F^{'}$, $i' = \vec{i'_1}, \vec{i'_2}, \ldots, \vec{i'_N}, \vec{i'_j} \in R^{F'}$, as output and the weight matrix can be denoted as $W \in R^{\vec{F} \times F}$.

Now, the self-attention transformation becomes $a : R^{\vec{F}} \times R^{\vec{F'}} \rightarrow R$ and the co-coefficients can be expressed as:

$$\mathbf{e_j k} = a(W\vec{i_j}, W\vec{i_k}) \tag{15}$$

This mechanism learns the relative positional vector among i and j suppressing minor structural details. The computed attention neighborhood learns across various nodes all using softmax function.

$$\alpha_j k = softmax_k(e_j k) = \frac{exp(e_j k)}{\sum_{l \in \mathcal{N}_j} exp(e_j l)} \tag{16}$$

As the self-attention mechanism is computationally costly, this method will perform better in large scale graph data rather than all sorts of non-euclidean data.

d: GATED PROPAGATION NETWORK

Liu *et al.* [49] proposed a gated propagation network (GPN) that resolves the problem of graph exploitation using the meta-learning approach. This meta-learning solution aggregates messages among classes to generate new classes for classification. Usually, a multi-head attention module is utilized to get the weighted vectors of similar messages looking around the neighborhood. A gate mechanism is implemented to check whether the aggregated messages are from the neighborhood or itself. This mechanism is applied to all classes and for multiple time steps. The attention module learns identical messages and avoids multiple messages from the same classes. Fig. 4 denotes the multi-head attention mechanism learning aggregated messages. Now, for a task T holding a subset of classes y^T and an N-way-K-shot training set $\mathcal{D}^T$. The initial message prototype for each class $y \in \mathcal{Y}^T$ can be calculated by measuring the average of all K-shot examples that belong to the y class [50]. Mathematically,

$$P_y^0 \triangleq \frac{1}{\left|\left\{(x_i, y_i) \in \mathcal{D}^T : y_i = y\right\}\right|} \sum_{(x_i, y_i) \in \mathcal{D}^T, y_i = y} f(x_i) \tag{17}$$

Through each iteration, for each class $y \in \mathcal{Y}^T$ this propagation system is updated with a new prototype P_{yt}. Now, the multi-head attention measuring the neighborhood aggregated messages $\mathcal{N}_y$ and itself can be calculated as:

$$\begin{aligned} \boldsymbol{P}_{\mathcal{N}_v \rightarrow y}^{t+1} &\triangleq \sum_{z \in \mathcal{N}_v} a\left(\boldsymbol{P}_y^t, \boldsymbol{P}_z^t\right) \times \boldsymbol{P}_z^t, \\ a(p, q) &= \frac{\langle h_1(p), h_2(q) \rangle}{\|h_1(p)\| \times \|h_2(q)\|} \end{aligned} \tag{18}$$

In equation (18), the elements Θ^{prop} of the meta-learning parameters Θ are the learning transformations $h_1(\cdot)$ and their parameters $h_2(\cdot)$. To overcome the problem of identical prototype message passing, each class y is sent to its own end-step prototype P_y^t to itself, i.e., $P_{y \rightarrow y}^{t+1} \triangleq P_y^t$. Then the gated mechanism decides whether to accept the message or not P_N^{t+1} from its neighbors or message $P_{y \rightarrow y}^{t+1}$.

$$\begin{aligned} \boldsymbol{P}_y^{t+1} &\triangleq g\boldsymbol{P}_{y \rightarrow y}^{t+1} + (1 - g)\boldsymbol{P}_{\mathcal{N}_y \rightarrow y'}^{t+1}, \\ g &= \frac{\exp\left[\gamma \cos\left(\boldsymbol{P}_y^0, \boldsymbol{P}_{y \rightarrow y}^{t+1}\right)\right]}{\exp\left[\gamma \cos\left(\boldsymbol{P}_y^0, \boldsymbol{P}_y^{t+1}\right)\right] + \exp\left[\gamma \cos\left(\boldsymbol{P}_y^0, \boldsymbol{P}_{\mathcal{N}_y \rightarrow y}^{t+1}\right)\right]} \end{aligned} \tag{19}$$

In the equation, the similarity index between two vector matrix p and q, and γ is denoted as $\cos(p, q)$ giving a smooth probability distribution using the softmax activation. To grab various types of relations for joint propagation, k modules of the attentive and gated mechanism have been used with untied parameters for $h_1(\cdot)$ and $h_2(\cdot)$ [51].

$$\boldsymbol{P}_y^{t+1} = \frac{1}{k} \sum_{i=1}^{k} \boldsymbol{P}_y^{t+1}[i] \tag{20}$$

where the *i-th* head's output is $P_y^{t+1}[i]$ following the same procedure of P_y^{t+1}. The Same procedure has been repeated for T steps to get the final prototype for y class. Mathematically,

$$\boldsymbol{P}_y \triangleq \lambda \times \boldsymbol{P}_y^0 + (1 - \lambda) \times \boldsymbol{P}_y^T \tag{21}$$

The GPN is designed to work in life-long settings that can learn relative tasks at various time steps using a memory of prototypes. These prototypes contain information on previous prototypes. This propagation helps to learn complex architecture and real-world graph problems for many classes. But when the dot-product attention unit cannot learn relevant neighborhood information, GPN cannot be well trained. Moreover, if the initial set of GPN is incorrect, it will suffer from poor results. Aside from these, GPN is capable of learning robust graph data structure building relation among classes and generating classes when possible.

B. DISCUSSION ON DATA-DRIVEN GRAPH-BASED METHODS

The data-driven graph exploitation strategies have entirely removed the traditional hand-crafted graph theory models. But there exist some limitations of graph network-based methods. The intuitive graph representations of GNN do not rely on local information. GNN models usually rely on message passing protocols exploiting graph-structured data. Thus, a very strong permutation invariant function is required in order to exploit variance graph data. Moreover, these data-driven graph-based methods do not consider the potential of hidden states of the nodes for neighborhood information. Existing solutions rely on different hyper-parameter for different information layers. More general settings can be introduced for the stability of node representation learning.

C. APPLICATION-ORIENTED METHODS

This subsection explains some solutions focusing on the general applications and real-world applications of graph networks including deep reinforcement learning (DRL) and NLP.

a: GRAPH ATTENTION CONVOLUTION

In [52], a graph attention convolution (GAC) has been introduced leveraging the robustness of graph attentional convolution for point-cloud segmentation. GAC is fabricated to learn the features from a weighted function $t : \mathbb{P}^c \rightarrow \mathbb{P}^d$ that maps the input features H to a unseen set of vertex representations $S = \{s_1, s_2, \ldots, s_N\}$ and $s_i \in \mathbb{P}^D$. It also maintains the structural connection among output features. Traditional graph models learn fixed neighbor relations among features but GAC learns dynamic neighbor features containing weighted sharing property. The sharing property is calculated using attention unit $\beta : \mathbb{P}^{3+C} \rightarrow \mathbb{P}^D$ relying on the important parts of neighbors. The dynamic features of each neighboring vertex are measured as:

$$\tilde{b}_{ij} = \beta\left(\Delta r_{ij}, \Delta l_{ij}\right), \quad j \in \mathcal{N}(i) \tag{22}$$

The attention weighted vectors are expressed as $\tilde{b}_{ij} = \left[\tilde{b}_{ij,1}, \tilde{b}_{ij,2}, \ldots, \tilde{b}_{ij,K}\right] \in \mathbb{P}^D$ from the vertex j to i. The representations for the mapping function is denoted as $M_g : \mathbb{P}^C \rightarrow \mathbb{P}^D$ for the multilayer perceptrons $\Delta r_{ij} = r_j - r_i$ and $\Delta l_{ij} = M_g\left(l_j\right) - M_g\left(l_i\right)$. The expression β defines the spatial nature of neighborhood vertices. It accelerates to learn useful representations of nodes for graph exploitation. The differentiable attention unit α is composed of multiple multilayer perceptrons expressed as :

$$\alpha\left(\Delta p_{ij}, \Delta l_{ij}\right) = M_\alpha\left(\left[\Delta p_{ij} \| \Delta l_{ij}\right]\right) \tag{23}$$

where the concatenation operation is denoted as $\|$ and M_α defines the multilayer perceptron.

In order to handle different sizes neighbors across various scales and vertices, the attention-aware weights are normalized around all neighbors.

$$b_{ij,d} = \frac{\exp\left(\tilde{b}_{ij,d}\right)}{\sum_{l \in \mathcal{N}(i)} \exp\left(\tilde{b}_{il,d}\right)} \tag{24}$$

where $\tilde{b}_{ij,k}$ is the weighted attention vectors from j to i vertices at d-channel. Thus, the final output of GAC becomes:

$$s_i = \sum_{j \in \mathcal{N}(i)} b_{ij} \odot M_g\left(l_j\right) + a_i \tag{25}$$

where $a_i \in \mathbb{P}^D$ is the learnable bias and $\odot$ defines the Hadamard product, that gives the output of the element-wise multiplication of two vectors. GAT creates a new direction using graph-based convolution strategies for point-cloud segmentation but consumes a lot of memory. A light-weight approach towards segmentation using graph networks will be useful in many terms.

b: GRAPH CONVOLUTION TRANSFORMER

The success of transformer in NLP has led researchers to use the potential of the transformer in graph-structured data. Transformer searches all meaningful representations in the attention spaces. GCT establishes weighted connections among all possible nodes creating a bridge between treatment codes and diagnosis codes. It uses conditional probabilities for fabricating weighted connections. The conditional probabilities can be defined as $\mathbf{P} \in [0.0, 1.0]^{|c| \times |c|}$ and the normalized softmax can be explained as $\left(\frac{\mathbf{R}^{(i)}\mathbf{L}^{0\ i}}{\sqrt{d}}\right)$). It should be noted that the mask M and conditional probabilities P have the same sizes. Taking into account both M and P, GCT tries to build connections by learning useful representations. Therefore, GCT can be formulated as:

$$\hat{A}^{(j)} := \text{softmax}\left(\frac{\mathbf{R}^{(i)}\mathbf{L}^{(i)\top}}{\sqrt{d}} + \mathbf{M}\right) \tag{26}$$

The self-attention measurement can be defined as:

$$\begin{aligned} \mathbf{E}^{(i)} &= \mathbf{F}\mathbf{E}^{(j)}\left(\mathbf{P}\mathbf{E}^{(i-1)}\mathbf{W}_V^{(j)}\right) \text{ when } i = 1 \\ \mathbf{E}^{(i)} &= \mathbf{F}\mathbf{E}^{(i)}\left(\hat{\mathbf{B}}^{(i)}\mathbf{E}^{(i-1)}\mathbf{W}_V^{(i)}\right) \text{ when } i > 1 \end{aligned} \tag{27}$$

Now calculating the loss function:

$$\begin{aligned} L_{reg}^{(j)} &= D_{KL}\left(\mathbf{P} \| \hat{\mathbf{B}}^{(i)}\right) \text{ when } i = 1 \\ L_{reg}^{(j)} &= D_{KL}\left(\hat{\mathbf{B}}^{(i-1)} \| \hat{\mathbf{B}}^{(i)}\right) \text{ when } i > 1 \\ L_{total} &= L_{\text{prediction}} + \lambda \sum_i L_{reg}^{(i)} \end{aligned} \tag{28}$$

In general terms, attention modules are used in the first unit of transformer. These settings enable the transformer to learn sophisticated connections. But GCT utilizes conditional probabilities followed by masked attention units. Therefore, GCT does not need any previously learned distribution for further processing. GCT combines the NLP and non-euclidean space in a supervised setting but a more robust technique is needed for better performance.

c: DEEP ATTENTIONAL EMBEDDED GRAPH CLUSTERING

The Deep Attentional Embedded Graph Clustering method utilizes a graph attentional encoder-decoder architecture for efficient integration of structure and contextual details to gather information about latent vectors [53]. The major problem of graph clustering is the non-existence of label guidance. As the whole process is unsupervised, it becomes a difficult task to learn optimized embedding. To solve this problem, considering these latent vectors, a self-training module is introduced for performance improvement. The total objective loss function is defined in two terms.

$$L_{\text{overall}} = L_{\text{reconstruction}} + L_{\text{clustering}} \tag{29}$$

From the equation (29), it can be observed that the overall loss function is the total sum of the reconstruction and clustering loss. This method explores the graph data in an unsupervised fashion. Using embedding graph clustering, this method has several advantages in exploiting non-euclidean space data but needs more clarification for harvesting robust graph data details.

d: ADAPTIVE GRAPH CONVOLUTION

Li *et al.* [54] proposed a k-order graph convolution operation for solving the attributed graph clustering problem. This method uses high-order graph convolution to collect global contextual information and adaptively finds out the most appropriate order for different graph structures.

$$\bar{X} = \left(J - \frac{1}{2}M_s\right)^k X \tag{30}$$

In equation (30), k is the corresponding integer, M_s represents sate update, J is the initial node data and the corresponding graph filter is:

$$G = \left(J - \frac{1}{2}M_s\right)^k = \left(J - \frac{1}{2}M_s\right)^k U^{-1} \tag{31}$$

The frequency response is calculated as:

$$P(\lambda_q) = (1 - \frac{1}{2}\lambda_q)^k \tag{32}$$

This method proves to provide a better solution in an unsupervised setting boosting the performance in multiple metrics. It uses high-order convolution operation for global details pooling. Despite the performance, it learns from static feature space which makes the network limited to constrained receptive regions.

e: VARIATIONAL GRAPH AUTOENCODERS

Graph VAE is composed of a neural encoder and a neural decoder. The encoder works as an inference layer and the decoder acts as a generative layer. Fig. 6 depicts the structure of the graph VAE model. The adjacency and identity matrix is given as input to the inference layer and a generated adjacency matrix is given as output from the decoder that processes a generated graph. Let denotes an undirected and unweighted graph as $\varsigma = (\nu, \varepsilon)$ with $N = |\nu|$ nodes. Now,

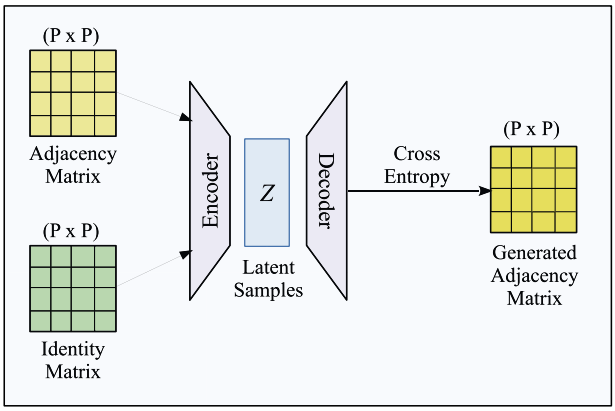


FIGURE 6. Illustration of graph variational autoencoder.

the stochastic latent variables z_i are aggregated by $N \times F$. The parameterized inference becomes:

$$q(Z|Y, A) = \prod_{i=1}^{N} q(z_i|Y, A), \quad \text{with} q(z_i|Y, A) = \mathcal{N}(z_i|\mu_i, diag(\sigma_i^2)) \tag{33}$$

$\mu = GCN_\mu(Y, B)$ can be defined as the mean of vector representations μ_i.Now, the generative VAE becomes:

$$p(B|Z) = \prod_{i=1}^{N}\prod_{j=1}^{N} p(B_{ij}|z_i, z_j), \quad \text{with } p(B_{ij} = 1|z_i, z_j) = \sigma(z_i^T z_j) \tag{34}$$

B_{ij} are the components of $\sigma(\cdot)$ and A (adjacency matrix) followed by a sigmoid activation function. Now, the optimization process can be written as:

$$\mathcal{L} = E_{q(Z|Y,B)}[\log p(A|Z)] - \text{KL}[q(Z|Y, B)\|p(Z)] \tag{35}$$

Now for real-valued label function $l_u : E \rightarrow IR$ on $\mathcal{G}$, which is constrained to take a specific value $l_u(v) = y_{uv}$ at node $v \in V \subseteq E$. In our context, $l_u(v) = 1$ if user finds the item relevant and has engaged with it, otherwise $l_u(v) = 0$. Now let's assume that the adjacency matrix in the KG is similar to the relevant label. So the energy function E becomes:

$$E(l_u, B_u) = \frac{1}{2}\sum_{e_i \in E, e_j \in E} B_u^{ij}(l_u(e_i) - l_u(e_j))^2 \tag{36}$$

This method aims to solve the graph representation learning problem with a generative model but lacks high-quality reconstruction result.

f: CONTEXTUAL GRAPH MARKOV MODEL

Bacciu *et al.* [27] introduced a generative markov model learning cyclic structures in an unsupervised manner. To prior our knowledge, this is the first attempt to use a generative model for variable-length graphs. The generative encoders gather important representations from the data that boosts the classification accuracy leveraging unlabeled examples in a

semi-supervised manner. The model uses an automatic reconstruction of the network in supervised tasks. Locally connected layers have been used without any iteration process. For each layer of this model, let denote $L^{-1}l$ as a set of layers and the present layer as l. $\hat{q}^{L}_{Ne(u)}$ defines the neighborhood nodes of the network for the set of layers $l' \in L^{-1}(l)$. Now, the likelihood $L(\theta|H)$ of the network can be measured as:

$$L(\theta|H) = \prod_{h\in H}\prod_{u=1}^{V_h}\sum_{i=1}^{C} P(y_u|R_u = i)P(R_u = i|\hat{q}^{L^{-1}(l)}_{Ne(u)}) \quad (37)$$

This method uses Markov models for learning graph representation as a generative model. The optimization process showed minimum time complexity but the result can be improved with weighted attention units.

g: GRAPH CONVOLUTIONAL REINFORCEMENT LEARNING

Jiang *et al.* [170] exploited graph convolution in a multi-agent environment. This approach utilizes the relation kernel in order to capture the simulation between agents using relational kernels. To measure the interaction between agents this method adopts a multi-head attention module. This can be explained as:

$$\alpha^{m}_{ij} = \frac{\exp(\tau \cdot X^{m}_{R} h_i \cdot (X^{m}_{L} h_j)^{T})}{\sum_{k\in B_{+i}} \exp(\tau \cdot X^{m}_{R} h_i \cdot (X^{m}_{L} h_L)^{T})} \quad (38)$$

In equation 38, T is denoted as scaling factor. To measure weighted attention, the value of all input features are calculated and joined together for agent i and j. Later, the output of all multi-head attention M are processed through F_c and ReLU activation function to the final output h'_i.

$$h'_i = \sigma(\text{concatenate}[\sum_{j\in B_{+i}} \alpha^{m}_{ij} X^{m}_{V} h_j, \forall m \in M]) \quad (39)$$

At the final phase, this output is fed into temporal regularization and *KL* divergence is calculated. This method combines the reinforcement learning and graph convolution operation. As the multi-head attention module works as relation kernel, a more light-weight approach can be considered to explore the simulation process between agents.

h: ADVERSARIAL TRAINING ON GRAPH

GAN is a kind of generative model based on min-max theory. The generator tries to generate random samples and the discriminator tries to differentiate whether the data sample is from G(fake) or real data. GAN has proved to generate high-resolution photo-realistic images and removed blurry artifacts. Wang *et al.* [171] introduced GraphGAN fusing generative models with graphs. In GraphGAN, the generator tries to calculate the true connectivity distribution and generates important vertices. The discriminator tries to classify the connectivity for the vertex pair. The discriminator gives an output vector calculating the probability of an edge existing. Lt denotes a graph as $G = (L, Q))$, the nodes as $L = \{L_1 \ldots, l_L\}$ and the edges as $E = \{q_{ij}\}^{L}_{i,j=1}$. The probability distribution for nodes can be defined as $p_{\text{true}}(l|l_c)$. Now, for $\mathcal{N}(l_c)$ samples $G(l|l_c; \theta_G)$ tries to learn the underlying distribution of the real samples and $D(l, l_c; \theta_D)$ aims to discriminate whether it is real or not. Mathematically, it can be presented as:

$$\min_{\theta_C}\max_{\theta_D} l(G, D) = \sum_{c=1}^{l}(Q_{l\sim p_{\text{true}}(\cdot|l_q)}[\log D(l, l_c; \theta_D)] + E_{l\sim G(\cdot|l_q;\theta_G)}[\log(1-D(l, l_c; \theta_D))]) \quad (40)$$

Training a GAN model is not an easy task due its convergence failure and mode collapse problem [172]. A more stable adversarial training process for the graph is yet to be explored.

D. DISCUSSION ON APPLICATION-ORIENTED METHODS

Application-oriented methods demonstrated excellent performance in terms of various performance metrics on the graph data analysis. These strategies create a new direction by combining different sections, including generative models, reinforcement learning with graph-structured data as well as also focusing on real-world applications. But these solutions are still in the preliminary stage as they only provided a new way of exploring graph-based networks. Generative models on graphs such as variational autoencoder, Markov models, adversarial networks and sp forth tend to focus on the reconstruction quality of graphs rather than learning the representations of nodes and neighbors.

V. LIMITATIONS & FUTURE DIRECTIONS

In this section, we show the limitations of existing GNN methods and provide a future direction towards the application of GNNs from a theoretical and application perspective.

A. THEORETICAL PERSPECTIVE

In this subsection, we discuss the limitations and future scopes of graph-based methods from a theoretical perspective.

a: ATTENTION MODULES

The potential of the attention network has been previously discussed. But the usage of attention units in GNN is still limited to traditional attention architecture such as self-attention. Despite the superiority of self-attention, it is computationally costly for CPU usage. Moreover, for larger graph inputs self-attention unit becomes very computationally resourceful. Lightweight attention modules should be focused to develop computationally effective graph-based models.

b: MULTISCALE NETWORKS

Multiscale networks learn from various receptive areas in order to perform specific tasks [173]. The multiscale properties harvest intuitive representation from different locations of feature spaces [174]. This makes the network more effective to learn robust node features. Despite the success of multiscale networks in spectral imagery, this feature has been overlooked in graph networks. The usage of multiscale

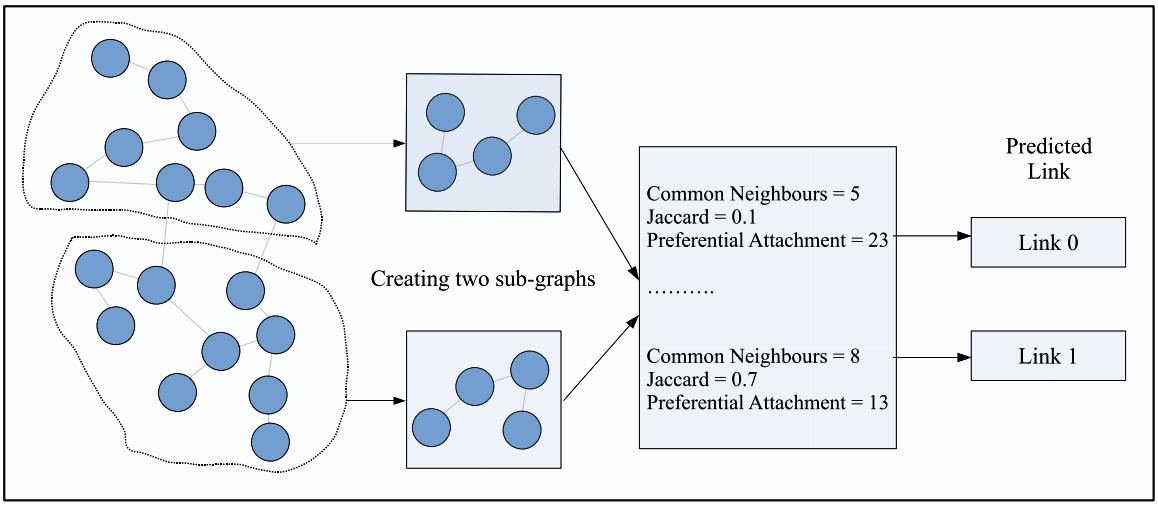


FIGURE 7. Link prediction with graph networks.

networks can be used to produce structural features such as small-world entities as well as sociological interactions.

c: DYNAMIC CONVOLUTION

Existing graph-based models rely on fixed receptive regions for the feature extraction process. This makes the model limited to explore fixed receptive areas (RFs). Dynamic convolution can explore various Rfs harvesting features from various nodes providing better performance. Despite the use of dynamic Rfs in many computer vision applications, it is still being unused in graph exploitation.

d: TRANSFER LEARNING

Inductive learning (IL) in GNN can discover unseen data more effectively than transductive learning. IL can handle graph-based models dynamically that creates this strategy to solve real-world problems [175]. This setting also enables the use of transfer learning (TL) [176]. Transfer learning is an approach that uses a pre-trained model trained on a dataset to use on a different dataset. Despite the success in the euclidean domain, it is still not popular in the non-euclidean domain. The attention-weighted transfer learning approach to graphs will be an interesting concept for non-euclidean space data exploitation.

e: REINFORCEMENT LEARNING

Deep Reinforcement Learning (DRL) is a sub-set of data-driven learning focused on agent interactions in an environment. The power of reinforcement learning creates a new direction for autonomous driving. Fig. 8 shows the basic structure of the RL processing system. In RL, the agent performs direct interaction with the ambiance and increases its efficiency through trial and error. It is worth to be mentioned that while performing this task, there is no need for labeled data. Value function approximation is one of the vital elements of RL that measures the possible return output in order to find the optimal policy. Despite the success of RL in solving various real-life problems the combination of attention unit, multiscale networks, reinforcement learning with graph models is still yet to discover. The representation learning capability of RL methods is not much capable of long-term aggregation. The potential of graph models with aggregated learning with RL can be a sophisticated approach towards GNNs.

B. APPLICATION PERSPECTIVE

This subsection focuses on the application perspective of graph-based models not only pointing to the limitations of these but also providing a future direction for graph researches.

a: LINK PREDICTION

Link prediction is one of the most popular applications of graph-based networks. Fig. 7 shows a general structure of link prediction using a graph neural network. This network converts the data into sub-graphs learning different transformed features from the data. In order to learn the relation between entities, this approach is quite naive to learn long-distance relationships among entities. A Transformer model can be used to alleviate that problem.

b: EEG SIGNAL ANALYSIS

Brain signal decoding has brought much attention and is being used in many applications including brain-computer interface (BCI) and creating connections with peripheral devices. For BCI, the EEG signal is being used by a data-driven strategy for classification purposes. Multiple solutions have been proposed using deep learning techniques for EEG signal classification. But a very few graph-based works are being explored for this task. Robust graph-based generative models can be utilized to learn valuable EEG features for classification purposes.

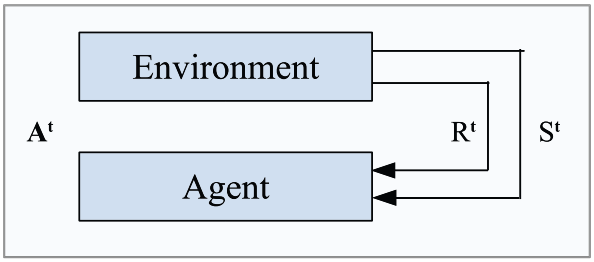


FIGURE 8. Overview of the reinforcement learning process. The state, action and reward of the trial-and-error process in the environment are denoted as S^t, A^t and R^t respectively.

VI. CONCLUDING REMARKS

Recent progress on graph networks has shown superior performance on multiple performance metrics. Motivated by the great success of graph-based models, we provided an application-oriented general categorization of graph-based methods. In this paper, we depict a clear algorithmic review of state-of-the-art methods of graph models presenting various tasks including graph classification, generation, and optimization. Though this paper presents a sophisticated categorization of graph networks, in future works, we will focus on spectral and spatial-based taxonomy of graph-based models.

REFERENCES

[1] W. Yu, W. Cheng, C. C. Aggarwal, K. Zhang, H. Chen, and W. Wang, "NetWalk: A flexible deep embedding approach for anomaly detection in dynamic networks," in *Proc. 24th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Jul. 2018, pp. 2672–2681.

[2] W. Yu, C. Zheng, W. Cheng, C. C. Aggarwal, D. Song, B. Zong, H. Chen, and W. Wang, "Learning deep network representations with adversarially regularized autoencoders," in *Proc. 24th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Jul. 2018, pp. 2663–2671.

[3] W. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 1024–1034.

[4] A. Sanchez-Gonzalez, N. Heess, J. T. Springenberg, J. Merel, M. Riedmiller, R. Hadsell, and P. Battaglia, "Graph networks as learnable physics engines for inference and control," 2018, *arXiv:1806.01242*. [Online]. Available: http://arxiv.org/abs/1806.01242

[5] P. Battaglia, R. Pascanu, M. Lai, D. J. Rezende, and K. Kavukcuoglu, "Interaction networks for learning about objects, relations and physics," in *Proc. Adv. Neural Inf. Process. Syst.*, 2016, pp. 4502–4510.

[6] A. Fout, J. Byrd, B. Shariat, and A. Ben-Hur, "Protein interface prediction using graph convolutional networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 6530–6539.

[7] T. Hamaguchi, H. Oiwa, M. Shimbo, and Y. Matsumoto, "Knowledge transfer for out-of-knowledge-base entities: A graph neural network approach," 2017, *arXiv:1706.05674*. [Online]. Available: http://arxiv.org/abs/1706.05674

[8] M. Zhang and Y. Chen, "Link prediction based on graph neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 5165–5175.

[9] D. Spielman, "Spectral graph theory," Yale Univ., Lect. Notes, 2009, pp. 740–776.

[10] M. Kampffmeyer, Y. Chen, X. Liang, H. Wang, Y. Zhang, and E. P. Xing, "Rethinking knowledge graph propagation for zero-shot learning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 11487–11496.

[11] Y. Zhang, Y. Xiong, X. Kong, S. Li, J. Mi, and Y. Zhu, "Deep collective classification in heterogeneous information networks," in *Proc. World Wide Web Conf.*, 2018, pp. 399–408.

[12] Z. Li, X. Shen, Y. Jiao, X. Pan, P. Zou, X. Meng, C. Yao, and J. Bu, "Hierarchical bipartite graph neural networks: Towards large-scale E-commerce applications," in *Proc. IEEE 36th Int. Conf. Data Eng. (ICDE)*, Apr. 2020, pp. 1677–1688.

[13] H. Peng, J. Li, Y. He, Y. Liu, M. Bao, L. Wang, Y. Song, and Q. Yang, "Large-scale hierarchical text classification with recursively regularized deep graph-CNN," in *Proc. World Wide Web Conf.*, 2018, pp. 1063–1072.

[14] K. S. Tai, R. Socher, and C. D. Manning, "Improved semantic representations from tree-structured long short-term memory networks," 2015, *arXiv:1503.00075*. [Online]. Available: http://arxiv.org/abs/1503.00075

[15] X. Liu, Z. Luo, and H. Huang, "Jointly multiple events extraction via attention-based graph information aggregation," 2018, *arXiv:1809.09078*. [Online]. Available: http://arxiv.org/abs/1809.09078

[16] J. B. Lee, R. A. Rossi, S. Kim, N. K. Ahmed, and E. Koh, "Attention models in graphs: A survey," *ACM Trans. Knowl. Discovery Data*, vol. 13, no. 6, pp. 1–25, 2019.

[17] S. Zhang, H. Tong, J. Xu, and R. Maciejewski, "Graph convolutional networks: A comprehensive review," *Comput. Social Netw.*, vol. 6, no. 1, pp. 1–23, Dec. 2019.

[18] M. M. Bronstein, J. Bruna, Y. LeCun, A. Szlam, and P. Vandergheynst, "Geometric deep learning: Going beyond Euclidean data," *IEEE Signal Process. Mag.*, vol. 34, no. 4, pp. 18–42, Jul. 2017.

[19] S. R. Fahim, D. Datta, M. R. I. Sheikh, S. Dey, Y. Sarker, S. K. Sarker, F. R. Badal, and S. K. Das, "A visual analytic in deep learning approach to eye movement for human-machine interaction based on inertia measurement," *IEEE Access*, vol. 8, pp. 45924–45937, 2020.

[20] S. R. Fahim, S. Niloy, A. H. Shatil, M. R. Hazari, S. K Sarker, and S. K. Das, "An unsupervised protection scheme for overhead transmission line with emphasis on situations during line and source parameter variation," in *Proc. 2nd Int. Conf. Robot., Electr. Signal Process. Techn. (ICREST)*, Jan. 2021, pp. 758–762.

[21] S. R. Fahim, Y. Sarker, M. Rashiduzzaman, O. K. Islam, S. K. Sarker, and S. K. Das, "A human-computer interaction system utilizing inertial measurement unit and convolutional neural network," in *Proc. 5th Int. Conf. Adv. Electr. Eng. (ICAEE)*, Sep. 2019, pp. 880–885.

[22] Q. Kang, S. Yao, M. Zhou, K. Zhang, and A. Abusorrah, "Effective visual domain adaptation via generative adversarial distribution matching," *IEEE Trans. Neural Netw. Learn. Syst.*, early access, Sep. 10, 2020, doi: 10.1109/TNNLS.2020.3016180.

[23] H. Wang, Y. Li, and X. Dong, "Generative adversarial network for desert seismic data denoising," *IEEE Trans. Geosci. Remote Sens.*, early access, Nov. 2, 2020, doi: 10.1109/TGRS.2020.3030692.

[24] X. Wang, S. Feng, and W. Q. Yan, "Human gait recognition based on self-adaptive hidden Markov model," *IEEE/ACM Trans. Comput. Biol. Bioinf.*, early access, Nov. 4, 2019, doi: 10.1109/TCBB.2019.2951146.

[25] X. Xu, J. Li, Y. Yang, and F. Shen, "Towards effective intrusion detection using log-cosh conditional variational AutoEncoder," *IEEE Internet Things J.*, early access, Oct. 29, 2020, doi: 10.1109/JIOT.2020.3034621.

[26] S. R. Fahim, S. K. Das, Y. Sarker, M. R. I. Sheikh, S. K. Sarker, and D. Datta, "A novel wavelet aided probabilistic generative model for fault detection and classification of high voltage transmission line," in *Proc. 2nd Int. Conf. Smart Power Internet Energy Syst. (SPIES)*, Sep. 2020, pp. 94–99.

[27] D. Bacciu, F. Errica, and A. Micheli, "Contextual graph Markov model: A deep and generative approach to graph processing," 2018, *arXiv:1805.10636*. [Online]. Available: http://arxiv.org/abs/1805.10636

[28] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 2672–2680.

[29] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.

[30] A. Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images," Univ. Toronto, Toronto, ON, Canada, Tech. Rep., 2009.

[31] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," 2014, *arXiv:1409.0473*. [Online]. Available: http://arxiv.org/abs/1409.0473

[32] Y. Pang and B. Liu, "SelfAT-fold: Protein fold recognition based on residue-based and motif-based self-attention networks," *IEEE/ACM Trans. Comput. Biol. Bioinf.*, early access, Oct. 22, 2020, doi: 10.1109/TCBB.2020.3031888.

[33] S. R. Fahim, Y. Sarker, S. K. Sarker, M. R. I. Sheikh, and S. K. Das, "Self attention convolutional neural network with time series imaging based feature extraction for transmission line fault detection and classification," *Electr. Power Syst. Res.*, vol. 187, Oct. 2020, Art. no. 106437.

[34] Y. Cai, X. Wang, Z. Yu, F. Li, P. Xu, Y. Li, and L. Li, "Dualattn-GAN: Text to image synthesis with dual attentional generative adversarial network," *IEEE Access*, vol. 7, pp. 183706–183716, 2019.

[35] M. Cao, Y. Zou, D. Yang, and C. Liu, "GISCA: Gradient-inductive segmentation network with contextual attention for scene text detection," *IEEE Access*, vol. 7, pp. 62805–62816, 2019.
[36] Y. Pei, Y. Huang, and X. Zhang, "Consistency guided network for degraded image classification," *IEEE Trans. Circuits Syst. Video Technol.*, early access, Aug. 14, 2020, doi: 10.1109/TCSVT.2020.3016863.
[37] Y. Du, G. Han, Y. Tan, C. Xiao, and S. He, "Blind image denoising via dynamic dual learning," *IEEE Trans. Multimedia*, early access, Jul. 8, 2020, doi: 10.1109/TMM.2020.3008057.
[38] J. Ji, C. Xu, X. Zhang, B. Wang, and X. Song, "Spatio-temporal memory attention for image captioning," *IEEE Trans. Image Process.*, vol. 29, pp. 7615–7628, 2020.
[39] S. R. Fahim, S. K. Sarker, S. K. Das, M. R. Islam, A. Z. Kouzani, and M. A. P. Mahmud, "A residual attention aware network based faults characterization of a power system with SFCL," in *Proc. IEEE Int. Conf. Appl. Supercond. Electromagn. Devices (ASEMD)*, Oct. 2020, pp. 1–2.
[40] P. Ramachandran, N. Parmar, A. Vaswani, I. Bello, A. Levskaya, and J. Shlens, "Stand-alone self-attention in vision models," 2019, *arXiv:1906.05909*. [Online]. Available: http://arxiv.org/abs/1906.05909
[41] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.
[42] F. A. Gers and J. Schmidhuber, "Recurrent nets that time and count," in *Proc. IEEE-INNS-ENNS Int. Joint Conf. Neural Netw. (IJCNN), Neural Comput., New Challenges Perspect. New Millennium*, vol. 3, Jul. 2000, pp. 189–194.
[43] F. Zhou, R. Hang, Q. Liu, and X. Yuan, "Hyperspectral image classification using spectral-spatial LSTMs," *Neurocomputing*, vol. 328, pp. 39–47, Feb. 2019.
[44] M. Z. Islam, M. M. Islam, and A. Asraf, "A combined deep CNN-LSTM network for the detection of novel coronavirus (COVID-19) using X-ray images," *Informat. Med. Unlocked*, vol. 20, Jan. 2020, Art. no. 100412.
[45] F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner, and G. Monfardini, "The graph neural network model," *IEEE Trans. Neural Netw.*, vol. 20, no. 1, pp. 61–80, Jan. 2009.
[46] M. A. Khamsi and W. A. Kirk, *An Introduction to Metric Spaces and Fixed Point Theory*, vol. 53. Hoboken, NJ, USA: Wiley, 2011.
[47] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," 2016, *arXiv:1609.02907*. [Online]. Available: http://arxiv.org/abs/1609.02907
[48] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, "Graph attention networks," 2017, *arXiv:1710.10903*. [Online]. Available: http://arxiv.org/abs/1710.10903
[49] L. Liu, T. Zhou, G. Long, J. Jiang, and C. Zhang, "Learning to propagate for graph meta-learning," in *Proc. Adv. Neural Inf. Process. Syst.*, 2019, pp. 1039–1050.
[50] J. Snell, K. Swersky, and R. Zemel, "Prototypical networks for few-shot learning," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 4077–4087.
[51] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 5998–6008.
[52] L. Wang, Y. Huang, Y. Hou, S. Zhang, and J. Shan, "Graph attention convolution for point cloud semantic segmentation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 10296–10305.
[53] C. Wang, S. Pan, R. Hu, G. Long, J. Jiang, and C. Zhang, "Attributed graph clustering: A deep attentional embedding approach," 2019, *arXiv:1906.06532*. [Online]. Available: http://arxiv.org/abs/1906.06532
[54] R. Li, S. Wang, F. Zhu, and J. Huang, "Adaptive graph convolutional neural networks," 2018, *arXiv:1801.03226*. [Online]. Available: http://arxiv.org/abs/1801.03226
[55] D. K. Duvenaud, D. Maclaurin, J. Iparraguirre, R. Bombarell, T. Hirzel, A. Aspuru-Guzik, and R. P. Adams, "Convolutional networks on graphs for learning molecular fingerprints," in *Proc. Adv. Neural Inf. Process. Syst.*, 2015, pp. 2224–2232.
[56] M. Zitnik, M. Agrawal, and J. Leskovec, "Modeling polypharmacy side effects with graph convolutional networks," *Bioinformatics*, vol. 34, no. 13, pp. i457–i466, Jul. 2018.
[57] B. Knyazev, X. Lin, M. R. Amer, and G. W. Taylor, "Spectral multigraph networks for discovering and fusing relationships in molecules," 2018, *arXiv:1811.09595*. [Online]. Available: http://arxiv.org/abs/1811.09595
[58] W. Jin, K. Yang, R. Barzilay, and T. Jaakkola, "Learning multimodal graph-to-graph translation for molecular optimization," 2018, *arXiv:1812.01070*. [Online]. Available: http://arxiv.org/abs/1812.01070
[59] N. Xu, P. Wang, L. Chen, J. Tao, and J. Zhao, "MR-GNN: Multi-resolution and dual graph neural network for predicting structured entity interactions," 2019, *arXiv:1905.09558*. [Online]. Available: http://arxiv.org/abs/1905.09558
[60] H. Dai, C. Li, C. Coley, B. Dai, and L. Song, "Retrosynthesis prediction with conditional graph logic network," in *Proc. Adv. Neural Inf. Process. Syst.*, 2019, pp. 8872–8882.
[61] B. Yu, H. Yin, and Z. Zhu, "Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting," 2017, *arXiv:1709.04875*. [Online]. Available: http://arxiv.org/abs/1709.04875
[62] S. Guo, Y. Lin, N. Feng, C. Song, and H. Wan, "Attention based spatial-temporal graph convolutional networks for traffic flow forecasting," in *Proc. AAAI Conf. Artif. Intell.*, vol. 33, 2019, pp. 922–929.
[63] Y. Wang, H. Yin, H. Chen, T. Wo, J. Xu, and K. Zheng, "Origin-destination matrix prediction via graph convolution: A new perspective of passenger demand modeling," in *Proc. 25th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Jul. 2019, pp. 1227–1235.
[64] J. Hu, C. Guo, B. Yang, and C. S. Jensen, "Stochastic weight completion for road networks using graph convolutional networks," in *Proc. IEEE 35th Int. Conf. Data Eng. (ICDE)*, Apr. 2019, pp. 1274–1285.
[65] L. Bai, L. Yao, S. S Kanhere, X. Wang, and Q. Z Sheng, "STG2Seq: Spatial-temporal graph to sequence model for multi-step passenger demand forecasting," 2019, *arXiv:1905.10069*. [Online]. Available: http://arxiv.org/abs/1905.10069
[66] W. Zhang, H. Liu, Y. Liu, J. Zhou, and H. Xiong, "Semi-supervised hierarchical recurrent graph neural network for city-wide parking availability prediction," in *Proc. AAAI Conf. Artif. Intell.*, vol. 34, 2020, pp. 1186–1193.
[67] C. Li and D. Goldwasser, "Encoding social information with graph convolutional networks for political perspective detection in news media," in *Proc. 57th Annu. Meeting Assoc. Comput. Linguistics*, 2019, pp. 2594–2604.
[68] H. Peng, J. Li, Q. Gong, Y. Song, Y. Ning, K. Lai, and P. S. Yu, "Fine-grained event categorization with heterogeneous graph convolutional networks," 2019, *arXiv:1906.04580*. [Online]. Available: http://arxiv.org/abs/1906.04580
[69] Y. Wu, D. Lian, Y. Xu, L. Wu, and E. Chen, "Graph convolutional networks with Markov random field reasoning for social spammer detection," in *Proc. AAAI Conf. Artif. Intell.*, vol. 34, 2020, pp. 1054–1061.
[70] T. Bian, X. Xiao, T. Xu, P. Zhao, W. Huang, Y. Rong, and A. Huang, "Rumor detection on social media with bi-directional graph convolutional networks," in *Proc. AAAI Conf. Artif. Intell.*, vol. 34, 2020, pp. 549–556.
[71] K. Rusek, J. Suárez-Varela, A. Mestres, P. Barlet-Ros, and A. Cabellos-Aparicio, "Unveiling the potential of graph neural networks for network modeling and optimization in SDN," in *Proc. ACM Symp. SDN Res.*, Apr. 2019, pp. 140–151.
[72] T. Ma, J. Chen, and C. Xiao, "Constrained generation of semantically valid graphs via regularizing variational autoencoders," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 7113–7124.
[73] Y. Li, O. Vinyals, C. Dyer, R. Pascanu, and P. Battaglia, "Learning deep generative models of graphs," in *Proc. ICLR*, 2018, pp. 1–21.
[74] N. De Cao and T. Kipf, "MolGAN: An implicit generative model for small molecular graphs," 2018, *arXiv:1805.11973*. [Online]. Available: http://arxiv.org/abs/1805.11973
[75] A. Bojchevski, O. Shchur, D. Zügner, and S. Günnemann, "NetGAN: Generating graphs via random walks," 2018, *arXiv:1803.00816*. [Online]. Available: http://arxiv.org/abs/1803.00816
[76] A. Grover, A. Zweig, and S. Ermon, "Graphite: Iterative generative modeling of graphs," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 2434–2444.
[77] M. Brockschmidt, M. Allamanis, A. L. Gaunt, and O. Polozov, "Generative code modeling with graphs," 2018, *arXiv:1805.08490*. [Online]. Available: http://arxiv.org/abs/1805.08490
[78] R. Liao, Y. Li, Y. Song, S. Wang, W. Hamilton, D. Kduvenaud, R. Urtasun, and R. Zemel, "Efficient graph generation with graph recurrent attention networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2019, pp. 4255–4265.
[79] A. Nowak, S. Villar, A. S. Bandeira, and J. Bruna, "A note on learning algorithms for quadratic assignment with graph neural networks," in *Proc. 34th Int. Conf. Mach. Learn. (ICML)*, vol. 1050, 2017, p. 22.
[80] Z. Li, Q. Chen, and V. Koltun, "Combinatorial optimization with graph convolutional networks and guided tree search," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 539–548.

[81] W. Kool, H. van Hoof, and M. Welling, "Attention, learn to solve routing problems!" 2018, *arXiv:1803.08475*. [Online]. Available: http://arxiv.org/abs/1803.08475

[82] M. Prates, P. H. Avelar, H. Lemos, L. C. Lamb, and M. Y. Vardi, "Learning to solve np-complete problems: A graph neural network for decision TSP," in *Proc. AAAI Conf. Artif. Intell.*, vol. 33, 2019, pp. 4731–4738.

[83] R. Sato, M. Yamada, and H. Kashima, "Approximation ratios of graph neural networks for combinatorial problems," in *Proc. Adv. Neural Inf. Process. Syst.*, 2019, pp. 4081–4090.

[84] D. Luo, W. Cheng, W. Yu, B. Zong, J. Ni, H. Chen, and X. Zhang, "Learning to drop: Robust graph neural network via topological denoising," 2020, *arXiv:2011.07057*. [Online]. Available: http://arxiv.org/abs/2011.07057

[85] F.-Y. Sun, J. Hoffmann, V. Verma, and J. Tang, "InfoGraph: Unsupervised and semi-supervised graph-level representation learning via mutual information maximization," 2019, *arXiv:1908.01000*. [Online]. Available: http://arxiv.org/abs/1908.01000

[86] C. Zheng, B. Zong, W. Cheng, D. Song, J. Ni, W. Yu, H. Chen, and W. Wang, "Robust graph representation learning via neural sparsification," in *Proc. Int. Conf. Mach. Learn.*, 2020, pp. 11458–11468.

[87] D. Luo, W. Cheng, D. Xu, W. Yu, B. Zong, H. Chen, and X. Zhang, "Parameterized explainer for graph neural network," 2020, *arXiv:2011.04573*. [Online]. Available: http://arxiv.org/abs/2011.04573

[88] L. Wang, B. Zong, Q. Ma, W. Cheng, J. Ni, W. Yu, Y. Liu, D. Song, H. Chen, and Y. Fu, "Inductive and unsupervised representation learning on graph structured objects," in *Proc. ICLR*, 2020, pp. 1–20.

[89] D. Xu, W. Cheng, D. Luo, Y. Gu, X. Liu, J. Ni, B. Zong, H. Chen, and X. Zhang, "Adaptive neural network for node classification in dynamic networks," in *Proc. IEEE Int. Conf. Data Mining (ICDM)*, Nov. 2019, pp. 1402–1407.

[90] W. Yu, W. Cheng, C. Aggarwal, B. Zong, H. Chen, and W. Wang, "Self-attentive attributed network embedding through adversarial learning," in *Proc. IEEE Int. Conf. Data Mining (ICDM)*, Nov. 2019, pp. 758–767.

[91] D. Xu, W. Cheng, D. Luo, X. Liu, and X. Zhang, "Spatio-temporal attentive RNN for node classification in temporal attributed graphs," in *Proc. IJCAI*, 2019, pp. 3947–3953.

[92] Y. Bai, H. Ding, Y. Qiao, A. Marinovic, K. Gu, T. Chen, Y. Sun, and W. Wang, "Unsupervised inductive graph-level representation learning via graph-graph proximity," 2019, *arXiv:1904.01098*. [Online]. Available: http://arxiv.org/abs/1904.01098

[93] R. Al-Rfou, B. Perozzi, and D. Zelle, "DDGK: Learning graph representations for deep divergence graph kernels," in *Proc. World Wide Web Conf.*, 2019, pp. 37–48.

[94] H. Peng, J. Li, Q. Gong, Y. Ning, S. Wang, and L. He, "Motif-matching based subgraph-level attentional convolutional network for graph classification," in *Proc. AAAI*, 2020, pp. 5387–5394.

[95] J. Wei, M. Goyal, G. Durrett, and I. Dillig, "LambdaNet: Probabilistic type inference using graph neural networks," 2020, *arXiv:2005.02161*. [Online]. Available: http://arxiv.org/abs/2005.02161

[96] L. Wang, Y. Wang, Z. Liang, Z. Lin, J. Yang, W. An, and Y. Guo, "Learning parallax attention for stereo image super-resolution," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 12250–12259.

[97] V. Zambaldi, D. Raposo, A. Santoro, V. Bapst, Y. Li, I. Babuschkin, K. Tuyls, D. Reichert, T. Lillicrap, E. Lockhart, M. Shanahan, V. Langston, R. Pascanu, M. Botvinick, O. Vinyals, and P. Battaglia, "Relational deep reinforcement learning," 2018, *arXiv:1806.01830*. [Online]. Available: http://arxiv.org/abs/1806.01830

[98] P. Ammanabrolu and M. O. Riedl, "Playing text-adventure games with graph-based deep reinforcement learning," 2018, *arXiv:1812.01628*. [Online]. Available: http://arxiv.org/abs/1812.01628

[99] Y. Liu, W. Wang, Y. Hu, J. Hao, X. Chen, and Y. Gao, "Multi-agent game abstraction via graph attention neural network," in *Proc. AAAI*, 2020, pp. 7211–7218.

[100] Y. Chen, L. Wu, and M. J. Zaki, "Reinforcement learning based graph-to-sequence model for natural question generation," 2019, *arXiv:1908.04942*. [Online]. Available: http://arxiv.org/abs/1908.04942

[101] A. Paliwal, F. Gimeno, V. Nair, Y. Li, M. Lubin, P. Kohli, and O. Vinyals, "Reinforced genetic algorithm learning for optimizing computation graphs," 2019, *arXiv:1905.02494*. [Online]. Available: http://arxiv.org/abs/1905.02494

[102] M. Allamanis, M. Brockschmidt, and M. Khademi, "Learning to represent programs with graphs," 2017, *arXiv:1711.00740*. [Online]. Available: http://arxiv.org/abs/1711.00740

[103] M. Cvitkovic, B. Singh, and A. Anandkumar, "Open vocabulary learning on source code with a graph-structured cache," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 1475–1485.

[104] M. Fey, J. E. Lenssen, C. Morris, J. Masci, and N. M. Kriege, "Deep graph matching consensus," 2020, *arXiv:2001.09621*. [Online]. Available: http://arxiv.org/abs/2001.09621

[105] D. Zügner, A. Akbarnejad, and S. Günnemann, "Adversarial attacks on neural networks for graph data," in *Proc. 24th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2018, pp. 2847–2856.

[106] H. Dai, H. Li, T. Tian, X. Huang, L. Wang, J. Zhu, and L. Song, "Adversarial attack on graph structured data," 2018, *arXiv:1806.02371*. [Online]. Available: http://arxiv.org/abs/1806.02371

[107] H. Wu, C. Wang, Y. Tyshetskiy, A. Docherty, K. Lu, and L. Zhu, "Adversarial examples on graph data: Deep insights into attack and defense," 2019, *arXiv:1903.01610*. [Online]. Available: http://arxiv.org/abs/1903.01610

[108] K. Xu, H. Chen, S. Liu, P.-Y. Chen, T.-W. Weng, M. Hong, and X. Lin, "Topology attack and defense for graph neural networks: An optimization perspective," 2019, *arXiv:1906.04214*. [Online]. Available: http://arxiv.org/abs/1906.04214

[109] D. Zhu, Z. Zhang, P. Cui, and W. Zhu, "Robust graph convolutional networks against adversarial attacks," in *Proc. 25th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Jul. 2019, pp. 1399–1407.

[110] A. Bojchevski and S. Günnemann, "Adversarial attacks on node embeddings via graph poisoning," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 695–704.

[111] A. Bojchevski and S. Günnemann, "Certifiable robustness to graph perturbations," in *Proc. Adv. Neural Inf. Process. Syst.*, 2019, pp. 8319–8330.

[112] D. D. Johnson, "Learning graphical state transitions," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2016.

[113] N. Peng, H. Poon, C. Quirk, K. Toutanova, and W.-T. Yih, "Cross-sentence N-ary relation extraction with graph LSTMs," *Trans. Assoc. Comput. Linguistics*, vol. 5, pp. 101–115, Dec. 2017.

[114] J. Bastings, I. Titov, W. Aziz, D. Marcheggiani, and K. Sima'an, "Graph convolutional encoders for syntax-aware neural machine translation," 2017, *arXiv:1704.04675*. [Online]. Available: http://arxiv.org/abs/1704.04675

[115] D. Marcheggiani and I. Titov, "Encoding sentences with graph convolutional networks for semantic role labeling," 2017, *arXiv:1703.04826*. [Online]. Available: http://arxiv.org/abs/1703.04826

[116] T. H. Nguyen and R. Grishman, "Graph convolutional networks with argument-aware pooling for event detection," in *Proc. 32nd AAAI Conf. Artif. Intell.*, 2018, pp. 1–20.

[117] L. Song, Z. Wang, M. Yu, Y. Zhang, R. Florian, and D. Gildea, "Exploring graph-structured passage representation for multi-hop reading comprehension with graph neural networks," 2018, *arXiv:1809.02040*. [Online]. Available: http://arxiv.org/abs/1809.02040

[118] Y. Zhang, P. Qi, and C. D. Manning, "Graph convolution over pruned dependency trees improves relation extraction," 2018, *arXiv:1809.10185*. [Online]. Available: http://arxiv.org/abs/1809.10185

[119] L. Song, Y. Zhang, Z. Wang, and D. Gildea, "N-ary relation extraction using graph state LSTM," 2018, *arXiv:1808.09101*. [Online]. Available: http://arxiv.org/abs/1808.09101

[120] L. Song, Y. Zhang, Z. Wang, and D. Gildea, "A graph-to-sequence model for AMR-to-text generation," 2018, *arXiv:1805.02473*. [Online]. Available: http://arxiv.org/abs/1805.02473

[121] D. Beck, G. Haffari, and T. Cohn, "Graph-to-sequence learning using gated graph neural networks," 2018, *arXiv:1806.09835*. [Online]. Available: http://arxiv.org/abs/1806.09835

[122] Y. Zhang, Q. Liu, and L. Song, "Sentence-state LSTM for text representation," 2018, *arXiv:1805.02474*. [Online]. Available: http://arxiv.org/abs/1805.02474

[123] A. Rahimi, T. Cohn, and T. Baldwin, "Semi-supervised user geolocation via graph convolutional networks," 2018, *arXiv:1804.08049*. [Online]. Available: http://arxiv.org/abs/1804.08049

[124] D. Sorokin and I. Gurevych, "Modeling semantics with gated graph neural networks for knowledge base question answering," 2018, *arXiv:1808.04126*. [Online]. Available: http://arxiv.org/abs/1808.04126

[125] V. Zayats and M. Ostendorf, "Conversation modeling on reddit using a graph-structured LSTM," *Trans. Assoc. Comput. Linguistics*, vol. 6, pp. 121–132, Dec. 2018.

[126] R. Palm, U. Paquet, and O. Winther, "Recurrent relational networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 3368–3378.

[127] M. Schlichtkrull, T. N. Kipf, P. Bloem, R. Van Den Berg, I. Titov, and M. Welling, "Modeling relational data with graph convolutional networks," in *Proc. Eur. Semantic Web Conf.* Cham, Switzerland: Springer, 2018, pp. 593–607.

[128] Z. Wang, Q. Lv, X. Lan, and Y. Zhang, "Cross-lingual knowledge graph alignment via graph convolutional networks," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2018, pp. 349–357.

[129] D. Kim, Y. Yoo, J. Kim, S. Lee, and N. Kwak, "Dynamic graph generation network: Generating relational knowledge from diagrams," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 4167–4175.

[130] N. Park, A. Kan, X. L. Dong, T. Zhao, and C. Faloutsos, "Estimating node importance in knowledge graphs using graph neural networks," in *Proc. 25th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Jul. 2019, pp. 596–606.

[131] D. Nathani, J. Chauhan, C. Sharma, and M. Kaul, "Learning attention-based embeddings for relation prediction in knowledge graphs," 2019, *arXiv:1906.01195*. [Online]. Available: http://arxiv.org/abs/1906.01195

[132] K. Xu, L. Wang, M. Yu, Y. Feng, Y. Song, Z. Wang, and D. Yu, "Cross-lingual knowledge graph alignment via graph matching neural network," 2019, *arXiv:1905.11605*. [Online]. Available: http://arxiv.org/abs/1905.11605

[133] X. Xu, W. Feng, Y. Jiang, X. Xie, Z. Sun, and Z.-H. Deng, "Dynamically pruned message passing networks for large-scale knowledge graph reasoning," 2019, *arXiv:1909.11334*. [Online]. Available: http://arxiv.org/abs/1909.11334

[134] C. Shang, Y. Tang, J. Huang, J. Bi, X. He, and B. Zhou, "End-to-end structure-aware convolutional networks for knowledge base completion," in *Proc. AAAI Conf. Artif. Intell.*, vol. 33, 2019, pp. 3060–3067.

[135] P. Wang, J. Han, C. Li, and R. Pan, "Logic attention based neighborhood aggregation for inductive knowledge graph embedding," in *Proc. AAAI Conf. Artif. Intell.*, vol. 33, 2019, pp. 7152–7159.

[136] Y. Zhang, X. Chen, Y. Yang, A. Ramamurthy, B. Li, Y. Qi, and L. Song, "Efficient probabilistic logic reasoning with graph neural networks," 2020, *arXiv:2001.11850*. [Online]. Available: http://arxiv.org/abs/2001.11850

[137] D. Teney, L. Liu, and A. Van Den Hengel, "Graph-structured representations for visual question answering," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 1–9.

[138] X. Qi, R. Liao, J. Jia, S. Fidler, and R. Urtasun, "3D graph neural networks for RGBD semantic segmentation," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 5199–5208.

[139] R. Li, M. Tapaswi, R. Liao, J. Jia, R. Urtasun, and S. Fidler, "Situation recognition with graph neural networks," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 4173–4182.

[140] S. Qi, W. Wang, B. Jia, J. Shen, and S.-C. Zhu, "Learning human-object interactions by graph parsing neural networks," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 401–417.

[141] W. Norcliffe-Brown, S. Vafeias, and S. Parisot, "Learning conditioned graph structures for interpretable visual question answering," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 8334–8343.

[142] M. Narasimhan, S. Lazebnik, and A. Schwing, "Out of the box: Reasoning with graph convolution nets for factual visual question answering," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 2654–2665.

[143] Z. Wang, T. Chen, J. Ren, W. Yu, H. Cheng, and L. Lin, "Deep reasoning with knowledge graph for social relationship understanding," 2018, *arXiv:1807.00504*. [Online]. Available: http://arxiv.org/abs/1807.00504

[144] M. Guo, E. Chou, D.-A. Huang, S. Song, S. Yeung, and A. Fei-Fei, "Neural graph matching networks for fewshot 3D action recognition," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 653–669.

[145] C.-W. Lee, W. Fang, C.-K. Yeh, and Y.-C.-F. Wang, "Multi-label zero-shot learning with structured knowledge graphs," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 1576–1585.

[146] S. Gidaris and N. Komodakis, "Generating classification weights with GNN denoising autoencoders for few-shot learning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 21–30.

[147] L. Liu, T. Zhou, G. Long, J. Jiang, and C. Zhang, "Attribute propagation network for graph zero-shot learning," in *Proc. AAAI*, 2020, pp. 4868–4875.

[148] H. Yao, C. Zhang, Y. Wei, M. Jiang, S. Wang, J. Huang, N. Chawla, and Z. Li, "Graph few-shot learning via knowledge transfer," in *Proc. AAAI Conf. Artif. Intell.*, vol. 34, 2020, pp. 6656–6663.

[149] J. Chauhan, D. Nathani, and M. Kaul, "Few-shot learning on graphs via super-classes based on graph spectral measures," 2020, *arXiv:2002.12815*. [Online]. Available: http://arxiv.org/abs/2002.12815

[150] J. Baek, D. B. Lee, and S. J. Hwang, "Learning to extrapolate knowledge: Transductive few-shot out-of-graph link prediction," 2020, *arXiv:2006.06648*. [Online]. Available: http://arxiv.org/abs/2006.06648

[151] R. van den Berg, T. N. Kipf, and M. Welling, "Graph convolutional matrix completion," 2017, *arXiv:1706.02263*. [Online]. Available: http://arxiv.org/abs/1706.02263

[152] R. Ying, R. He, K. Chen, P. Eksombatchai, W. L. Hamilton, and J. Leskovec, "Graph convolutional neural networks for Web-scale recommender systems," in *Proc. 24th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Jul. 2018, pp. 974–983.

[153] J. Zhang, X. Shi, S. Zhao, and I. King, "STAR-GCN: Stacked and reconstructed graph convolutional networks for recommender systems," 2019, *arXiv:1905.13129*. [Online]. Available: http://arxiv.org/abs/1905.13129

[154] H. Wang, D. Lian, and Y. Ge, "Binarized collaborative filtering with distilling graph convolutional networks," 2019, *arXiv:1906.01829*. [Online]. Available: http://arxiv.org/abs/1906.01829

[155] C. Xu, P. Zhao, Y. Liu, V. S. Sheng, J. Xu, F. Zhuang, J. Fang, and X. Zhou, "Graph contextualized self-attention network for session-based recommendation," in *Proc. IJCAI*, Aug. 2019, pp. 3940–3946.

[156] Y. Gong, Y. Zhu, L. Duan, Q. Liu, Z. Guan, F. Sun, W. Ou, and K. Q. Zhu, "Exact-K recommendation via maximal clique optimization," in *Proc. 25th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Jul. 2019, pp. 617–626.

[157] X. Wang, X. He, Y. Cao, M. Liu, and T.-S. Chua, "KGAT: Knowledge graph attention network for recommendation," in *Proc. 25th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Jul. 2019, pp. 950–958.

[158] H. Wang, F. Zhang, M. Zhang, J. Leskovec, M. Zhao, W. Li, and Z. Wang, "Knowledge-aware graph neural networks with label smoothness regularization for recommender systems," in *Proc. 25th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Jul. 2019, pp. 968–977.

[159] M. Zhang and Y. Chen, "Inductive matrix completion based on graph neural networks," 2019, *arXiv:1904.12058*. [Online]. Available: http://arxiv.org/abs/1904.12058

[160] L. Chen, L. Wu, R. Hong, K. Zhang, and M. Wang, "Revisiting graph based collaborative filtering: A linear residual graph convolutional network approach," in *Proc. AAAI Conf. Artif. Intell.*, vol. 34, 2020, pp. 27–34.

[161] D. Raposo, A. Santoro, D. Barrett, R. Pascanu, T. Lillicrap, and P. Battaglia, "Discovering objects and their relations from entangled scene representations," 2017, *arXiv:1702.05068*. [Online]. Available: http://arxiv.org/abs/1702.05068

[162] A. Santoro, D. Raposo, D. G. Barrett, M. Malinowski, R. Pascanu, P. Battaglia, and T. Lillicrap, "A simple neural network module for relational reasoning," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 4967–4976.

[163] N. Watters, D. Zoran, T. Weber, P. Battaglia, R. Pascanu, and A. Tacchetti, "Visual interaction networks: Learning a physics simulator from video," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 4539–4547.

[164] S. Wang, Y. Li, J. Zhang, Q. Meng, L. Meng, and F. Gao, "PM2.5-GNN: A domain knowledge enhanced graph neural network for PM2.5 forecasting," 2020, *arXiv:2002.12898*. [Online]. Available: http://arxiv.org/abs/2002.12898

[165] Y. Zhou, S. Liu, J. Siow, X. Du, and Y. Liu, "Devign: Effective vulnerability identification by learning comprehensive program semantics via graph neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2019, pp. 10197–10207.

[166] F. Tehranipoor, N. Karimian, M. M. Kermani, and H. Mahmoodi, "Deep RNN-oriented paradigm shift through bocanet: Broken obfuscated circuit attack," in *Proc. Great Lakes Symp. VLSI*, 2019, pp. 335–338.

[167] Z. Chen, G. Kolhe, S. Rafatirad, C.-T. Lu, S. P. D. Manoj, H. Homayoun, and L. Zhao, "Estimating the circuit de-obfuscation runtime based on graph deep learning," in *Proc. Design, Automat. Test Eur. Conf. Exhib. (DATE)*, Mar. 2020, pp. 358–363.

[168] Z. Wang, N. Luo, and P. Zhou, "GuardHealth: Blockchain empowered secure data management and graph convolutional network enabled anomaly detection in smart healthcare," *J. Parallel Distrib. Comput.*, vol. 142, pp. 1–12, Aug. 2020.

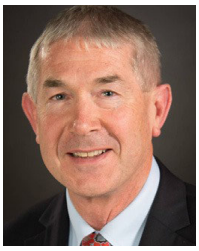
[169] X. Pei, L. Yu, and S. Tian, "AMalNet: A deep learning framework based on graph convolutional networks for malware detection," *Comput. Secur.*, vol. 93, Jun. 2020, Art. no. 101792.

[170] J. Jiang, C. Dun, T. Huang, and Z. Lu, "Graph convolutional reinforcement learning," 2018, *arXiv:1810.09202*. [Online]. Available: http://arxiv.org/abs/1810.09202
[171] H. Wang, J. Wang, J. Wang, M. Zhao, W. Zhang, F. Zhang, X. Xie, and M. Guo, "GraphGAN: Graph representation learning with generative adversarial nets," 2017, *arXiv:1711.08267*. [Online]. Available: http://arxiv.org/abs/1711.08267
[172] M. Arjovsky and L. Bottou, "Towards principled methods for training generative adversarial networks," 2017, *arXiv:1701.04862*. [Online]. Available: http://arxiv.org/abs/1701.04862
[173] Y. Sarker, S. R. Fahim, S. K. Sarker, F. R. Badal, S. K. Das, and M. N. I. Mondal, "A multidimensional pixel-wise convolutional neural network for hyperspectral image classification," in *Proc. IEEE Int. Conf. Robot., Automat., Artif.-Intell. Internet-Things (RAAICON)*, Nov. 2019, pp. 104–107.
[174] Y. Sarker, S. R. Fahim, M. S. Hosen, S. K. Sarker, M. N. I. Mondal, and S. K. Das, "Regularized singular value decomposition based multidimensional convolutional neural network for hyperspectral image classification," in *Proc. IEEE Region Symp. (TENSYMP)*, Jun. 2020, pp. 1502–1505.
[175] M. A. K. Niloy, A. Shama, R. K. Chakrabortty, M. J. Ryan, F. R. Badal, Z. Tasneem, M. H. Ahamed, S. I. Moyeen, S. K. Das, M. F. Ali, M. R. Islam, and D. K. Saha, "Critical design and control issues of indoor autonomous mobile robots: A review," *IEEE Access*, vol. 9, pp. 35338–35370, 2021.
[176] W. Dai, G.-R. Xue, Q. Yang, and Y. Yu, "Transferring naive Bayes classifiers for text classification," in *Proc. AAAI*, vol. 7, 2007, pp. 540–545.

NURUL A. ASIF is currently pursuing the degree with the Department of Mechatronics Engineering, Rajshahi University of Engineering & Technology (RUET), Rajshahi, Bangladesh. His research interests include computer vision, natural language processing, reinforcement learning, robotics, and power system control. His current research interest includes graph neural network methods.

YEAHIA SARKER (Student Member, IEEE) is currently pursuing the bachelor's degree with the Department of Mechatronics Engineering, Rajshahi University of Engineering & Technology (RUET), Rajshahi, Bangladesh. He is also an Active Member of the Control System Research Group, delivering data-driven solutions for control system problems. His research interests include geometric deep learning, remote sensing, generative models, and metric learning. He is also interested in low-level computer vision problems, such as image super-resolution, image denoising, and image dehazing. His current research interest includes meaningful representation learning from spatial and remote sensing images.

RIPON K. CHAKRABORTTY (Member, IEEE) received the B.Sc. and M.Sc. degrees in industrial and production engineering and the Ph.D. degree from the Bangladesh University of Engineering and Technology, in 2009, 2013, and 2017, respectively. He is currently a Lecturer in system engineering and project management and also the Program Coordinator for Master of Decision Analytics and Master of Engineering Science at the School of Engineering and Information Technology, University of New South Wales (UNSW), Canberra, Australia. He is the Group Leader of the Cross-Disciplinary Optimization Under Capability Context Research Team. His research program has been funded by many organizations, such as the Department of Defence-Commonwealth Government, Australia. He has written two book chapters and over 80 technical journal and conference papers. His research interests include wide range of topics in operations research, project management, supply chain management, artificial intelligence, cyber-physical systems, and information systems management.



MICHAEL J. RYAN (Senior Member, IEEE) received the bachelor's, master's, and Ph.D. degrees in engineering. In addition, he has completed two years formal engineering management training in U.K. He is currently a Professor and the Director of the Capability Systems Centre (CSC), University of New South Wales (UNSW), Canberra. He has over 35 years of experience in communications engineering, systems engineering, project management, and management. Since joining UNSW, he has lectured in a range of subjects including communications and information systems, systems engineering, requirements engineering, and project management, and he regularly consults in those fields. He has authored/coauthored 12 books, three book chapters, and over a 250 refereed journals and conference papers. He is a Fellow of Engineers Australia (FIEAust), the International Council on Systems Engineering (INCOSE), and the Institute of Managers and Leaders (FIML), and a Chartered Professional Engineer (CPEng) in electrical and ITEE colleges.

MD. HAFIZ AHAMED received the B.Sc. degree in engineering from the Department of Mechatronics Engineering, Rajshahi University of Engineering & Technology (RUET), Bangladesh, where he is currently pursuing the M.Sc. degree in engineering with the Department of Computer Science and Engineering. He is currently working as a Lecturer with the Department of Mechatronics Engineering, Rajshahi University of Engineering & Technology (RUET). His research interests include machine vision, artificial intelligence, machine learning, robotics, and image processing.

DIP K. SAHA received the B.Sc. degree in mechanical engineering from the Rajshahi University of Engineering & Technology (RUET), where he is currently pursuing the M.Sc. degree in mechanical engineering. He was with the Refrigerator Cooling Design Section, Walton Hi-Tech Industries Ltd., as a Research and Development Engineer. He is currently working as an Assistant Professor with the Department of Mechatronics Engineering, RUET. His research interests include vibration-based condition monitoring, machine learning, and mechatronics.

FAISAL R. BADAL received the B.Sc. degree in engineering from the Department of Mechatronics Engineering, Rajshahi University of Engineering & Technology (RUET). He is currently working as a Lecturer with the Department of Mechatronics Engineering, RUET. His research interests include smartgrid, artificial intelligence, machine learning, natural language processing, and robotics.

SAJAL K. DAS (Member, IEEE) received the Ph.D. degree from the University of New South Wales (UNSW), Australia. He worked as a Research Engineer with the National University of Singapore (NUS), Singapore. He was a Visiting Academic with the University of Newcastle, Australia, and a Faculty Member of the Department of Electrical and Electronic Engineering, American International University, Bangladesh. He is currently the Head of the Department of Mechatronics Engineering, Rajshahi University of Engineering & Technology, Bangladesh, and the Director of the Control System Research Group, RUET. His research interests include renewable energy generation and control, microgrid, smart grid, virtual power plant, cyber-security, and nano-positioning control. He is also the President of the Robotic Society, RUET and Advisor of Robotics and Automation Society, IEEE RUET SB, Bangladesh. He serves as a Guest Editor for the IET Renewable Power Generation, Sustainability, and Energies.

MD. FIROZ ALI is currently working as an Assistant Professor with the Department of Mechatronics Engineering, Rajshahi University of Engineering & Technology (RUET). His research interests include power electronics, control theory, and applications, mechatronics, artificial intelligence.

SUMAYA I. MOYEEN received the B.Sc. degree in computer science and engineering from the Rajshahi University of Engineering & Technology (RUET), Rajshahi, Bangladesh, where she is currently pursuing the M.Sc. degree with the Department of Computer Science and Engineering. Previously, she was working as a Lecturer with the Department of Computer Science and Engineering, North Bengal International University, Rajshahi. In November 2019, she has joined the Department of Mechatronics Engineering, Rajshahi University of Engineering & Technology (RUET), as a Lecturer. Her research interests include machine learning, cloud computing, and the IoT, web development, data mining, and big data, image processing, and artificial Intelligence.

MD. ROBIUL ISLAM (Member, IEEE) is currently pursuing the M.Sc. degree in mechanical engineering from RUET. From February 2015 to August 2018, he was appointed as a Lecturer with the Department of Mechanical Engineering, Bangladesh Army University of Science and Technology (BAUST). He is also working as an Assistant Professor with the Department of Mechatronics Engineering, Rajshahi University of Engineering & Technology (RUET). His research interests include mechatronics systems design, wind turbine aerodynamics, machine learning, and renewable energy.

ZINAT TASNEEM (Member, IEEE) received the M.Sc. degree in electrical and electronic engineering from the Rajshahi University of Engineering & Technology (RUET), in October, 2017. In September 2015, she joined RUET, as a Lecturer. She is currently working as an Assistant Professor with the Department of Mechatronics Engineering, RUET. Her research interests include wind turbine aerodynamics, power electronics in renewable energy technology, and control systems.